NEURAL FIELD TOKENIZATIONS WITH HIERARCHY AND SPATIAL LOCALITY PRIORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural fields parameterize data as functions from coordinates to values, providing a unified framework for representation learning across modalities. Existing approaches are dominated by per-sample meta-learning, which scales poorly due to memory-intensive inner-loop optimization. The natural alternative –feed-forward encoding– typically introduces modality-specific assumptions, sacrificing the generality that makes learning with neural fields attractive. We argue that *locality* and *hierarchy* are useful priors for learning field representations that can be injected without compromising modality-agnosticism. We propose LH-NeF, a framework to learn general-purpose tokenized representations of continuous signals. A locality-preserving hierarchical encoder maps raw coordinate-value field observations to structured tokens, from which the field is reconstructed during training. By replacing meta-learning's inner loop with a single forward pass, LH-NeF uses 42× less memory and supports 133× larger batches than the strongest modality-agnostic baseline. Across images, 3D shapes, and climate fields, our learned representations match or exceed performance of modality-agnostic, modality-specific, and specialized generative neural field baselines on both reconstruction and downstream tasks.

1 INTRODUCTION

Neural fields represent data as continuous functions from coordinates to values parameterized by neural networks that can be queried at arbitrary resolution, e.g. an image as a mapping $f_\theta : [0,1]^2 \to \mathbb{R}^3$. Learning over entire datasets is enabled by conditional neural fields, where a shared network $f_\theta(x; z)$ is conditioned on latent variables z representing each instance in the data (Park et al., 2019; Mescheder et al., 2019). While many methods focused on specific modalities (e.g. 2D grids or radiance fields), Dupont et al. (2022a) proposed ***Functa***, a unified framework to learn latent representations $z \in \mathbb{R}^n$ of neural fields *across modalities*. The resulting latents are then used as data surrogates for arbitrary downstream tasks such as generation and classification (Bauer et al., 2023; Wessels et al., 2025). However, existing modality-agnostic neural field methods face a fundamental tradeoff between introducing useful structure into these representations and preserving the modality-agnosticism that makes them attractive.

In standard representation learning, structure is introduced through encoder design, e.g., local receptive fields in CNN (LeCun et al., 1998), patch tokenization in ViT (Dosovitskiy et al., 2021), message passing in GCN (Kipf & Welling, 2017). These design choices shape the latent space in ways that improve efficiency and generalization (Bronstein et al., 2021), but typically assume specific input structure, which constrains the modality-agnosticism of neural fields when used to obtain the conditioning latent z. As a result, current modality-agnostic neural field methods (Functa and its descendants) avoid obtaining latents through structured encoders entirely, and resort to decoder-only setups where the latents are obtained per-sample through meta-learning (Finn et al., 2017; Dupont et al., 2022a) or auto-decoding (Park et al., 2019). While effective, this reliance on per-sample optimization has practical and theoretical costs.

First, the resulting optimization-derived latent space lacks explicit structure, which makes learning inefficient. Recent methods address this by introducing structure through geometry-grounded mechanisms in the decoding process (Wessels et al., 2025) or modality-specific latent layouts (Bauer et al., 2023). However, these approaches remain tied to optimization-based latent fitting and in many instances sacrifice modality-agnosticism at the cost of adding structure –see Table 1 for an overview.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

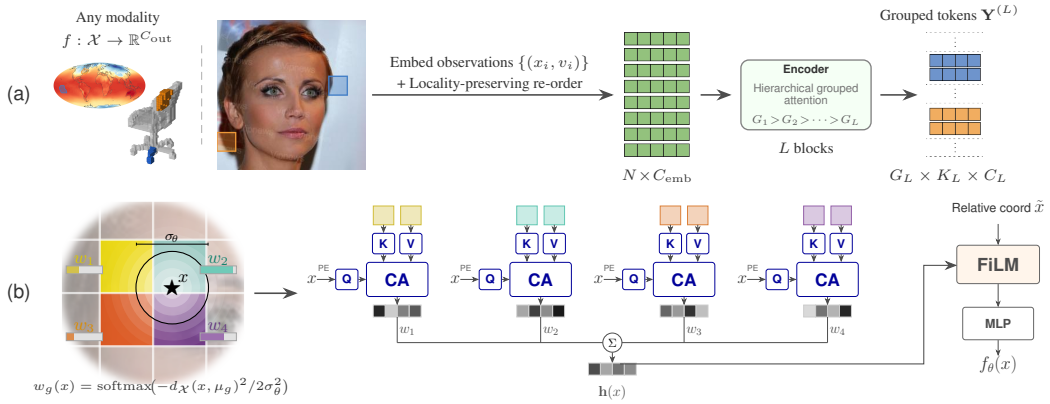


Figure 1: LH-NeF overview. (a) Observations from an input are embedded into the C_{emb}-dimensional tokenizer input space (Appx. D.1) and sorted by a locality-preserving ordering, then processed by L grouped attention blocks, yielding the tokenized representation $\mathbf{Y}^{(L)}$. The locality-preserving ordering keeps each group's spatial support compact in the coordinate space (■/■). (b) To render the field f_θ at any coordinate x, the renderer routes x to the k nearest groups, weighting each group's contribution via a learnable Gaussian kernel. Per-group cross-attention outputs are then aggregated into $\mathbf{h}(x)$ via a weighted sum, FiLM-conditioned on the query's relative coordinate $\tilde{x}$, and decoded into the field value $f_\theta(x)$.

Second, meta-learning approaches –the dominant paradigm on state-of-the-art methods– require storing the full computation graph in memory across all inner steps to compute second-order gradients. At moderate resolutions, this severely limits batch sizes even at moderate resolutions (Table 5), a major bottleneck for scaling to high resolutions or large datasets. Overall, current approaches face a trade-off between (i) structural priors that aid learning, (ii) modality-agnosticism, and (iii) the scalability bottleneck of training with per-sample optimization.

We argue that *hierarchy* (fine-to-coarse organization) and *locality* (correlation between nearby coordinates) are inductive biases that can be used to *learn continuous signal tokenizations* without compromising modality-agnosticism –both have in fact been proved to be excellent priors across images (LeCun et al., 1998; Vahdat & Kautz, 2020), 3D geometry (Qi et al., 2017; Wang, 2023) and continuous signals in general (Mallat, 1989). To this end, we propose LH-NeF (Locality-preserving Hierarchical Neural Fields). LH-NeF consists of a *tokenizer* that builds upon general perception systems exploiting hierarchy through layered grouped attention (Hierarchical Perceiver (Carreira et al., 2022)), and modifies them to respect coordinate-space locality through locality-preserving reorderings. As a result, the spatial support of each group corresponds to a compact region in the domain (Figure 2). We show that this reordering provably preserves locality: on regular grids every group is a box of near-optimal diameter, and on arbitrary observation sets Morton ordering keeps all but a small, bounded fraction of groups compact (Appx. B). To render field values from the resulting tokenization, LH-NeF defines a *renderer* that exploits this structure through Gaussian soft group routing and group-wise cross-attention. The same architecture handles both Euclidean and Riemannian domains and infers neural field representations in a single forward pass –eliminating the memory bottleneck that limits the scalability of existing modality-agnostic neural field methods.

Our contributions are:

- We identify a tradeoff between scalability, structural priors and modality-agnosticism in neural field representation learning, and show that hierarchy and locality can be introduced without modality-specific architectural assumptions.
- We propose LH-NeF, a modality-agnostic framework for learning neural field representations. The LH-NeF *tokenizer* produces a spatially structured grouped tokenization that conditions a *renderer*, which decodes the field at any coordinate via soft group routing and cross-attention aggregation.
- We demonstrate empirically that LH-NeF stays competitive or exceeds state-of-the-art reconstruction and downstream performance on images, 3D shapes, and climate data, while using $\sim 42\times$ less memory and fitting $133\times$ larger batch sizes than the strongest modality-agnostic baseline.

Our code is publicly available at `link-hidden-for-double-blind-review`.

Table 1: Methods for learning on datasets of neural fields. OPT = optimization-based. † labels used during meta-learning.

Method	Latent structure	Latent inference	Modality	Objective
GEM (Du et al., 2021)	Vector	OPT (autodec.)	Any	
GASP (Dupont et al., 2022b)	Vector	—	Any	Generation
DPF (Zhuang et al., 2023)	—	—	Any	
MWT (Gielisse & van Gemert, 2025)	Weights	OPT (MAML)	Any	Classif.†
NF2Vec (Ramirez et al., 2024)	Vector	OPT (fit+enc.)	3D	
3DS2VS (Zhang et al., 2023)	Vector set	Forward pass	3D	
Spatial Functa (Bauer et al., 2023)	2D vector grid	OPT (MAML)	2D	Repr. learn
Functa (Dupont et al., 2022a)	Vector	OPT (MAML)	Any	
ENF (Wessels et al., 2025)	Point cloud	OPT (MAML)	Any	
LH-NeF (Ours)	Structured tokens	Forward pass	Any	Repr. learn

2 METHOD

We consider the problem of representing a signal as a continuous field $f_\theta : \mathcal{X} \to \mathbb{R}^{C_{\text{out}}}$, where the coordinate domain $(\mathcal{X}, d_\mathcal{X})$ is a metric space of dimension $d = \dim(\mathcal{X})$ equipped with a metric $d_\mathcal{X}$ (e.g., for images, $\mathcal{X} = [-1, 1]^2$ with the Euclidean distance; for climate data, $\mathcal{X} = S^2$ with the Riemannian distance). Given N observations $\{(x_i, v_i)\}_{i=1}^N$ of f with coordinates $x_i \in \mathcal{X}$ and values $v_i = f(x_i) \in \mathbb{R}^{C_{\text{out}}}$, our goal is to learn a (conditional) neural field f_θ that enables querying the input at any coordinate $x \in \mathcal{X}$.

Overview. Our method has two main components (Fig. 1). First, the LH-NeF *tokenizer* takes observations $\{(x_i, v_i)\}_{i=1}^N$ of a field f and produces a *grouped token representation* $\mathbf{Y}^{(L)}$ with spatial metadata (§2.1), representing the field f. Then, we define a conditional neural field: given a tokenization $\mathbf{Y}^{(L)}$ of f and a query coordinate x, we route x to the spatially relevant groups of $\mathbf{Y}^{(L)}$ and aggregate information via cross-attention to decode the field value $f_\theta(x)$ (§2.2). The pipeline is trained end-to-end by minimizing reconstruction loss between $f_\theta(x)$ and the ground truth $f(x)$.

2.1 LH-NEF TOKENIZER

Our tokenizer builds on Hierarchical Perceivers (HiP) (Carreira et al., 2022), which process input sequences by splitting tokens into contiguous groups, applying attention within each group, and merging groups into coarser ones across blocks. The original formulation operates on generic token sequences with a fixed (flattened) rasterization order. In the neural field setting, each field observation carries a coordinate $x_i \in \mathcal{X}$ that lives on a metric space $(\mathcal{X}, d_\mathcal{X})$. As a result, the contiguous grouping of HiP leads to each group's coordinates occupying a spatial region of $\mathcal{X}$ over which attention exchanges information – essentially *receptive fields*. Under HiP's default raster ordering, these regions induce *slices* over $\mathcal{X}$, e.g., consecutive rows of pixels in a 2D image, or slices in a 3D voxelization as shown in Figure 2 (left). This violates the spatial locality of $(\mathcal{X}, d_\mathcal{X})$, as coordinates that are far in $\mathcal{X}$ share the same token group. Notably, the initial sequence ordering determines each group's spatial support and, by extension, the geometry of the entire downstream hierarchy (Figure 9).

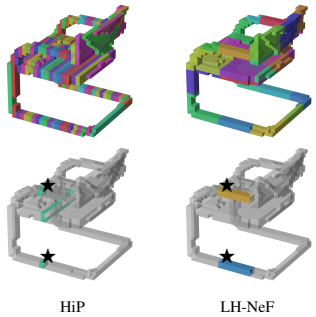


Figure 2: Group assignments on a 3D chair. Top: all groups. Bottom: routed groups for two queries (★).

Building on this observation, we reorder input sequences before grouping using a locality preserving permutation $\pi : \{1, \ldots, N\} \to \{1, \ldots, N\}$ derived from a *hierarchical spatial partition* as explained next. The aim is for contiguous groups to cover spatially compact, small-diameter regions rather than raster slices (■/■ in Figure 1a). Successive encoder blocks merge consecutive groups, combining their spatial supports into a fine-to-coarse hierarchy (Appx. C.1). Appendix B analyzes the resulting group diameters (i.e. measures the locality of the resulting groups) under different ordering and sampling conditions.

Locality-preserving permutation π. Such permutations can be obtained by traversing a hierarchical spatial partition depth-first: list all observations in one region before moving to the next, recursively applying the same rule within each region. Thus, each region's observations occupy consecutive positions in the sequence. For Euclidean domains $\mathcal{X} \subseteq \mathbb{R}^d$, we consider two established constructions: Morton ordering and k-d trees (Appx. A.1; Table 4). More generally, on compact, connected Riemannian manifolds without boundary, hierarchical partitions can be constructed using the geodesic distance, with region diameters controlled in that same distance (Hytönen & Kairema, 2010, Theorem 2.2). Traversing these partitions provides an intrinsic ordering even when no domain-specific hierarchy is readily available (Appx. B.5). In the case of our ERA5 experiment, we use the existing S2 cell hierarchy, a standard hierarchy for this domain (Appx. A.2).

Appendix B establishes locality guarantees under different ordering and sampling conditions. In particular, on regular grids with 2^b points per axis, Morton and k-d-tree orderings admit a closed form for the group diameters and achieve the optimal $G^{-1/d}$ scaling up to dimension-dependent constants. For arbitrary observation sets, we give an explicit, deterministic upper bound on the fraction of Morton groups exceeding a specified diameter threshold, independently of how the observations are distributed, and an analogous bound for the intrinsic manifold construction (Appx. B.5).

After reordering, we embed the sequence as $e_i = \mathrm{Proj}_v(v_{\pi(i)}) + \mathrm{PE}(x_{\pi(i)})$, where Proj_v is a learned projection, $\pi(i)$ denotes the original index of the observation placed at position i in the reordered sequence, and PE is a sinusoidal positional encoding (Vaswani et al., 2017) applied to the chosen coordinate representation. We then form groups by taking contiguous chunks of this reordered sequence and process it through L grouped attention blocks as in HiP. We denote by $\mathcal{I}_g^{(\ell)} \subseteq \{1, ..., N\}$ the set of (reordered) sequence indices that fall into group g at block ℓ; from here on, x_i and v_i denote the coordinate and value at position i of the reordered sequence.

Grouped tokens with spatial metadata. The output of each block ℓ is a set of grouped tokens $\mathbf{Y}^{(\ell)}=\{\mathbf{Y}_g^{(\ell)}\}_{g=1}^{G_\ell}$, $\mathbf{Y}_g^{(\ell)} \in \mathbb{R}^{K_\ell \times C_\ell}$. Note that each group g is a spatially local subset of the *observed* coordinate-value input pairs and is therefore input-adaptive. The final block output $\mathbf{Y}^{(L)}$ is the tokenized representation of the field. In contrast to standard HiP, which collapses onto a single group at the final block (G_L=1), we retain multiple groups at the last block to control the granularity of the resulting tokenization (e.g., G_L=32). We summarize the spatial support of groups $g \in \{1, ..., G_L\}$ at the final block by their centroid μ_g and extent λ_g:

$$\mu_g = \operatorname*{arg\,min}_{y \in \mathcal{X}} \sum_{i \in \mathcal{I}_g} d_{\mathcal{X}}(y, x_i)^2, \qquad \lambda_g = \max_{i \in \mathcal{I}_g} d_{\mathcal{X}}(\mu_g, x_i), \tag{1}$$

where $\mathcal{I}_g = \mathcal{I}_g^{(L)}$ denotes the indices assigned to group g at the final encoder block. The centroid is the Fréchet mean of the group's observed coordinates (the arithmetic mean on Euclidean domains $\mathcal{X} \subseteq \mathbb{R}^d$), and the extent is the group radius. On Euclidean domains, we replace the scalar radius with the axis-aligned bounding-box extent $\lambda_g = \max_i x_i - \min_i x_i \in \mathbb{R}^d_{>0}$, which provides richer information that is not available on general metric spaces. Together with $\mathbf{Y}^{(L)}$, the pairs $\{(\mu_g, \lambda_g)\}_{g=1}^{G_L}$ (*routing metadata*) condition the field renderer as we explain next.

2.2 LH-NeF Renderer

Given conditioning variables $(\mathbf{Y}^{(L)}, \{(\mu_g, \lambda_g)\}_{g=1}^{G_L})$ of f and a query coordinate $x \in \mathcal{X}$, the LH-NeF renderer produces the field value $f_\theta(x)$ in three steps: *(i) routing* identifies which token groups are spatially relevant to x, and assign soft weights to them, *(ii) aggregation* uses cross-attention to read out information of each relevant group and *(iii) modulation* combines per-group outputs and applies a geometry-conditioned modulation to produce the final field value $f_\theta(x)$.

(i) k-Group soft routing. We identify the k groups whose centroids are nearest to x:

$$\mathcal{N}(x) = \operatorname*{arg\,min}_{S \subseteq \{1,\ldots,G\}, |S|=k} \sum_{g \in S} d_{\mathcal{X}}(x, \mu_g)^2, \tag{2}$$

according to the distance $d_{\mathcal{X}}$ in $\mathcal{X}$, and each selected group gets a weight assigned via a Gaussian kernel with learnable bandwidth $\sigma_\theta \in \mathbb{R}_{>0}$:

$$w_g(x) = \frac{\exp\left(-d_{\mathcal{X}}(x, \mu_g)^2 / 2\sigma_\theta^2\right)}{\sum_{g' \in \mathcal{N}(x)} \exp\left(-d_{\mathcal{X}}(x, \mu_{g'})^2 / 2\sigma_\theta^2\right)}, \quad g \in \mathcal{N}(x). \tag{3}$$

Table 2: Reconstruction quality across modalities. Test set PSNR (dB, ↑) for images, IoU (↑) for 3D shapes, MSE (↓) for climate temperatures, and field parameter count (#Param, ↓). **Bold**: best.

	Images						3D Shapes		Data on Manifolds	
	CIFAR-10		CelebA-HQ 64^2		ImageNet1k 256^2		ShapeNet16		ERA5	
Method	PSNR↑	#Param	PSNR↑	#Param	PSNR↑	#Param	IoU↑	#Param	T_t-MSE↓	#Param
Functa	38.1	2.6M	28.0	3.4M‡	–	–	92.1	4.0M‡	5.75E-05	4.1M‡
Spatial Functa†	39.0	425K‡	–	–	**38.4** / 28.9§	1.4M‡	–	–	–	–
ENF	42.2	522K	34.6	3.2M‡	27.5	**817K**‡	92.9	813K‡	8.04E-06	817K‡
LH-NeF (Ours)	**44.4**±0.3	**198K**	**36.1**±0.2	**657K**	28.0±0.3	1.8M	**93.1**±0.2	**649K**	**7.92E-06**±0.90E-06	**307K**

† 2D-grid data only, non modality-agnostic. ‡ Not reported; reproduced from provided code (cf. Appx. G).

§ Spatial Functa uses 65K conditioning dimensions vs. ≤14K for ENF / LH-NeF; at comparable latent budget (16K, $16\times16\times64$), they report 28.9 dB.

The motivation for multi-group routing is *continuity*. With hard routing (k=1), queries would switch abruptly from one group to another as a query moves across the boundary of their receptive fields. This would cause potential discontinuities in the reconstructed field. Multi-group routing ensures f_θ changes *smoothly* across $\mathcal{X}$. A sensitivity analysis on k is provided in Appx. E.3.

While bounded receptive fields already provide coarse locality (each group only "sees" tokens from a spatial region), Gaussian weighting adds *distance-dependent relevance* during inference: groups whose centroids are farther away from the queried coordinate contribute less. The bandwidth σ_θ defines a characteristic scale –the contribution of groups beyond $\sim 2\sigma_\theta$ away from the query becomes negligible. We learn σ_θ during training and parameterize it as $\sigma_\theta = \exp(\log \sigma_\theta)$ to ensure positivity.

(ii) Cross-attention aggregation. We embed the query coordinate into a query vector:

$$\mathbf{q}(x) = \mathrm{MLP}_q\big(\mathrm{PE}(x)\big) \in \mathbb{R}^D \tag{4}$$

and use cross-attention to extract relevant information from the k-nearest groups according to the queried coordinate x. In particular, for each routed group $g \in \mathcal{N}(x)$, we apply cross-attention with $\mathbf{Y}_g$ as keys/values:

$$\mathbf{h}_g(x) = \mathrm{CrossAttn}\big(\mathbf{q}(x),\ \mathbf{Y}_g\big) \in \mathbb{R}^D. \tag{5}$$

Each group's contribution is then aggregated into a feature vector via a weighted sum:

$$\mathbf{h}(x) = \sum_{g\in\mathcal{N}(x)} w_g(x) \cdot \mathbf{h}_g(x). \tag{6}$$

(iii) Geometry-conditioned modulation. Next, we use FiLM-style modulations (Perez et al., 2018) to condition $\mathbf{h}(x)$ based on the position of the query x *within* its nearest group. On both Euclidean and Riemannian coordinate domains, the query's extent-normalized position relative to its nearest group center μ_{g^*} is:

$$\tilde{x} = \frac{\log_{\mu_{g^*}}(x)}{\lambda_{g^*}}, \tag{7}$$

where $g^* = \arg\min_{g\in\mathcal{N}(x)} d_{\mathcal{X}}(x, \mu_g)$ is the nearest group, and $\log_{\mu_{g^*}}$ is the logarithmic map[1] mapping x into the tangent space at μ_{g^*} (which reduces to $x - \mu_{g^*}$ on Euclidean spaces). This normalized coordinate $\tilde{x}$ then modulates $\mathbf{h}(x)$ as:

$$(\gamma, \beta) = \mathrm{MLP}_{\mathrm{FiLM}}\big(\mathrm{PE}(\tilde{x})\big), \qquad \mathbf{h}'(x) = (1+\gamma)\odot\mathbf{h}(x) + \beta, \tag{8}$$

which lets the renderer adapt its output based on intra-group position (e.g. near vs. far from group boundaries). The final field value is

$$f_\theta(x) = \mathrm{MLP}_{\mathrm{out}}(\mathbf{h}'(x)) \in \mathbb{R}^{C_{\mathrm{out}}}. \tag{9}$$

[1]We assume this map is uniquely defined at the evaluated query-center pairs.

Table 3: Downstream tasks on frozen latent representations. **Bold**: best among modality-agnostic representation learning methods. Underline: best overall.

	Generation (FID↓)		Classification (Acc↑)			Forecasting (MSE↓)
Method	CIFAR-10	CelebA-HQ 64^2	CIFAR-10	ImageNet1k	ShapeNet16	ERA5
GEM‡	23.8	30.4	–	–	–	–
GASP‡	–	13.5	–	–	–	–
DPF‡	15.1	13.2	–	–	–	–
Spatial Functa†	16.5	–	90.3%	80.4%	–	–
Functa	78.2	40.4	68.3%	–	90.3%	3.45E-03
ENF	23.5	33.8	82.1%	–	96.6%	**9.44E-06**
LH-NeF (Ours)	**18.5**	**9.7**	**91.2%**	71.2%	**96.8%**	1.11E-05

† 2D-grid data only, non modality-agnostic. ‡ Generative only, non general repr. learning.

2.3 LH-NeF properties

The design choices above give rise to several properties worth highlighting:

- *Locality* enters at three scales: the tokenizer's coordinate-ordered grouping summarizes each input into spatially compact regions; Gaussian-weighted routing at query time blends information from the k nearest groups; and FiLM modulation provides finer adaptation based on where within the nearest group's receptive field the query lies. Because $\tilde{x}$ is group-normalized, this modulation is invariant to translations and axis-aligned rescalings of the group's support, automatically sharing the same conditioning across groups (Appx. C.2, visualized in Figure 12).
- *Hierarchy.* The tokenizer's layered blocks induce a multi-scale hierarchy that is aligned with coordinate-space geometry; e.g., on dense grids, successive layers produce progressively coarser compact spatial tilings (Appendix C.1 visualizes this for 2D inputs and 3D shapes).
- *Dynamic tokenization.* Because grouping operates *over observations* rather than over the coordinate space itself, the group support (μ_g, λ_g) adapts to the sampling density and geometry of the input. For signals with uniform sampling (e.g., images, voxelized data), the support of groups is consistent for all instances. For signals with non-uniform sampling however (e.g. point clouds), our tokenization adapts to the geometry of each instance (see Figure 11).
- *Generality.* Our framework applies to any modality expressed as coordinate-value observations over Euclidean and Riemannian domains, without requiring a regular grid or a modality-specific encoder.
- *Efficiency.* Our approach replaces second-order MAML's $\mathcal{O}(K \cdot \mathrm{Mem}(\nabla\mathcal{L}))$ inner-loop cost (Rajeswaran et al., 2019) with a single forward-backward pass, yielding $\sim 42\times$ memory reduction and $133\times$ larger batch sizes in practice at 64×64 resolution (§4.3). At inference, k-group routing maintains token cross-attention costs at $\mathcal{O}(kK_LD)$ rather than $\mathcal{O}(GK_LD)$ (often $k \in \{1, 2, 4\}$).

3 Related work

Neural fields and conditional neural fields. Neural fields parameterize signals as continuous functions $f_\theta : \mathbb{R}^d \to \mathbb{R}^c$ via coordinate-based neural networks. Early works fit a separate network per signal (NeRF (Mildenhall et al., 2021), SIREN (Sitzmann et al., 2020)), with acceleration via hash grids (Müller et al., 2022) and other spatial structures (Chen et al., 2022; Barron et al., 2021). To extend to datasets, DeepSDF (Park et al., 2019) introduced Conditional Neural Fields with auto-decoding: a single shared MLP conditioned on per-shape latent codes optimized jointly with network weights. Functa (Dupont et al., 2022a) extended this paradigm to learning modality-agnostic representations for downstream tasks. This paradigm has been widely adopted, e.g., for medical imaging (Friedrich et al., 2026), video (Wolleb et al., 2025; 2026), PDEs (Jo et al., 2026; Knigge et al., 2024), and generation (Du et al., 2021; Dupont et al., 2022b; Zhuang et al., 2023) –with diffusion models trained on conditioning variable spaces (Peebles & Xie, 2023; Ho et al., 2020; Chen et al., 2024; Kim et al., 2023). All modality-agnostic methods in this family rely on per-sample optimization (MAML or auto-decoding) to obtain latents. Several works observe that the resulting latent vectors are difficult to classify directly (Wolleb et al., 2025; Gielisse & van Gemert, 2025), and Gielisse & van Gemert (2025) showed that more fitting steps improve reconstruction but hurt classification. Recent

modality-agnostic works improve performance by adding structure: 2D latent grids (Bauer et al., 2023), geometry-grounded point clouds with symmetry-specific bi-invariants and Gaussian-weighted conditioning (Wessels et al., 2025) or weight-space embeddings directly (Ramirez et al., 2024; Navon et al., 2023; Gielisse & van Gemert, 2025). Akin to our method, 3DShape2VecSet (Zhang et al., 2023) encodes shapes as unstructured latent vector sets, but lacks spatial structure and is modality-specific. At query time, several local conditioning methods read from multiple features via interpolation or nearest neighbors on feature grids (Chen et al., 2021; Lee & Jin, 2022; Yu et al., 2021; Peng et al., 2020). Our renderer applies the same principle to modality-agnostic grouped tokens under a locality-preserving ordering.

Forward-pass inference of neural field representations. Encoder-based methods that bypass per-sample optimization of the field representation exist, but typically in modality-specific settings. An early example is Occupancy Networks (Mescheder et al., 2019), which pair a PointNet/ResNet encoder with a shared occupancy decoder; amortized, but restricted to 3D shapes. In 2D, LIIF (Chen et al., 2021) and LTE (Lee & Jin, 2022) pair CNN encoders with local coordinate-conditioned MLP decoding, conditioning each query on its nearest grid features with distance-based weighting. INFD (Chen et al., 2024) uses a CNN encoder for neural field diffusion and leverages neural field properties to obtain state-of-the-art multi-scale generation. In 3D, pixelNeRF (Yu et al., 2021) and LRM (Hong et al., 2024) condition radiance/tri-plane representations on image features with camera models. ConvOccNet (Peng et al., 2020) uses trilinear interpolation from volumetric feature grids and 3DILG (Zhang et al., 2022) conditions on irregular latent point sets with Gaussian-weighted aggregation. For PDEs, AROMA (Serrano et al., 2024) uses a Perceiver-style encoder. In general, these methods are either tied to a specific input modality or lack a more general structure that enables learning across modalities. LH-NeF bridges this gap. It encodes the field in grouped tokens by operating on the field's coordinate-value observations, without relying on modality-specific components. Outside of neural fields literature, modality-agnostic models exist. Perceiver IO (Jaegle et al., 2022) uses cross-attention to learn in a modality agnostic way. However, its learned latent bottleneck lacks spatial structure and locality. Building on Perceiver IO, Hierarchical Perceiver (HiP) (Carreira et al., 2022) adds hierarchy but violates locality, which our ablations find to significantly hurt performance for data defined on metric spaces that carry a corresponding geometry (Table 4). These methods learn representations in modality-agnostic settings, but can not query inputs at arbitrary resolutions as neural fields allow.

Hierarchy, locality, and dynamic tokenization. Hierarchy and locality are fundamental inductive biases in deep learning. CNNs learn fine-to-coarse features through stacked layers of local filters with increasing receptive fields (LeCun et al., 1998), ViTs introduce locality through explicit patch-based tokenization before applying global self-attention (Dosovitskiy et al., 2021). Swin Transformer (Liu et al., 2021) combines both via hierarchical shifted windows over image patches and HiP (Carreira et al., 2022) generalizes hierarchical processing to arbitrary modalities via grouped hierarchical attention. However, HiP's grouping follows a fixed raster scan that ignores coordinate-space geometry, violating locality. Our approach respects locality and produces tokenized representations whose geometry is input-adaptive and locality-preserving (Figure 11). The benefits of hierarchy extend beyond continuous signals. Recently, H-Net (Hwang et al., 2026) used dynamic chunking to replace fixed tokenization with input-adaptive hierarchical grouping and showed that learned hierarchical representations improve efficiency in large language models. Other work on dynamic (or adaptive) tokenization addresses this in vision settings. TokenLearner (Ryoo et al., 2021) selects tokens via spatial attention, ALIT (Duggal et al., 2025) and ElasticTok (Yan et al., 2025) adapt token count to input complexity, and GPSToken (Zhang et al., 2026) parameterizes tokens as 2D Gaussians. However, all these methods assume fixed modalities (text and images, respectively). LH-NeF explotes structure explicitly at encoding and decoding time, creating spatially compact receptive fields at every level without grid assumptions or modality-specific windowing (Appx. C.1).

4 Experiments

We evaluate LH-NeF across three modalities: 2D images (CIFAR-10 32^2 (Krizhevsky et al., 2009), CelebA-HQ 64^2 (Karras et al., 2018), ImageNet1k 256^2 (Russakovsky et al., 2015)), 3D voxel occupancy (ShapeNet16, 32^3 (Yi et al., 2016)), and global climate fields (ERA5 (Dupont et al., 2022b)). The coordinate dimension d and the locality-preserving ordering π are set by the corresponding coordinate domain. Other hyperparameters are chosen based on best validation loss (full configurations in Appx. D).

We follow the usual two-stage pipeline: *(i)* we fit LH-NeF to the data through a reconstruction objective, and then *(ii)* use the learned tokenizations, obtained through a forward pass on the frozen LH-NeF tokenizer, to solve downstream tasks. We compare against three functa-style neural field representation learning methods: Functa (Dupont et al., 2022a), Spatial Functa (Bauer et al., 2023) and ENF (Wessels et al., 2025). For generation, we additionally compare against dedicated generative neural field based methods (GEM (Du et al., 2021), GASP (Dupont et al., 2022b), DPF (Zhuang et al., 2023)).

4.1 Reconstruction

Table 2 summarizes reconstruction quality across modalities. On both images and 3D shapes, LH-NeF consistently achieves state of the art performance among modality-agnostic neural field methods, while using significantly fewer parameters –up to $5\times$ less on CelebA 64^2. On ImageNet1k 256^2, Spatial Functa reaches 38.4 dB but uses a 2D conditioning latent grid of size $32\times32\times64{=}65,536$. At a comparable conditioning budget of size $16\times16\times64{=}16,384$ (vs. ours of $14,336$), Spatial Functa reaches 28.9 dB and we obtain a competitive 28.0 dB, while providing a general modality-agnostic formulation (Spatial Functa is restricted to 2D signals). On ERA5, LH-NeF is on par with ENF and improves over Functa by a factor of about seven.

4.2 Downstream tasks

We evaluate whether the tokenizations learned by LH-NeF support downstream tasks they were never trained for, by training off-the-shelf models on the frozen tokenizations. Detailed experimental descriptions are provided in Appx. F.

Generation. We train a Diffusion Transformer (Peebles & Xie, 2023) on the (frozen) LH-NeF tokenizations. On CelebA-HQ 64^2, LH-NeF achieves state of the art generation (FID↓), notably outperforming specialized generative methods and modality-agnostic neural field learning baselines (Table 3, samples in Figure 3). On CIFAR-10, we achieve the best generation score among modality-agnostic representation learning methods, while remaining behind Spatial Functa and Diffusion Probabilistic Fields (DPF), which benefit from domain and task-specific advantages respectively.

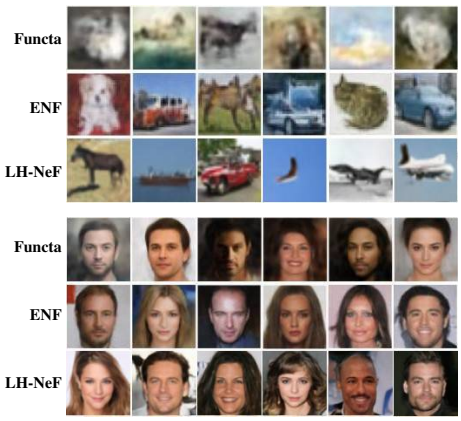


Figure 3: Unconditional generation on CIFAR-10 (top) and CelebA-HQ 64^2 (bottom). Larger visualizations in Fig. 13.

Classification. On ShapeNet16, a lightweight classifier on frozen latents reaches 96.8% accuracy, outperforming ENF (96.6%) and Functa (90.3%). On CIFAR-10, a ConvNeXt (Liu et al., 2022) classifier trained on the data's learned tokenizations reaches 91.2% accuracy, outperforming all modality-agnostic baselines (ENF 82.1%, Functa 68.3%) as well as Spatial Functa (Bauer et al., 2023) (90.3%, modality-specific). On ImageNet-1k, the same classifier design reaches 71.2% top-1 accuracy from 14K-dimensional tokenizations; Spatial Functa reaches 80.4% at a comparable latent budget ($16\times16\times64$), although we note that it benefits from domain specific advantages since it's restricted to 2D grids.

Global temperature forecasting. On ERA5, we follow the ENF protocol: consecutive hourly snapshots are tokenized, a temporal predictor maps $\mathbf{Y}_t \mapsto \mathbf{Y}_{t+1}$ in latent space, and predictions are decoded through the frozen renderer (details in Appx. F.3). LH-NeF improves over Functa by more than two orders of magnitude and over the persistence baseline (2.42×10^{-5}, predicting $T_{t+1} = T_t$), and trails ENF by 17%.

4.3 Ablations and efficiency assessment

We summarize the most informative ablations here and discuss all of them in Appx. E. Locality is the component that matters most (Table 4, first row): replacing our tokenizer with HiP, which groups observations in raster order, costs 6.4 dB PSNR and 16.0 FID on CelebA-HQ and 16.9 IoU on ShapeNet16. The shape of the groups explains this gap. On the 64^2 CelebA-HQ grid, the 64 groups that condition the renderer are 8×8 blocks under Morton ordering but full image rows under raster

Table 4: Component ablation on CelebA-HQ 64^2 (recon PSNR ↑, gen FID ↓) and ShapeNet16 (recon IoU ↑, cls acc ↑). †k-d tree on CelebA, Morton on ShapeNet.

	CelebA-HQ 64^2		ShapeNet16	
Configuration	PSNR ↑	FID ↓	IoU ↑	Acc ↑
Full model	**36.3**	**9.7**	**93.2**	**96.8**
LH-NeF Tokenizer → HiP	29.9 (−6.4)	25.7 (+16.0)	76.3 (−16.9)	95.6 (−1.2)
Other coord. ordering†	35.5 (−0.8)	10.7 (+1.0)	92.5 (−0.7)	96.0 (−0.8)
Gaussian → power weight.	34.1 (−2.3)	13.4 (+3.7)	85.5 (−7.7)	96.8 (0.0)
$k \to 1$ (hard routing)	35.6 (−0.7)	10.2 (+0.5)	84.8 (−8.4)	96.7 (−0.1)
No FiLM modulation	34.0 (−2.3)	13.0 (+3.3)	88.9 (−4.3)	97.0 (+0.2)

Table 5: Training efficiency on CelebA-HQ 64^2 (H100 47 GB MIG). Mem: peak step memory at B=1; Max BS: largest fitting batch.

	Functa	ENF	**LH-NeF**
Training step	MAML	MAML	Fwd. + Bwd.
Mem (MB)	1,583	7,347	**173**
Max BS	18	2	**266**

ordering. All row centroids then lie on the vertical midline of the image, so the routed groups depend only on the row of the query, and the k=4 groups routed to it form a strip four pixels high across the whole image. Counting pixels on the grid, this strip contains on average 54.6% of the 64 pixels nearest to a query, whereas the four Morton blocks routed to it contain 99.95% (Appx. B.4). Which locality-preserving ordering is used matters much less: Morton and k-d tree orderings are within 0.8 dB/IoU, as both provably produce boxes of the same size on regular grids, and differ only in the orientation of rectangular groups (Appx. B.2.2). Multi-group routing matters little on images (0.7 dB) but does on sparse 3D occupancy (8.4 IoU), where groups covering empty space lend capacity to groups on the object surface. Finally, we ablate the tokenizer architecture with a parameter-matched ViT tokenizer whose patches coincide with our groups. This configuration outperforms ours (37.2 vs. 36.3 dB) and is cheaper at this resolution. However, it requires a full regular grid, and its attention cost grows quadratically with the number of observations N whereas ours grows linearly (Appx. E.2).

We benchmark the computational cost of LH-NeF according to the protocol described in Appx. H (Table 5). We use the official JAX implementations for Functa and ENF. LH-NeF uses $\sim 42\times$ less memory (173 MB vs. 7.3 GB) and supports $133\times$ larger batch sizes (266 vs. 2) than ENF on the same GPU. This results from second-order MAML methods requiring maintaining the full inner-loop computation graph across all K inner steps, inflating per-sample memory by a factor of K relative to a single forward-backward pass. Our method fits latents in one forward pass and still matches or improves over baselines in terms of quality.

5 Conclusion

We presented LH-NeF, a framework for learning tokenizations of neural fields across metric coordinate domains that uses hierarchy and locality as inductive biases in its encoding and decoding process. LH-NeF achieves competitive or state-of-the-art performance on reconstruction and downstream tasks across 2D images, 3D shapes, and spherical climate data. By replacing per-sample meta-learning with a learned tokenization, our method uses significantly less memory and facilitates much larger batch sizes. Altogether, LH-NeF serves as a scalable modality-agnostic foundational method for neural field representation learning.

Limitations and broader perspective. Fundamentally, our choice of leveraging hierarchy and locality biases is a *deductive* approach motivated by structural properties observed in data across modalities. A complementary, more *inductive* approach is learning which biases generalize across tasks from a distribution over modalities and tasks (Baxter, 2000); this is a promising future work direction, as well as an active area of research, particularly on symmetry-related priors (Romero & Lohit, 2022; Van der Linden et al., 2024; Urbano et al., 2026). Generally, characterizing the full set of beneficial inductive biases across modalities, and understanding when learned biases should replace hand-designed ones, remains an open question. Another natural direction comes from defining a multi-hierarchy conditioning method that exploits information at all encoder levels rather than the last one, akin in spirit to multi-resolution hash encodings (Müller et al., 2022) or hierarchical latent diffusion (Kim et al., 2023). Our non-Euclidean evaluation setting is limited to the sphere: Appendix B.5 extends the ordering principle to compact Riemannian domains, but we do not evaluate this general construction in practice. A systematic study on datasets with very irregular sampling (e.g. LiDAR), and scaling to substantially higher-resolution signals, remains future work.

AI USE STATEMENT

In this work, we used generative AI tools for designing or providing feedback on experiments, implement methods, assist with translation, cleaning and reformating datasets, drafting parts of the research paper, identifying relevant literature, searching for information, assist in the writing of proofs and improving readability. We have not used generative AI tools for generating synthetic datasets, helping develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, supporting qualitative and thematic data analysis, and interpreting results. We have reviewed all AI-assisted work. We have manually checked AI-generated code and reviewed AI-recommended literature, as well as checked numbers updated in figures through AI agents. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

As a representation learning method that can be paired with generative models on images, it shares the standard misuse risks of image generation systems, e.g. the synthesis of unauthorized or misleading visual content. Positive impacts of modality-agnostic representation learning could include its use in scientific applications. Besides this, we do not anticipate broader societal impacts beyond those typical for foundational representation learning research.

REPRODUCIBILITY STATEMENT

Our code is publicly available at `link-hidden-for-double-blind-review`, from which the main results of the paper (Table 2 and 3) can be fully reproduced.

REFERENCES

Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5835–5844, 2021. doi: 10.1109/ICCV48922.2021.00580.

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

Jonathan Baxter. A model of inductive bias learning. *Journal of artificial intelligence research*, 12: 149–198, 2000.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

Carsten Burstedde, Johannes Holke, and Tobin Isaac. Bounds on the number of discontinuities of morton-type space-filling curves. *arXiv preprint arXiv:1505.05055*, 2015.

Joao Carreira, Skanda Koppula, Daniel Zoran, Adria Recasens, Catalin Ionescu, Olivier Henaff, Evan Shelhamer, Relja Arandjelovic, Matt Botvinick, Oriol Vinyals, Karen Simonyan, Andrew Zisserman, and Andrew Jaegle. Hip: Hierarchical perceiver, 2022. URL `https://arxiv.org/abs/2202.10890`.

Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European conference on computer vision*, pp. 333–350. Springer, 2022.

Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8628–8638, June 2021.

Yinbo Chen, Oliver Wang, Richard Zhang, Eli Shechtman, Xiaolong Wang, and Michael Gharbi. Image neural field diffusion models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8007–8017. IEEE, 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Yilun Du, Katie Collins, Josh Tenenbaum, and Vincent Sitzmann. Learning signal-agnostic manifolds of neural fields. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 8320–8331. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/4639475d6782a08c1e964f9a4329a254-Paper.pdf.

Shivam Duggal, Phillip Isola, Antonio Torralba, and William T. Freeman. Adaptive length image tokenization via recurrent allocation. In *First Workshop on Scalable Optimization for Efficient and Adaptive Foundation Models*, 2025. URL https://openreview.net/forum?id=AgevIIF7LN.

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 Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 5694–5725. PMLR, 17–23 Jul 2022a. URL https://proceedings.mlr.press/v162/dupont22a.html.

Emilien Dupont, Yee Whye Teh, and Arnaud Doucet. Generative models as distributions of functions. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera (eds.), *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 2989–3015. PMLR, 28–30 Mar 2022b. URL https://proceedings.mlr.press/v151/dupont22a.html.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Paul Friedrich, Florentin Bieder, Julian McGinnis, Julia Wolleb, Daniel Rueckert, and Philippe C. Cattin. Medfuncta: A unified framework for learning efficient medical neural fields. In *Medical Imaging with Deep Learning*, 2026. URL https://openreview.net/forum?id=URCOfkQOwP.

Maximilien Gadouleau and Tobias Weinzierl. The maximum discrete surface-to-volume ratio of space-filling curve partitions. *arXiv preprint arXiv:2106.12856*, 2021.

Alexander Gielisse and Jan van Gemert. End-to-end implicit neural representations for classification. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18728–18737. IEEE, 2025.

Google. S2 Geometry Library. https://s2geometry.io, 2024. Hierarchical decomposition of the sphere via Hilbert curves on the six faces of an enclosing cube.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 50678–50702, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/dcad3425f5c8c36b5b3885c091bf1257-Paper-Conference.pdf.

Sukjun Hwang, Brandon Wang, and Albert Gu. Dynamic chunking for end-to-end hierarchical sequence modeling. In *International Conference on Learning Representations*, volume 2026, pp. 149273–149313, 2026.

Tuomas Hytönen and Anna Kairema. Systems of dyadic cubes in a doubling metric space. *arXiv preprint arXiv:1012.1985*, 2010.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J Henaff, Matthew Botvinick, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. Perceiver IO: A general architecture for structured inputs & outputs. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=fILj7WpI-g.

Minju Jo, Woojin Cho, Uvini Balasuriya Mudiyanselage, Seungjun Lee, Noseong Park, and Kookjin Lee. Pdefuncta: Spectrally-aware neural representation for pde solution modeling. *Advances in Neural Information Processing Systems*, 38:2005–2034, 2026.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Hk99zCeAb.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Seung Wook Kim, Bradley Brown, Kangxue Yin, Karsten Kreis, Katja Schwarz, Daiqing Li, Robin Rombach, Antonio Torralba, and Sanja Fidler. Neuralfield-ldm: Scene generation with hierarchical latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8496–8506, June 2023.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

David M Knigge, David R Wessels, Riccardo Valperga, Samuele Papa, Jan-Jakob Sonke, Efstratios Gavves, and Erik J Bekkers. Space-time continuous pde forecasting using equivariant neural fields. *Advances in neural information processing systems*, 37:76553–76577, 2024.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Jaewon Lee and Kyong Hwan Jin. Local texture estimator for implicit representation function. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1919–1928, 2022. doi: 10.1109/CVPR52688.2022.00197.

Adam Liddell. s2cell: Minimal python s2 cell id, s2point and lat/lon conversion library. https://pypi.org/project/s2cell/, 2024.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9992–10002, 2021. doi: 10.1109/ICCV48922.2021.00986.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11966–11976. IEEE, 2022.

Stephane G Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE transactions on pattern analysis and machine intelligence*, 11(7):674–693, 1989.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4455–4465. IEEE, 2019.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4): 1–15, 2022.

Aviv Navon, Aviv Shamsian, Idan Achituve, Ethan Fetaya, Gal Chechik, and Haggai Maron. Equivariant architectures for learning in deep weight spaces. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 25790–25816. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/navon23a.html.

Samuele Papa. Spatial functa — unofficial JAX implementation. https://github.com/samuelepapa/spatial_functa, 2024. Unofficial reimplementation.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 165–174. IEEE, 2019.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *European conference on computer vision*, pp. 523–540. Springer, 2020.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-learning with implicit gradients. *Advances in neural information processing systems*, 32, 2019.

Pierluigi Zama Ramirez, Luca De Luigi, Daniele Sirocchi, Adriano Cardace, Riccardo Spezialetti, Francesco Ballerini, Samuele Salti, and Luigi Di Stefano. Deep learning on object-centric 3d neural fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9940–9956, 2024.

David W Romero and Suhas Lohit. Learning partial equivariances from data. *Advances in Neural Information Processing Systems*, 35:36466–36478, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2015. URL https://arxiv.org/abs/1409.0575.

Michael Ryoo, AJ Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. Tokenlearner: Adaptive space-time tokenization for videos. *Advances in neural information processing systems*, 34:12786–12797, 2021.

Louis Serrano, Thomas X Wang, Etienne Le Naour, Jean-Noël Vittaut, and Patrick Gallinari. Aroma: Preserving spatial structure for latent pde modeling with local neural fields. *Advances in Neural Information Processing Systems*, 37:13489–13521, 2024.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020.

Alonso Urbano, David W. Romero, Max Zimmer, and Sebastian Pokutta. RECON: Robust symmetry discovery via explicit canonical orientation normalization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=bpWzTPDybh.

Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. *Advances in neural information processing systems*, 33:19667–19679, 2020.

Putri A Van der Linden, Alejandro García-Castellanos, Sharvaree Vadgama, Thijs P Kuipers, and Erik J Bekkers. Learning symmetries via weight-sharing with doubly stochastic tensors. *Advances in Neural Information Processing Systems*, 37:42738–42764, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Peng-Shuai Wang. Octformer: Octree-based transformers for 3d point clouds. *ACM Transactions on Graphics (TOG)*, 42(4):1–11, 2023.

David Wessels, David Knigge, Riccardo Valperga, Samuele Papa, Sharvaree Vadgama, Efstratios Gavves, and Erik Bekkers. Grounding continuous representations in geometry: Equivariant neural fields. In *International Conference on Learning Representations*, volume 2025, pp. 59774–59794, 2025.

Julia Wolleb, Florentin Bieder, Paul Friedrich, Hemant D Tagare, and Xenophon Papademetris. Vidfuncta: Towards generalizable neural representations for ultrasound videos. In *International Workshop on Advances in Simplifying Medical Ultrasound*, pp. 109–119. Springer, 2025.

Julia Wolleb, Cristiana Baloescu, Alicia Durrer, Hemant D Tagare, and Xenophon Papademetris. Low-rank-modulated functa: Exploring the latent space of implicit neural representations for interpretable ultrasound video analysis. *arXiv preprint arXiv:2603.25951*, 2026.

Wilson Yan, Volodymyr Mnih, Aleksandra Faust, Matei Zaharia, Pieter Abbeel, and Hao Liu. Elastictok: Adaptive tokenization for image and video. In *International Conference on Learning Representations*, volume 2025, pp. 38036–38056, 2025.

Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016.

Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4576–4585, 2021. doi: 10.1109/CVPR46437.2021.00455.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Biao Zhang, Matthias Nießner, and Peter Wonka. 3dilg: Irregular latent grids for 3d generative modeling. *Advances in Neural Information Processing Systems*, 35:21871–21885, 2022.

Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions On Graphics (TOG)*, 42(4):1–16, 2023.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1Ddp1-Rb.

Zhengqiang Zhang, Rongyuan Wu, Lingchen Sun, and Lei Zhang. Gpstoken: Gaussian parameterized spatially-adaptive tokenization for image representation and generation. *Advances in Neural Information Processing Systems*, 38:11218–11249, 2026.

Peiye Zhuang, Samira Abnar, Jiatao Gu, Alex Schwing, Joshua M. Susskind, and Miguel Ángel Bautista. Diffusion probabilistic fields. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=ik91mY-2GN.

A Coordinate Ordering Details

A.1 Euclidean locality-preserving orderings

Morton / Z-order key Let $x \in [-1, 1]^d$. Define the per-dimension integer quantization $q_k(x) = \left\lfloor \frac{x^{(k)}+1}{2} (2^b - 1) + \frac{1}{2} \right\rfloor \in \{0, \ldots, 2^b - 1\}$, i.e. rounded to the nearest integer, where b is the number of quantization bits. Let $\mathrm{bit}_j(q_k(x)) \in \{0, 1\}$ denote the j-th least significant bit of $q_k(x)$. The Morton key is obtained by interleaving bits across dimensions:

$$\kappa(x) = \sum_{j=0}^{b-1} \sum_{k=1}^{d} \mathrm{bit}_j(q_k(x))\, 2^{dj+(k-1)}. \tag{10}$$

Sorting tokens by κ produces a space-filling curve that visits points in a spatially coherent order. Since the quantization uses $2^b - 1$ rather than 2^b, the regions defined by the leading bits of the key are wider than the midpoint halvings of Appendix B by a factor $2^b/(2^b - 1)$ (1.004 for $b = 8$), and only the first b levels of halving are resolved; the upper diameter bounds of Appendix B (e.g. Proposition B.3) hold for this key with this factor, at those levels. For d=2 this is the standard Z-curve; for d=3 it generalizes to a 3D Z-curve. On regular grids, Morton ordering produces roughly square (2D) or cubic (3D) groups that respect spatial locality at every scale.

k-d tree linearization For sparse or irregularly sampled data, Morton ordering can produce groups that span empty regions of the coordinate space (see Section B.3). An alternative ordering is the k-d tree linearization:

1. Given a set of N points with coordinates in $\mathbb{R}^d$, compute the bounding box and identify the axis of maximum spread.
2. Sort the points along that axis and split at the median.
3. Recurse on each half until each subset contains a single point.
4. Concatenate the leaves in depth-first, left-before-right order.

The resulting permutation places spatially nearby points at adjacent positions in the sequence.

A.2 Non-Euclidean locality-preserving orderings

For data living on a non-Euclidean manifold, the same locality-preserving idea applies. The construction is the same as in the Euclidean case (Section A.1): traverse a hierarchical partition of the manifold depth-first (Appendix B.5), then use the resulting permutation as the input ordering for the contiguous-chunk grouping in the LH-NeF tokenizer.

Spherical data (ERA5). For ERA5, points live on the unit sphere S^2 embedded as $(x, y, z) = (\cos\theta\cos\varphi,\ \cos\theta\sin\varphi,\ \sin\theta)$, which naturally handles periodicity at the antimeridian and convergence at the poles. We sort points using the S2 cell hierarchy (Google, 2024): each point is mapped to its S2 cell token at a level chosen so that the total number of cells is at least N (so that most cells contain at most one point), and the token strings are sorted lexicographically. Because S2 cell IDs are constructed from a face index plus a Hilbert-curve position within that face, a lexicographic sort over tokens directly yields a Hilbert-curve traversal of the sphere. We implement it using the `s2cell` Python package (Liddell, 2024).

B Locality guarantees

In this section, we measure and bound the locality of the groups produced by a given ordering under certain conditions, and explain how this locality affects rendering. In general, locality depends on how the observations fill the domain, and different guarantees can be obtained under different scenarios.

When observations form a regular grid with 2^b points per axis, Morton and k-d tree groups are boxes whose diameter we can compute exactly: it shrinks as $1/\sqrt[d]{G}$ with the number of groups, and no

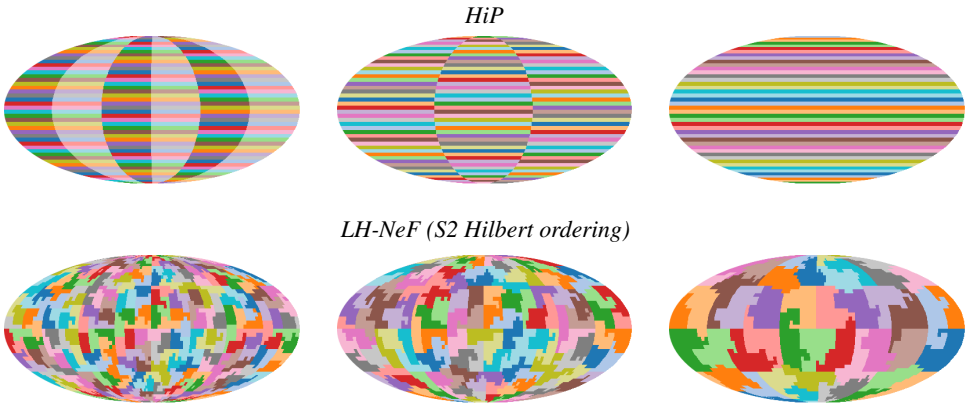


Figure 4: Receptive fields of the tokenizer hierarchy for the ERA5 (46×90 lat-lon grid on S^2) LH-NeF tokenizer configuration. *HiP* (top): groups follow flat lat-lon raster order, producing latitude bands that lump unrelated climate regions together. *LH-NeF* (bottom): the S2 cell ordering produces compact, geographically coherent regions at every level. Each color represents one group.

other ordering can make a group smaller by more than a factor $\sqrt{d}$, whereas raster ordering is never better (Section B.2). For arbitrary observations, no ordering can promise that every group has small diameter (i.e. that respects locality), since one can easily construct observation sets that force at least one large group (Section B.1). What Morton ordering does guarantee, for any observation set, is that large groups are *rare*: fewer than a fraction 2^{-d} of them exceed the well-behaved diameter upper bound of the grid case (Section B.3). The k-d tree offers no such closed guarantee without further assumptions, because its groups adapt to the sampling density, but empirically it yields the most compact groups. Finally, Section B.4 shows how these diameters govern which tokens the renderer reads, which in turn explains the mechanism by which the locality bias exercises most of the performance gains in our experiments.

Measuring locality The spatial support of a group $g \in \{0, \dots, G-1\}$ is the set of coordinates $\mathcal{X}_g = \{x_i : i \in \mathcal{I}_g\}$. We measure locality as the *diameter* of $\mathcal{X}_g$:

$$\operatorname{diam}(\mathcal{X}_g) := \max_{i,j \in \mathcal{I}_g} d_{\mathcal{X}}(x_i, x_j).$$

When $\operatorname{diam}(\mathcal{X}_g)$ is small, the group's tokens summarize a small neighborhood of the domain; when it is large, observations that are far apart in $\mathcal{X}$ are forced to share tokens. We compare $\operatorname{diam}(\mathcal{X}_g)$ with the largest distance attainable in the domain, $\operatorname{diam}(\mathcal{X})$. The number of groups G is fixed by the architecture, so a group is always "$m = N/G$ consecutive positions of the sorted list", whatever the ordering. Therefore, the ordering (raster, Morton, ...) fully determines $\mathcal{X}_g$ and hence its diameter.

B.1 Guaranteeing locality needs assumptions about how observations fill the coordinate space

We start by stating one fact: without any assumption on how observations are distributed on $\mathcal{X}$, well-behaved locality or compact groups (i.e. small diameters) cannot be guaranteed in general. In effect, one can construct examples in which there is always a group that contains coordinates that are far apart. Consider a set of any number of observations –say, a sequence of $N-1$ observations with coordinates x_i for $i \in \{0, \dots, N-2\}$– that fill a section of the space e.g. uniformly (without loss of generality) in a blob. Then, consider an extra observation with coordinate x_{N-1} that stands at a large distance from this blob (Figure 5 illustrates this example). In this case, regardless of how we sort the sequence of observations, when we partition the N observations into G groups of $m \geq 2$ observations, there exists a group $\mathcal{X}_g$ that contains both coordinate x_{N-1} and some other x_i, $i \in \{0, \dots, N-2\}$. By construction this group has a large diameter, i.e. it "violates locality".

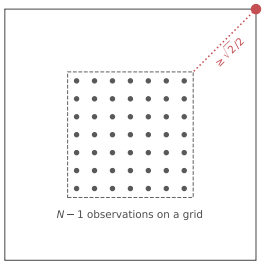


Figure 5: One observation at a corner of $\mathcal{X} = [-1, 1]^2$, all others in the central square (dashed). For any possible ordering, the corner observation shares a group with some other observation, the nearest of which is $\sqrt{2}/2 = \mathrm{diam}(\mathcal{X})/4$ away, independently of N and G.

Therefore, to offer conditions in which we can bound or guarantee small diameters for all groups, *we need assumptions about how observations fill the space* – in other words, for arbitrary observation sets, no ordering can guarantee a small diameter for every group.

Section B.2 therefore starts from the most regular case, grids with 2^b points per axis, where the diameter of every group is known in closed form. Section B.3 drops all assumptions on the observations and shows what remains of the guarantee.

B.2 Regular grids

B.2.1 Morton ordering

Morton ordering splits a region in half along every axis *at its midpoint*, giving 2^d equal sub-regions listed in a fixed order (e.g. in 2D, $d = 2$: bottom-left, bottom-right, top-left, top-right; an observation on a dividing line belongs to the lower half). We work recursively: start with the whole domain as one region containing all N observations. Split it into sub-regions by halving all axes and list the observations sub-region by sub-region, in a fixed order of sub-regions; inside each sub-region, list its observations by the same rule; stop when a region contains at most one observation. Listing every region completely before moving to the next one and up the hierarchy, we obtain the Morton-ordered sequence.

Let the observations form a regular grid on $\mathcal{X} = [-1, 1]^d$ with $n = 2^b$ points per axis at the cell centres $-1 + (j + \frac{1}{2})\Delta$, $j = 0, \ldots, n - 1$, with spacing $\Delta = 2/n$, and $N = n^d = 2^{db}$ observations in total. The diameter of the domain is the distance between two opposite corners,

$$\mathrm{diam}(\mathcal{X}) = d_{\mathcal{X}}\big((-1, \ldots, -1), (1, \ldots, 1)\big) = 2\sqrt{d}.$$

Let G divide N, so that the group size $m = N/G$ is a power of two, $m = 2^t$ with $t < db$, and write $t = dj + r$ with $j \in \mathbb{N}$ and $0 \le r < d$.

Proposition B.1 (Morton groups on a 2^b grid)**.** *In this setting, every Morton group is a box of grid points with 2^{j+1} points along r of the axes and 2^j along the remaining $d - r$, and*

$$\mathrm{diam}(\mathcal{X}_g) = \Delta\sqrt{r\,(2^{j+1} - 1)^2 + (d - r)\,(2^j - 1)^2} \;<\; \frac{2}{\sqrt[d]{G}}\,\mathrm{diam}(\mathcal{X}). \tag{11}$$

Proof. Every Morton midpoint falls between grid points, so each halving splits every axis of a region into two halves with the same number of grid points. After k halvings a region is therefore a box with $2^b/2^k = 2^{b-k}$ points per axis, i.e. $(2^{b-k})^d = 2^{d(b-k)}$ observations, and there are 2^{dk} such regions.

Morton ordering lists each region completely before starting the next. Number the regions at level k as $q = 0, 1, 2, \ldots$ in the order in which they are visited. Since all of them contain the same number of observations, region 0 occupies positions $0, \ldots, 2^{d(b-k)} - 1$ of the sorted list, region 1 the positions $2^{d(b-k)}, \ldots, 2 \cdot 2^{d(b-k)} - 1$, and in general

$$\text{region } q \text{ occupies positions } q\,2^{d(b-k)},\ \ldots,\ (q{+}1)\,2^{d(b-k)} - 1.$$

We look for a level k^* at which every region contains exactly $m = 2^t$ observations:

$$2^{d(b-k^*)} = 2^t \iff t = d(b - k^*) \iff k^* = b - \frac{t}{d}.$$

Since k^* counts halvings, it must be a natural number, which holds if and only if t is a multiple of d, i.e. $r = 0$.

Case 1: $r = 0$, i.e. $t = dj$ and $k^* = b - j$. At level k^* every region has m observations, so region q occupies positions $qm, \dots, (q+1)m - 1$. The tokenizer cuts the sorted list in exactly the same way: group g occupies positions $gm, \dots, (g+1)m - 1$. Hence $\mathcal{X}_g$ is region g at level k^*, a box with $2^{b-k^*} = 2^j$ grid points per axis. Let $(a_1, \dots, a_d)$ be its corner with the smallest coordinates; its points are $(a_1 + i_1\Delta, \dots, a_d + i_d\Delta)$ with $i_1, \dots, i_d \in \{0, \dots, 2^j - 1\}$. Two such points differ by at most $\Delta(2^j - 1)$ along every axis, and the opposite corner $(a_1 + \Delta(2^j - 1), \dots, a_d + \Delta(2^j - 1))$ attains this along all axes at once, so

$$\operatorname{diam}(\mathcal{X}_g) = \sqrt{\Delta^2(2^j - 1)^2 + \cdots + \Delta^2(2^j - 1)^2} = \sqrt{d}\,\Delta\,(2^j - 1).$$

Case 2: $0 < r < d$. There is no level whose regions contain exactly m observations, so we use the level $k^* = b - j$ of Case 1, at which every region contains 2^{dj} observations, fewer than a group: $m = 2^{dj+r} = 2^r \cdot 2^{dj}$. Group boundaries (multiples of m) are therefore also region boundaries (multiples of 2^{dj}), and every group consists of exactly 2^r consecutive regions of level k^*. Going one level up in the hierarchy ("de-halving"), the regions of level $k^* - 1 = b - j - 1$ contain $2^{d(j+1)} = 2^d \cdot 2^{dj}$ observations, a multiple of m; so each group lies inside a single region of level $k^* - 1$, its *parent*, and consists of 2^r of the parent's 2^d sub-regions. If a group contained all 2^d sub-regions it would be the parent itself, a full de-halving with 2^{j+1} points along every axis. Since $r < d$, it contains only 2^r of them, and these undo the last halving along r axes only: the sub-regions of a region are visited in a fixed order in which consecutive pairs are neighbours along axis 1, consecutive quadruples fill axes 1 and 2, and so on, so 2^r consecutive sub-regions are joined along the first r axes. The group is therefore a box with 2^{j+1} points along the first r axes and 2^j along the remaining $d - r$. Its opposite corners are $(a_1, \dots, a_d)$ and

$$\big(a_1 + \Delta(2^{j+1} - 1), \dots, a_r + \Delta(2^{j+1} - 1),\ a_{r+1} + \Delta(2^j - 1), \dots, a_d + \Delta(2^j - 1)\big),$$

and as in Case 1,

$$\begin{aligned}\operatorname{diam}(\mathcal{X}_g) &= \sqrt{r\,\Delta^2(2^{j+1} - 1)^2 + (d - r)\,\Delta^2(2^j - 1)^2} \\ &= \Delta\sqrt{r\,(2^{j+1} - 1)^2 + (d - r)\,(2^j - 1)^2}.\end{aligned}$$

For $r = 0$ this is the formula of Case 1, so it holds for every group size.

Finally, we want to relate this closed formula to the diameter of the whole coordinate space. Replacing both $2^{j+1} - 1$ and $2^j - 1$ by the larger 2^{j+1},

$$\begin{aligned}\operatorname{diam}(\mathcal{X}_g) &< \Delta\sqrt{r\,(2^{j+1})^2 + (d - r)\,(2^{j+1})^2} = \Delta\,2^{j+1}\sqrt{d} \\ &= \frac{2}{n}\,2^{j+1}\sqrt{d} = \operatorname{diam}(\mathcal{X})\,\frac{2^{j+1}}{2^b} = 2^{\,j+1-b}\operatorname{diam}(\mathcal{X}),\end{aligned}$$

using $\operatorname{diam}(\mathcal{X}) = 2\sqrt{d}$ and $n = 2^b$. To express this through the number of groups, note $1/G = m/N = 2^{dj+r}/2^{db} = 2^{d(j-b)+r}$, i.e. $1/\sqrt[d]{G} = 2^{\,j-b+r/d}$. Splitting the exponent as $j - b + 1 = (j - b + r/d) + (1 - r/d)$,

$$\begin{aligned}2^{\,j+1-b}\operatorname{diam}(\mathcal{X}) &= 2^{\,j-b+r/d} \cdot 2^{\,1-r/d}\operatorname{diam}(\mathcal{X}) \\ &= \frac{2^{\,1-r/d}}{\sqrt[d]{G}}\operatorname{diam}(\mathcal{X}) \le \frac{2}{\sqrt[d]{G}}\operatorname{diam}(\mathcal{X}),\end{aligned}$$

since $0 \le r < d$ gives $2^{1-r/d} \le 2$. □

Equation (11) bounds the diameter of *every* group by $2\operatorname{diam}(\mathcal{X})/\sqrt[d]{G}$: multiplying the number of groups by 2^d halves every group along every axis, so groups shrink in all directions at once and never turn into strips. On the 64^2 grid, the three tokenizer blocks ($G = 256, 128, 64$) give $4{\times}4$, $8{\times}4$ and $8{\times}8$ pixel blocks with diameters 0.133, 0.238 and 0.309.

The proof used two assumptions, and both serve the same purpose: they make the boundaries between groups in the sorted list coincide with the boundaries between regions, which in turn enables a closed-form solution for the group diameters.

- $n = 2^b$ *points per axis.* Every midpoint then falls between grid points, so all regions at a level contain the same number of observations and region boundaries fall at the multiples of the region size. With $n = 5$, for instance, the first midpoint falls on the middle row and column, and the four quadrants contain 9, 6, 6 and 4 observations.
- G *divides* N. Since $N = 2^{db}$, this means G is a power of two, so all groups contain the same number $m = 2^t$ of observations and group boundaries fall at the multiples of m, which are also multiples of the region sizes used in the proof. With $G = 3$ on a 4×4 grid, for instance, the groups contain 5, 5 and 6 observations, and their boundaries at positions 5 and 10 cut through the quadrants, which occupy positions 0–3, 4–7, ...

If either fails, some group boundaries fall inside regions, and a group can contain observations from two regions that are not adjacent in the domain. This is what rules out a closed form for all groups off such grids. However, it does not rule out locality: Section B.3 shows that, for every observation set, only a bounded fraction of the Morton groups can be large (Proposition B.3).

B.2.2 K-D TREE ORDERING EQUIVALENCE WITH MORTON ORDERING

k-d tree ordering follows the same recursive principle, but splits a region in two *at the median of its observations* along the axis of largest spread (ties broken towards the lowest axis index), so each split gives exactly half of the observations to each side. We split until there are G regions, the leaves, and take them as the groups; listing the leaves in the order of the recursion gives the k-d-ordered sequence. The difference with Morton ordering is which quantity is fixed: Morton fixes the regions (always the halves of the domain) and lets the number of observations per region follow the data; the k-d tree fixes the number of observations per region and lets the regions follow the data.

On the grid of Proposition B.1, every region has an even number of grid points along each axis, so the median falls between the two middle grid lines, i.e. at the midpoint: each k-d split is a Morton halving along a single axis. Since all axes start with the same spread, the splits cycle through axes $1, \ldots, d$, so every d consecutive k-d splits amount to one Morton halving. With $G = 2^\ell$ groups and $m = N/G = 2^{dj+r}$, the tree performs $\ell = d(b-j) - r$ splits. For $r = 0$ this is a whole number of cycles, and the leaves are exactly the Morton groups. For $0 < r < d$ the tree stops $d - r$ splits into a cycle, which leaves a box with 2^j points along $d - r$ axes and 2^{j+1} along the other r: the Morton group of Case 2 up to a permutation of the axes. Equation (11) therefore holds for k-d tree groups as well.

On regular 2^b grids the two orderings are thus equivalent: they produce boxes of the same size and diameter, differing at most in which r axes are the long ones (the first r for Morton, the last r for the k-d tree). This is consistent with Table 4, where Morton and k-d tree ordering are within 0.8 dB/IoU of each other.

B.2.3 RASTER ORDERING

Raster (row-major) ordering, as used by HiP, lists the observations along axis 1 through the whole domain before advancing one step along axis 2, and so on. It is not a region-by-region ordering, and its groups behave in the opposite way.

Call a *row* the n grid points that differ only in their first coordinate. Raster ordering lists the rows one after another, so row q occupies positions $qn, \ldots, (q{+}1)n - 1$, and group g occupies positions $gm, \ldots, (g{+}1)m - 1$ with $m = N/G$. Since m and n are powers of two, one divides the other. If $G > n^{d-1}$, i.e. $m < n$, every group consists of m consecutive points of a single row, so

$$\operatorname{diam}(\mathcal{X}_g) = \Big(\frac{N}{G} - 1\Big)\Delta.$$

If $G \le n^{d-1}$, i.e. $m \ge n$, every group consists of whole rows and contains the two end points of a row, so

$$\operatorname{diam}(\mathcal{X}_g) \;\ge\; (n-1)\Delta \;=\; 2 - \Delta,$$

the full width of the sampled domain, however many groups there are. A raster group thus grows along axis 1 until it spans the domain, and only then along the other axes, whereas a Morton group grows along all axes at once.

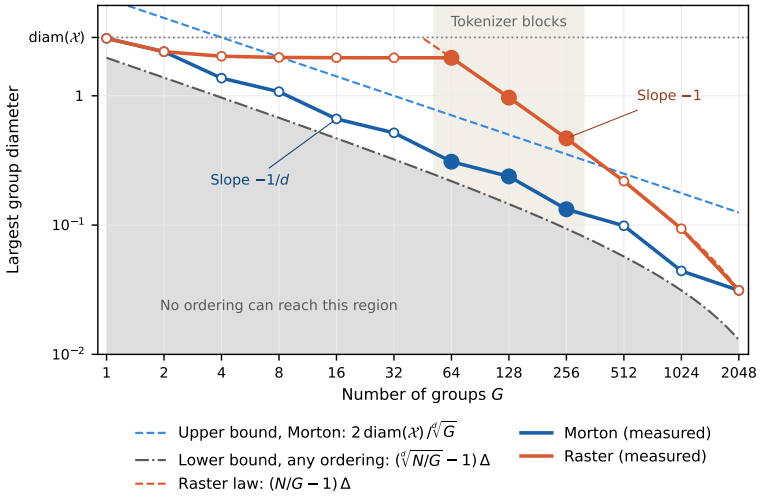


Figure 6: Largest group diameter against the number of groups G on the 64^2 grid ($d = 2$) of the CelebA-HQ setting on log-log axes (large markers and gray band represent the tokenizer block's configuration on this experiment). On log–log axes, a law $\mathrm{diam} \propto G^{-a}$ is a straight line of slope $-a$. Morton groups shrink with slope $-1/d$ from the first split on. Raster groups stay at least as wide as the image as long as each group contains full rows ($G \leq 64$), and only then shrink as strips, with slope -1. The dashed blue line is the upper bound of Proposition B.1, not the exact law, so the measured values stay below it. The small zig-zag between consecutive values of G comes from Morton groups alternating between squares and 2:1 rectangles as G doubles. The grey region lies below the lower bound of Proposition B.2 and cannot be reached by any ordering.

At the conditioning level of the 64^2 model ($G = 64$), each raster group is one full image row with diameter $63\Delta = 1.969$, against 0.309 for the 8×8 Morton or k-d tree group. For the three tokenizer blocks ($G = 256, 128, 64$), raster diameters are 0.469, 0.969 and 1.969, against 0.133, 0.238 and 0.309. The two orderings also respond differently to the number of groups: Morton and k-d tree groups shrink as $1/\sqrt[d]{G}$ from the first split on, whereas raster groups keep the width of the domain as long as $G \leq n^{d-1}$, and only then shrink, as one-dimensional strips.

B.2.4 Optimality of Morton and k-d tree ordering

The bound of Proposition B.1 is an upper bound. We now show that, on the same grid, no ordering can make its groups much smaller, so Morton and k-d tree ordering are close to the best possible.

Proposition B.2 (Lower bound for any ordering)**.** *In the setting of Proposition B.1, every group produced by any ordering satisfies*

$$\mathrm{diam}(\mathcal{X}_g) \;\geq\; \left(\sqrt[d]{\frac{N}{G}} - 1\right)\Delta.$$

Proof. Take any group of $m = N/G$ grid points and let $D = \mathrm{diam}(\mathcal{X}_g)$. We count how many grid points can fit in a set of diameter D.

First, look at a single axis. Two points of the group are at distance at most D, so their coordinates along that axis differ by at most D. Hence all the group's coordinates along that axis lie in an interval of length at most D.

Second, count grid values in that interval. Along each axis, grid points are spaced Δ apart, so an interval of length D contains at most $D/\Delta + 1$ of them. For example, an interval of length 3Δ contains at most 4 grid values.

Third, combine the axes. A grid point is determined by its d coordinates, and each coordinate can take at most $D/\Delta + 1$ values. So the group contains at most $(D/\Delta + 1)^d$ distinct grid points.

Finally, the group contains exactly $m = N/G$ distinct grid points, so

$$\frac{N}{G} \le \Big(\frac{D}{\Delta}+1\Big)^d \iff \sqrt[d]{\frac{N}{G}} \le \frac{D}{\Delta}+1 \iff D \ge \Big(\sqrt[d]{\frac{N}{G}}-1\Big)\Delta. \qquad \square$$

The bound does not depend on the ordering: it only uses that a group contains N/G distinct grid points. When $N/G = 2^{dj}$, Morton and k-d tree groups are cubes with $\sqrt[d]{N/G} = 2^j$ points per side and diameter $\sqrt{d}\,\big(\sqrt[d]{N/G}-1\big)\Delta$. They are therefore larger than the smallest possible group by at most a factor $\sqrt{d}$, the ratio between the diagonal and the side of a cube. Raster groups with $G > n^{d-1}$ have diameter $(N/G-1)\Delta$: a strip of N/G points instead of a box with $\sqrt[d]{N/G}$ points per side, larger than the lower bound by a factor of about $(N/G)/\sqrt[d]{N/G}$, i.e. $\sqrt{N/G}$ in 2D. Figure 6 shows both bounds together with the measured diameters of Morton and raster groups for every power-of-two number of groups on the 64^2 grid.

Relation to prior work. Cutting a space-filling curve into contiguous pieces has been previously studied. For continuous curves such as Hilbert's, Gadouleau & Weinzierl (2021, Thm. A.1 with $c = 0$) show that every contiguous piece P satisfies $U^{-1}\delta(P)^d \le \mathrm{vol}(P) \le U\,\delta(P)^d$, with δ the ℓ_∞-diameter and U a constant depending on d and on the curve's refinement factor. For pieces of volume $1/G$ this is the rate $1/\sqrt[d]{G}$ of Proposition B.1, for arbitrary cut points. However, their argument requires consecutive cells to be adjacent, which Morton ordering violates: a contiguous Morton segment can consist of two disconnected parts (Burstedde et al., 2015, Thm. 1). Proposition B.1 shows that the rate still holds for Morton ordering when the cuts fall at multiples of the group size, as in contiguous grouping on 2^b grids, with an explicit constant. Proposition B.2 is the ordering-independent counterpart of the upper inequality $\mathrm{vol}(P) \le U\,\delta(P)^d$ of their theorem.

B.3 Arbitrary observation sets

The closed form above requires equal numbers of observations in all regions of a level and in all groups. We first show, on a small, visually intuitive example, what happens to each ordering when this is not the case. We then prove that, for Morton ordering, the resulting large groups are always few, whatever the observations, and measure both orderings on grids of other sizes, random samples and real point clouds.

Morton ordering. Figure 7 shows Morton ordering with $G = 4$ on a 4×4 and a 5×5 grid. On the 4×4 grid every quadrant contains 4 observations and the four groups are the four quadrants. On the 5×5 grid, the first midpoint falls on the middle row and column, which go to the lower halves, so the quadrants contain 9, 6, 6 and 4 observations, whereas the groups contain 7, 6, 6 and 6. Region boundaries in the sorted list therefore fall at positions 9, 15 and 21, and group boundaries at 7, 13 and 19. Each group after the first contains the end of one quadrant and the beginning of the next. Since the ordering finishes one quadrant at its last-visited corner and starts the next at its first-visited corner, which may be far apart, such a group contains observations from distant parts of the domain: in Figure 7, group 3 spans the full width of the image. The ordering itself has not changed, and consecutive observations are still close except where the list moves from one region to the next; what is lost is the guarantee *for every group*.

What is not lost is that large-diameter groups (exceeding the bound of regular grids) are *rare*. Morton ordering lists the observations region by region at every level of the hierarchy, so only a group that contains the passage from one region to the next can be large, and there are few such passages. This holds for every observation set:

Proposition B.3 (Morton groups on any observation set)**.** *Let the observations be any finite set in $\mathcal{X} = [-1,1]^d$, and let the Morton-ordered list be cut into G consecutive groups of any sizes. For every level $k \ge 0$, at most $2^{dk}-1$ groups have diameter larger than $\mathrm{diam}(\mathcal{X})/2^k$. In particular, when $\sqrt[d]{G}$ is a power of two, fewer than a fraction 2^{-d} of the groups exceed the bound $2\,\mathrm{diam}(\mathcal{X})/\sqrt[d]{G}$ of Proposition B.1, and, more generally, fewer than a fraction 2^{-ds} exceed $2^s\,\mathrm{diam}(\mathcal{X})/\sqrt[d]{G}$, for every $s \ge 1$ with $2^s \le \sqrt[d]{G}$; the case $s = 1$ is the bound of Proposition B.1.*

Proof. Halving $\mathcal{X}$ k times along every axis gives 2^{dk} cubes of side $2/2^k$, the regions of level k. Morton ordering lists all observations of one region before those of the next, so the observations

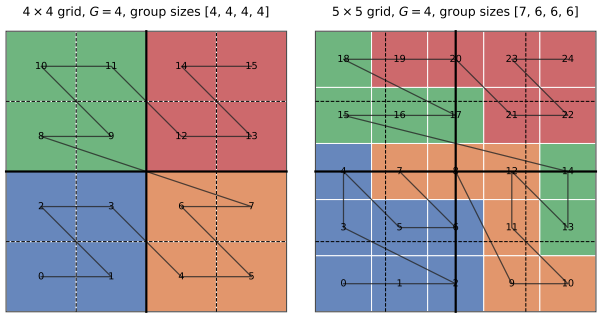


Figure 7: Morton ordering with $G = 4$ groups on a 4×4 (*left*) and a 5×5 grid (*right*). Numbers are positions in the sorted list, thick and dashed lines are the first and second halvings respectively, and colours represent different groups. Left: the groups are the four quadrants. Right: the quadrants hold $9, 6, 6, 4$ observations but the groups $7, 6, 6, 6$, so groups straddle quadrants.

of each region occupy consecutive positions of the list. Consider a group. If it does not contain a position at which the list passes from one region to the next, all its observations lie in one region, and its diameter is at most the diagonal of that region, $\sqrt{d} \cdot 2/2^k = \operatorname{diam}(\mathcal{X})/2^k$. Otherwise, it contains such a passage. Since at most 2^{dk} regions contain observations, the list passes from one region to the next at most $2^{dk} - 1$ times, and each passage lies in at most one group. Hence at most $2^{dk} - 1$ groups can have diameter larger than $\operatorname{diam}(\mathcal{X})/2^k$.

For the second statement, choose k with $2^k = \sqrt[d]{G}/2^s$. Then $\operatorname{diam}(\mathcal{X})/2^k = 2^s \cdot \operatorname{diam}(\mathcal{X})/\sqrt[d]{G}$, and the number of groups exceeding it is at most $2^{dk} - 1 < G/2^{ds}$. The case $s = 1$ is the bound of Proposition B.1. □

The proposition requires nothing of the observations: it holds for grids of any size, random samples, clusters and point clouds, and for groups of unequal size. Every exception must straddle two regions, as on the 5×5 grid, and nothing prevents them from being large, as the isolated observation of Figure 5 shows. With $G = 64$ groups, it allows at most 15 of the 64 groups to exceed the bound in 2D, and at most 7 in 3D.

k-d tree ordering. Splitting at the median gives each side half of the observations, whatever their positions. Every k-d tree group, as defined in Section B.2.2, is therefore a single box of neighbouring observations, and never joins the end of one region with the beginning of a distant one, as Morton groups do on the 5×5 grid. Figure 8 compares both orderings on the 4×4 and 5×5 grids: the largest k-d tree group has diameter 2.83 grid spacings, against 5.00 for Morton ordering. This does not bound how large a box is: in a sparsely sampled part of the domain a leaf is large, and when the data itself contains a gap, as in Figure 5, a leaf can span it. How many leaves are large therefore depends on the sampling density, and we do not establish an analogue of Proposition B.3 for the k-d tree. It is nevertheless why we use the k-d tree ordering on ShapeNet16 (Table 8).[2]

Point clouds are the extreme case for Morton ordering: regions on the object surface contain many observations, neighbouring regions in empty space contain none, so the sorted list moves between distant surface patches wherever the surface leaves one region and re-enters another. The k-d tree ordering avoids these transitions inside a group, since each of its groups is a single box.

Empirical measurements of locality in challenging scenarios Table 6 measures group diameters for $G = 64$ groups under the three orderings on different observation sets: the 64^2 grid of our CelebA-HQ configuration, grids with one point more or less per side, uniformly random samples, two separated clusters, the isolated observation of Figure 5, and real point clouds, namely the $2{,}874$

[2]Our implementation (Appendix A.1) recurses the k-d tree down to single observations and then cuts the resulting list into pieces of m. On a 2^b grid with G dividing N this gives the same groups, since all nodes at a depth contain the same number of observations. When N is not a power of two, however, the pieces need not coincide with tree nodes, and a group can then contain parts of two leaves.

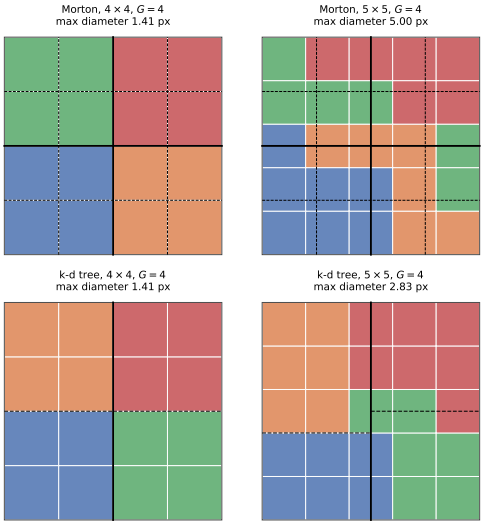


Figure 8: Morton (*top*) and k-d tree (*bottom*) ordering with $G = 4$ on a 4×4 (left) and a 5×5 grid (right), one colour per group. On the 4×4 grid both orderings produce the four 2×2 blocks (Proposition B.1 and Section B.2.2). On the 5×5 grid, Morton group boundaries fall inside regions and one group spans the full width, while every k-d tree group remains a single box.

test shapes of the ShapeNet part-segmentation benchmark (Yi et al., 2016), each with about 2,700 points sampled on the object's surface and centred and scaled into $[-1, 1]^3$. For each ordering, the table reports the largest group diameter of an observation set (averaged over the sets when there are several), the mean diameter over all groups, and the percentage of groups within the upper bound $2\,\mathrm{diam}(\mathcal{X})/\sqrt[d]{G}$ of Equation (11).

Table 6: Group diameters for $G = 64$ groups. The first six rows have $N = 4096$ observations (3969 and 4225 for the 63^2 and 65^2 grids) on $\mathcal{X} = [-1, 1]^2$ ($\mathrm{diam}(\mathcal{X}) = 2.83$, bound $2\,\mathrm{diam}(\mathcal{X})/\sqrt{G} = 0.71$); the last row has point clouds on $\mathcal{X} = [-1, 1]^3$ ($\mathrm{diam}(\mathcal{X}) = 3.46$, bound $2\,\mathrm{diam}(\mathcal{X})/\sqrt[3]{G} = 1.73$). Max: largest group diameter of an observation set, averaged over the sets. Mean: over all groups. % in bound: percentage of all groups within the bound of Equation (11).

	Morton			k-d tree			raster		
observations	max	mean	% in bound	max	mean	% in bound	max	mean	% in bound
64×64 grid	0.309	0.309	100	0.309	0.309	100	1.969	1.969	0
63×63 grid	1.985	0.579	89	0.359	0.331	100	1.969	1.968	0
65×65 grid	1.989	0.569	91	0.348	0.333	100	1.969	1.969	0
uniform random (20 draws)	1.956	0.554	89	0.369	0.314	100	1.994	1.939	0
two clusters (20 draws)	0.992	0.280	95	0.796	0.224	98	1.235	0.713	53
grid in central square + one corner point	0.873	0.166	98	0.873	0.166	98	1.576	0.994	0
ShapeNet point clouds ($d = 3$, 2874 shapes)	1.851	0.526	98	0.691	0.337	>99.9	1.911	1.434	63

Four observations follow. First, off the 2^b grid, Morton ordering loses the guarantee only for a small number of groups, as Proposition B.3 predicts: on the 63^2 and 65^2 grids and on uniformly random samples, about 90% of the groups still satisfy the bound (the proposition guarantees at least 49 of 64, i.e. 76.6%) and the mean diameter stays within a factor of two of the grid value, but the largest group reaches about 70% of the domain diameter; it is the single group containing the transition between the first two quadrants. Locality is therefore violated only slightly, in the sense that the violation concerns few groups, not in the sense that those groups are only slightly too large. Second, the k-d tree keeps every group within the bound in all regularly or uniformly sampled cases. Third, when the data itself contains gaps (two separated clusters, or an isolated observation as in Figure 5), no ordering keeps every group compact, but Morton and k-d tree ordering confine the violation to one or a few groups, whereas raster groups remain as wide as the sampled region. Fourth, on point clouds, where observations lie on a surface and neighbouring regions of space are empty, the same pattern holds at a larger scale: the k-d tree gives the most compact groups (mean 0.34, largest 0.69 per shape

on average), Morton groups are larger on average (0.53) and each shape has a group spanning about half of the domain diameter (1.85), and raster groups are wide throughout (mean 1.43).

B.4 Relation to routing

The renderer conditions a query x on the tokens of the k groups whose centroids are nearest to x (Equation (2)). For this to retrieve the tokens that describe the surroundings of x, the distance from x to a centroid must be a good proxy for the distance from x to the observations of that group. Let $\delta_g(x) := \min_{i \in \mathcal{I}_g} d_{\mathcal{X}}(x, x_i)$ be the distance from x to the nearest observation of group g. On Euclidean domains the centroid μ_g lies in the convex hull of $\mathcal{X}_g$, so $d_{\mathcal{X}}(x_i, \mu_g) \le \operatorname{diam}(\mathcal{X}_g)$ for every $i \in \mathcal{I}_g$. By the triangle inequality, $|d_{\mathcal{X}}(x, x_i) - d_{\mathcal{X}}(x, \mu_g)| \le d_{\mathcal{X}}(x_i, \mu_g)$, and taking the nearest observation,

$$\big|\, d_{\mathcal{X}}(x, \mu_g) - \delta_g(x) \,\big| \;\le\; \operatorname{diam}(\mathcal{X}_g). \tag{12}$$

Consequently, for two groups g and h,

$$\delta_h(x) - \delta_g(x) > \operatorname{diam}(\mathcal{X}_g) + \operatorname{diam}(\mathcal{X}_h) \implies d_{\mathcal{X}}(x, \mu_g) < d_{\mathcal{X}}(x, \mu_h) : \tag{13}$$

whenever one group holds observations closer to the query than another by more than the two diameters, routing ranks them correctly. Both statements hold for any ordering and any observation set; the ordering determines how large the diameters in them are.

For Morton and k-d tree groups on a 2^b grid, Equation (11) bounds every diameter by $2\operatorname{diam}(\mathcal{X})/\sqrt[d]{G}$ (and the centroid is the center of the box, so the tolerance in Equation (12) is in fact $\operatorname{diam}(\mathcal{X}_g)/2$): centroid routing resolves the surroundings of every query at the scale of one group. For raster groups every diameter is of the order of the domain (Section B.2.3), so Equations (12) and (13) carry no information, and in addition all row centroids share the same coordinate along the row, $\mu_g = (c, y_g)$. For a query $x = (x_1, x_2)$,

$$d_{\mathcal{X}}(x, \mu_g)^2 - d_{\mathcal{X}}(x, \mu_h)^2 = (x_2 - y_g)^2 - (x_2 - y_h)^2 \qquad \text{for all groups } g, h,$$

so neither the routed groups nor their weights depend on x_1: routing carries no information about the query's position along the row, and the k routed groups are the k rows nearest to the query.

We quantify both cases in the setting of our CelebA-HQ model: the 64^2 pixel grid, the $G = 64$ groups of the conditioning block (64 pixels each, i.e. 8×8 blocks under Morton ordering and single rows under raster ordering), and $k = 4$. Each of the 4096 pixels serves as a query x. We compute the routed set $\mathcal{N}(x)$ by Equation (2) and record the fraction of the 64 pixels nearest to x (a neighbourhood the size of one group, x included) that belong to a group in $\mathcal{N}(x)$. The computation uses only the grid and the ordering: it involves no trained weights and, since $\mathcal{N}(x)$ does not depend on σ_θ, no bandwidth. Averaged over all queries, the routed groups contain 99.95% of the 64 nearest pixels under Morton ordering and 54.6% under raster ordering, where the four routed rows form a strip four pixels high while the 64 nearest pixels span about nine rows.

Off the grid, Equations (12) and (13) still hold, but for the few groups with a large diameter (Proposition B.3 and Section B.3) they are uninformative: the centroid of such a group lies between the distant parts it contains, near neither, so queries in those parts are routed elsewhere and the observations of that group are summarized by tokens that are not retrieved from their own neighbourhood. Finally, this is a statement about which tokens are read, not about reconstruction error: the renderer also receives the exact query coordinate through its positional encoding and FiLM path, which is why the raster-ordered model still reaches 29.9 dB.

B.5 Riemannian domains

The region-by-region ordering principle is not restricted to Euclidean domains. A compact, connected, smooth Riemannian manifold without boundary, equipped with its geodesic distance, is geometrically doubling, and we can apply Theorem 2.2 of Hytönen & Kairema (2010), which provides nested partitions whose regions have intrinsic diameter at most $C\delta^k$ at level k, for constants $C > 0$ and $0 < \delta < 1$. Each level contains finitely many regions, since their centers are separated and the domain is compact.

For a finite observation set, traverse the occupied part of this hierarchy depth-first, listing all observations in a region before moving to the next. Stop refining a branch when it contains only one

distinct observed coordinate, listing repeated observations together. This terminates because the region diameters tend to zero, whereas the finitely many distinct observed coordinates have a positive minimum separation. Let q_k be the number of regions at level k. If the resulting sequence is cut into G nonempty consecutive groups, then

$$\#\{g : \mathrm{diam}(\mathcal{X}_g) > C\delta^k\} \;\le\; q_k - 1.$$

Indeed, a group contained in one region has diameter at most $C\delta^k$, so any larger group must contain a transition between regions. There are at most $q_k - 1$ such transitions, and each can occur inside at most one group. This is the counting argument of Proposition B.3, now applied to the intrinsic distance of the manifold. Intrinsic hierarchical orderings therefore admit a bound on the number of large-diameter groups without assumptions on how the observations are distributed. This establishes the existence of orderings with this guarantee on every manifold satisfying the assumptions above, even when no domain-specific hierarchy is readily available. We do not evaluate this general construction in practice: our non-Euclidean experiment, ERA5, uses the existing S2 cell hierarchy on the sphere. The bound controls the exceptional fraction by $(q_k - 1)/G$; therefore, at levels with few regions compared with the number of groups, most groups have diameter at most $C\delta^k$.

C Tokenizer Hierarchy and Renderer Geometry

C.1 Receptive field hierarchy

Figure 9 visualizes the receptive fields induced by the LH-NeF tokenizer hierarchy on a 32×32 CIFAR-10 input. Under HiP raster ordering (top row), each group covers a horizontal stripe, i.e. information from distant coordinates must be encoded in the same token group. Under Morton ordering (bottom row), groups form compact, spatially coherent patches at every hierarchical level: from 128 fine patches at block 1, through 64 medium patches at block 2, to 32 coarse regions at block 3. Note that groups at blocks 1 and 3 appear as 2:1 rectangles rather than perfect squares: this is because the Z-curve alternates which spatial dimension it doubles at each recursion level, so contiguous chunks of 2^{2m} tokens form squares while chunks of 2^{2m+1} tokens form 2:1 rectangles. Groups remain spatially compact however, and soft group routing aggregates group information smoothly regardless.

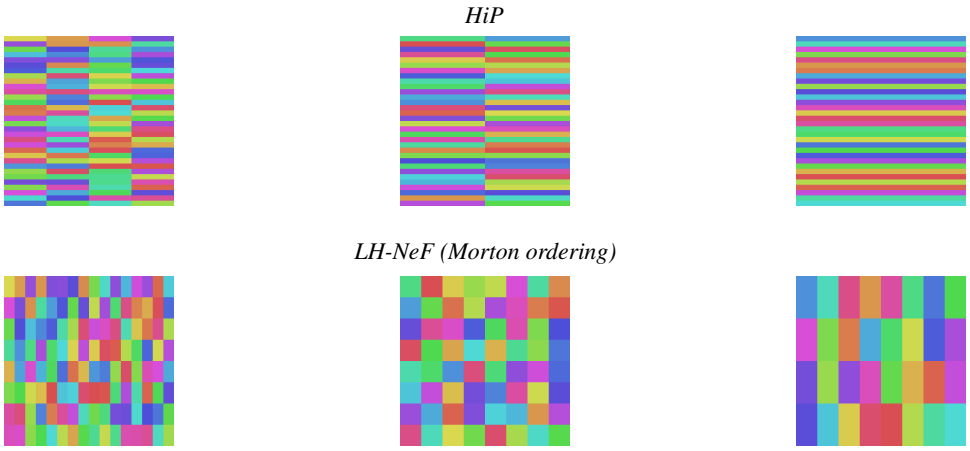


Figure 9: Receptive fields of the tokenizer hierarchy for the CIFAR-10 (32×32) LH-NeF tokenizer configuration. *HiP* (top): groups are horizontal stripes with no 2D locality. *LH-NeF* (bottom): groups are spatially compact patches that coarsen hierarchically. Each color represents one group.

Figure 10 shows the same visualization on a ShapeNet chair (32^3 voxels) using k-d tree ordering. Since grouping depends on the observed voxel distribution, groups adapt to the shape geometry, i.e. different inputs get different group distributions (and therefore routing metadata).

Figure 4 shows the same visualization for ERA5 global temperature (46×90 lat-lon grid on the unit sphere S^2) under our LH-NeF tokenizer with the S2 cell ordering described in Section A.2. Under

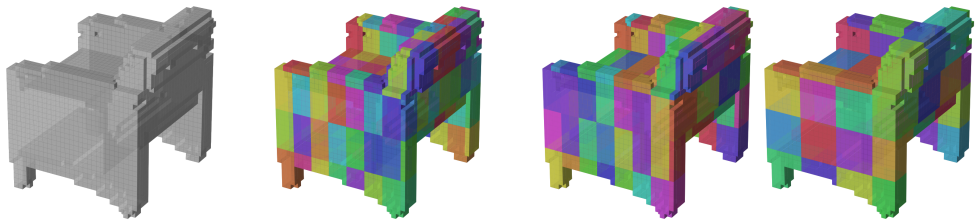


Figure 10: Receptive field LH-NeF tokenizer hierarchy on a ShapeNet chair (32^3 voxels, k-d tree ordering).

flat lat-lon raster ordering each group is a horizontal latitude band that spans the entire globe; under S2 ordering, groups become compact, geographically coherent patches at every level of the hierarchy.

Spatial partition on $\mathcal{X}$ induced by the LH-NeF tokenizer The LH-NeF renderer assigns each query coordinate to its k-nearest group centroids. Note that this mechanism induces a partition on the *entire* continuous coordinate space, not just the observed points. This matters on e.g. point clouds. We visualize these partitions on different ShapeNet point clouds, illustrating how (arbitrary) query coordinates in unoccupied, distant regions would be routed to nearby surface groups (Figure 11). Colored points show observations belonging to each highlighted group, while the semi-transparent colored cells show the full 3D Voronoi-like region that would assign queries to that group. Cells on the object surface extend into the surrounding empty space: any query in that region is routed to the nearest surface group(s). Each shape induces a different partition since group centroids depend on the distribution of observed samples.

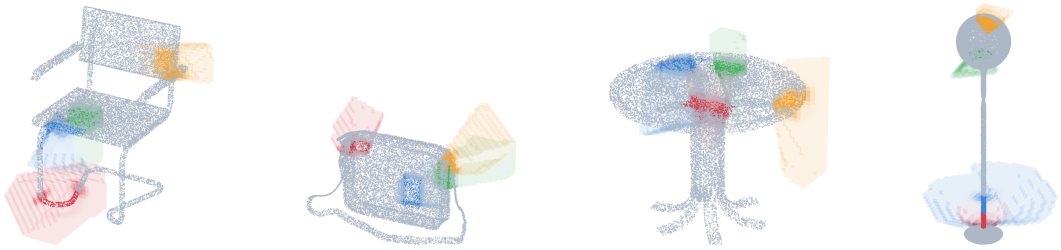


Figure 11: Voronoi partition of 3D space induced by k-d tree group centroids (G=64, four groups highlighted per shape). Colored points: observed surface samples belonging to each highlighted group. Semi-transparent colored cells: the corresponding Voronoi regions extending into unoccupied space. Groups adapt each input's point cloud geometry. From left to right: chair, bag, table, lamp.

C.2 Invariance of the FiLM coordinate frame

The FiLM modulation in the LH-NeF renderer (Section 2.2) conditions on the group-normalized relative coordinate $\tilde{x}$. On Euclidean domains, we show that $\tilde{x}$ is invariant under natural families of transformations of a fixed group, so that the FiLM output $(\gamma, \beta) = \mathrm{MLP}_{\mathrm{FiLM}}(\mathrm{PE}(\tilde{x}))$ is identical for the same local pattern at any absolute position or scale.

Consider $\mathcal{X} \subseteq \mathbb{R}^d$. The group-normalized coordinate is $\tilde{x}^g = (x - \mu_g)/\lambda_g$ with $\mu_g = \frac{1}{|\mathcal{I}_g|}\sum_i x_i$ and $\lambda_g = \max_i x_i - \min_i x_i \in \mathbb{R}^d_{>0}$ (Section 2.1). Under any transformation $T_{t,S} : x \mapsto Sx + t$ with $t \in \mathbb{R}^d$ and $S = \mathrm{diag}(s_1, \ldots, s_d)$, $s_k > 0$, applied jointly to all observations and the query,[3] the centroid transforms as $\mu'_g = S\mu_g + t$ and the extent as $\lambda'_g = S\lambda_g$ (element-wise, since S is a positive

[3]For translations and uniform scalings the transformation is a similarity and preserves the nearest-group assignment $g^* = \arg\min_g d_{\mathcal{X}}(x, \mu_g)$. For anisotropic per-axis scalings (S with non-equal diagonal entries), routing preservation is not guaranteed in general, since stretching one axis can swap which centroid is nearest.

diagonal), giving

$$\tilde{x}' = \frac{(Sx + t) - (S\mu_g + t)}{S\lambda_g} = \frac{S(x - \mu_g)}{S\lambda_g} = \frac{x - \mu_g}{\lambda_g} = \tilde{x}. \tag{14}$$

This covers translations ($S = I$), uniform scaling ($S = sI$), and anisotropic per-axis rescaling (distinct s_k). The per-axis bounding-box extent is what enables the anisotropic case: each coordinate dimension normalizes independently, so the diagonal entries of S cancel component-wise.

The invariance provides an inductive bias analogous to weight sharing in convolutional networks: the FiLM MLP learns a single mapping from relative position to modulation parameters, and the normalization by (μ_g, λ_g) ensures this mapping generalizes across groups at different absolute locations and of different spatial extents without requiring the network to learn position- or scale-dependent features. The ablation in Table 4 confirms the empirical impact: removing FiLM modulation costs 2.3 dB PSNR on CelebA-HQ and 4.3 IoU on ShapeNet.

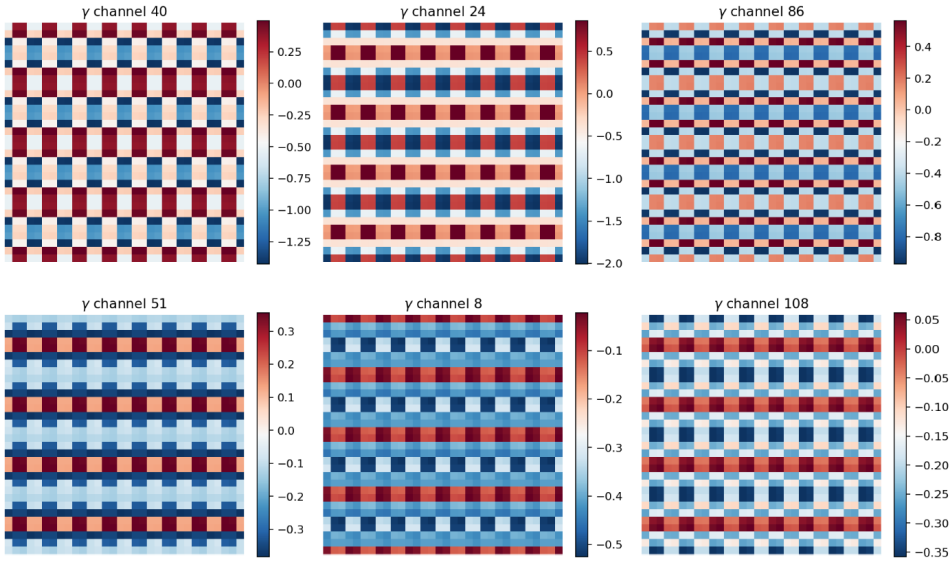


Figure 12: FiLM scale parameter γ (six different channels) evaluated on a 32×32 query grid with the trained CIFAR10 checkpoint. The modulation pattern repeats identically across all spatial groups (see overlay in Figure 9 Block 3), showing how the group-normalized coordinate frame $\tilde{x} = (x - \mu_g)/\lambda_g$ makes the FiLM output invariant to the spatial position of the group. Since all groups share the same spatial extent on uniform grids, this figure demonstrates translation invariance; on variable-geometry data (e.g. point clouds), the same property additionally ensures scale invariance across groups of different sizes.

D LH-NeF Implementation Details

D.1 LH-NeF tokenizer configuration

The LH-NeF tokenizer processes re-ordered inputs with Hierarchical Perceiver (Carreira et al., 2022) grouped attention blocks. Each block is characterized by its number of groups G, tokens per group K, channels C, and number of self-attention layers. Tokens are progressively regrouped from many small groups (fine locality) to fewer, larger groups (coarser, more global context). The conditioning representation is the output of the final block: a structured token set of shape $[B, G, K, C]$ that serves as the per-sample representation for the LH-NeF renderer. Unlike the original HiP, our bottleneck representation is multi-group (rather than single group), since we aim to exploit the locality of each group at query time.

Table 7 shows the block hierarchy for each dataset. Compute is concentrated at the final block (18-36 self-attention layers) with lighter processing at earlier stages (1-3 layers).

Table 7: LH-NeF tokenizer block hierarchy per dataset. Each row shows the LH-NeF tokenizer variant used on each dataset; the conditioning representation is extracted from the final block. G: groups, K: tokens/group, C: channels, SA: self-attention layers. ImageNet uses 4 blocks (block 0–3); all others use 3 (block 0–2).

	Block 0			Block 1			Block 2			Block 3			
Dataset	G	K/C	SA	G	K/C	SA	G	K/C	SA	G	K/C	SA	Cond. dims
CIFAR-10	128	4/64	1	64	4/96	2	32	4/16	18	–	–	–	2,048
CelebA-HQ 64^2	256	4/96	1	128	4/128	3	64	2/24	36	–	–	–	3,072
ImageNet1k 256^2	1,024	4/64	1	512	4/96	2	256	4/128	24	128	4/28	6	14,336
ShapeNet16	256	4/64	1	128	4/32	2	64	3/8	24	–	–	–	1,536
ERA5	288	4/96	1	144	4/128	3	72	2/16	36	–	–	–	2,304

Input embedding. All variants use sinusoidal positional encoding on input coordinates before projection into the HiP embedding space. Table 8 lists the embedding hyperparameters.

Table 8: LH-NeF input embedding and ordering configuration per dataset.

	CIFAR-10	CelebA-HQ 64^2	ImageNet1k 256^2	ShapeNet16	ERA5
Coord. dim d	2	2	2	3	3
Value dim	3 (RGB)	3 (RGB)	3 (RGB)	1 (occ.)	1 (temp.)
Fourier bands	16	32	32	32	32
Max frequency	40	40	40	20	20
Embedding channels	64	64	64	64	128
Coord. ordering	Morton	Morton	Morton	k-d tree	S2 cell

D.2 LH-NeF RENDERER CONFIGURATION

Table 9 lists the LH-NeF renderer hyperparameters per dataset. All configurations render from the LH-NeF tokenization, with a point MLP head, Gaussian-windowed KNN routing with learnable σ, and FiLM modulation.

Table 9: LH-NeF renderer hyperparameters per dataset.

	CIFAR-10	CelebA-HQ 64^2	ImageNet1k 256^2	ShapeNet16	ERA5
d_{model} (cross-attn)	128	128	256	256	128
Num. heads	8	8	8	8	4
k (KNN groups)	4	4	4	8	2
σ_{init} (Gaussian)	0.25	0.35	0.5	0.75	0.35
MLP hidden	128	384	768	256	192
MLP depth	3	5	4	4	3
FiLM hidden	256	256	256	256	512
FiLM PE dim	16	16	16	8	8
Coord PE dim	32	16	16	32	32

Conditioning dimensions. The $G \times K \times C$ token representation is the primary per-sample conditioning. In addition, the LH-NeF renderer receives per-group *routing metadata*: the group center $\mu_g \in \mathbb{R}^d$ (mean coordinate of input points assigned to group g) and group scale $\lambda_g \in \mathbb{R}^d$ (bounding-box extent), for a total of $2Gd$ additional dimensions. These are computed from the input *coordinates* only (i.e. they are not learnable parameters); the renderer uses them for KNN group routing and relative-coordinate normalization in the FiLM path. For fixed-grid data (images, ERA5, voxels), μ_g and λ_g are constant across samples and carry zero bits of per-sample information. For variable-input data (e.g. point clouds), the routing metadata varies per sample. Table 10 reports all dimensions per dataset.

Table 10: Per-sample representation dimensions: token representation ($G \times K \times C$) and routing metadata ($2Gd$ dimensions for group centers and scales).

Dataset	d	Tokens dim. (GKC)	Routing dim. ($2Gd$)	Total
CIFAR-10	2	2,048	128*	2,176
CelebA-HQ 64^2	2	3,072	256*	3,328
ImageNet1k 256^2	2	14,336	512*	14,848
ShapeNet16	3	1,536	384*	1,920
ERA5	3	2,304	432*	2,736

* Constant across samples (fixed-grid, uniformly sampled data).

D.3 Training configuration

All datasets use AdamW for the tokenizer and Adam for the renderer, with cosine learning rate decay. Checkpoints are selected by the primary reconstruction metric: PSNR for images, IoU for 3D occupancy, MSE for ERA5.

Table 11: LH-NeF training configuration per dataset.

	CIFAR-10	CelebA-HQ 64^2	ImageNet1k 256^2	ShapeNet16	ERA5
Tokenizer observations N	1,024	4,096	16,384*	32,768	4,140
Batch size	384	128	52	40	64
Epochs	1,000	1,000	100	300	1,000
Tokenizer LR	1.2e-3	2.8e-4	5.5e-4	1.3e-3	1.8e-3
Tokenizer weight decay	5.3e-3	1.6e-4	1.7e-2	2.1e-3	7.0e-3
Renderer LR	1.8e-4	9.0e-4	1.6e-4	4.9e-4	2.7e-4
Loss	L1	L1	L1	L1	MSE
Selection metric	PSNR ↑	PSNR ↑	PSNR ↑	IoU ↑	MSE ↓
GPUs	1×A40	1×A100	2×A100	2×H200	1×A40
Training time	∼18 h	∼29 h	∼87 h	∼25 h	∼64 h

* The ImageNet tokenizer observes the image downsampled to 128^2; the renderer is supervised and evaluated at 256^2.

E Detailed Component Ablation Analysis

We extend the discussion of Table 4 from the main text.

Locality-preserving ordering. Locality dominates the ablations on dense tasks (reconstruction, generation), but classification is the exception (-1.2 pp without locality). This is because our classifier pools the spatial structure of the LH-NeF tokens via a global `[CLS]` readout (Appx. F.2), which discards geometric structure from the representation. Dense tasks like reconstruction and generation rely directly on the spatial structure of the tokens, where locality has a much larger effect.

Renderer components on CelebA-HQ reconstruction. *Gaussian weighting* and *relative-coordinate FiLM modulation* tie at -2.3 dB each, while *multi-group routing* ($k{>}1$) contributes less (-0.7 dB). The latter result mirrors Spatial Functa (Bauer et al., 2023), which reports that single-neighbor latent lookup matches or beats bilinear interpolation on images, and LIIF (Chen et al., 2021), which reports small but consistent gains from using four local neighbors for super-resolution; a task in which the field is queried often close to group boundaries, where continuity through multiple neighbors is more important.

Renderer components on CelebA-HQ generation. For generation, every renderer component except $k{>}1$ becomes significant, and both locality-preserving orderings remain effective with minor gains for Morton over k-d tree.

Renderer components on ShapeNet16. The ranking flips on ShapeNet: $k{>}1$ and Gaussian weighting become the second most impactful (-8.4 and -7.7 IoU). Multiple neighbors help on

ShapeNet because the renderer queries a full 32^3 grid that includes many empty-space voxels: queries near shape boundaries can borrow representational capacity from empty-space group tokens via multi-group routing. Renderer ablations barely move ShapeNet classification ($<$1 pp), by the same global-readout argument as above.

E.1 Ablation on resolution scaling

To test whether the benefit of locality persists as the number of observations grows, we repeat the ordering ablation of Table 4 on CelebA-HQ at double the resolution 128^2 (i.e. four times as many observations) and analyze the reconstruction gap w.r.t. raster ordering. The two runs differ only in the ordering, and the number of groups is multiplied by four ($64 \to 256$ at the conditioning block) so that the number of observations per group and the compression ratio match the 64^2 setting.

At 128^2 resolution, Morton ordering reaches 36.4 dB test PSNR and raster ordering 29.9 dB, a gap of 6.4 dB; roughly the same gap measured at 64^2 resolution (36.3 vs. 29.9 dB). This is consistent with the geometry of Appendix B.2 at matched compression: a Morton group is still a compact 8×8 block, which covers an ever smaller part of the domain, and a raster group of 64 consecutive pixels remains a segment of a single row at any resolution. The penalty of raster ordering therefore does not shrink as the number of observations grows.

E.2 Ablation on tokenizer with a ViT

We compare the LH-NeF tokenizer with a ViT tokenizer (Dosovitskiy et al., 2021) matched in parameter count (1.30M) and emitting the identical token interface: 8×8 patches, so that on the 64^2 grid its patches coincide with our conditioning-block groups, and 64 groups $\times$ 2 tokens $\times$ 24 channels. Renderer, losses and training protocol are shared, and both runs use our group interface and routing, so the swap isolates the tokenizer choice only. All measurements are on CelebA-HQ 64^2.

Reconstruction. The ViT tokenizer reaches 37.2 dB test PSNR, against 36.3 dB for ours. This near-parity is expected: on a regular grid, patchification hard-codes exactly the partition that our tokenizer computes from raw coordinates (contiguous Morton groups of 64 pixels on the 64^2 grid are the aligned 8×8 patches, Proposition B.1). The ViT tokenizer ablation is therefore an alternative pipeline akin to ours that implements locality through 2D patchification, and the components that turn tokens into a continuous field (centroid routing, Gaussian weighting and FiLM modulation) are ours in both; removing Gaussian weighting or FiLM modulation costs 2.3 dB each (Table 4). What the ViT tokenizer cannot do is leave the 2D grid: patchification requires a full regular grid, whereas the same LH-NeF tokenizer runs unchanged on sampled point sets, sparse 3D occupancy and the sphere.

Computational cost. At a fixed patch size, the ViT attends globally over all patches at every layer, at cost $O(N^2)$ in the number of observations N. Our tokenizer keeps the group size constant and grows the number of groups with N, so all attention happens inside constant-size groups: $O(N)$ attention plus an $O(N \log N)$ sort. Table 12 reports forward FLOPs per image. Ours grows by a factor 4.0 for every $4\times$ increase in pixels, whereas the ViT approaches the factor 16 of its quadratic term; fitting $aN + bN^2$ to the ViT entries places the crossover at a resolution of about 620^2. We keep the patch size at 8×8 because this is the token-matched condition: both tokenizers then produce $N/64$ groups ($N/32$ tokens) at every resolution.

On the bigger picture, at 64^2, the tokenizer is not the bottleneck: a full training step (tokenization, rendering of all 4096 queries, loss, forward and backward) costs 4.8 ms per image with our tokenizer and 3.6 ms with a ViT tokenizer, because the shared renderer and loss account for 3.4 ms in both. Our configuration was not tuned for cost (it stacks 36 self-attention layers over 2-token groups at the conditioning block, which is likely redundant), so we regard the gap at 64^2 as a property of this configuration rather than of the design, and matched FLOPS at this resolution could in principle be explored through other hyperparameter choices.

Table 12: Forward GFLOPs per image of the LH-NeF tokenizer and the parameter-matched ViT tokenizer. The ViT keeps 8×8 patches; ours keeps the group size and multiplies the number of groups by 4 per doubling of the resolution, so both produce $N/64$ groups ($N/32$ tokens).

GFLOPs / image	64^2	128^2	256^2	512^2	1024^2
ViT, parameter-matched (1.3M)	0.18	0.84	5.64	58.8	815
LH-NeF tokenizer (1.3M)	1.27	5.09	20.3	81.4	325.4

E.3 Sensitivity analysis on number of neighbors k

Table 13 trains each k from scratch with the same recipe on the two datasets of Table 4; $k = 1$ is the hard-routing row of that table. The configurations reported in the main text use $k = 4$ on CelebA-HQ and $k = 8$ on ShapeNet16 (Table 9).

Table 13: Reconstruction as a function of the number of routed groups k, each trained from scratch.

routed groups k	1 (hard)	2	4	8
CelebA-HQ 64^2, test PSNR (dB)	35.6	36.2	36.3	36.3
ShapeNet16, test IoU	84.8	87.5	93.8	93.2

Sensitivity is low above an elbow, whose position we attribute to how unevenly signal complexity is distributed over the domain. On sparse 3D occupancy, groups covering empty space have capacity to spare, while groups on the surface of the shape benefit from borrowing it from further neighbours, so borrowing matters: relative to $k = 4$, $k = 2$ costs 6.3 IoU and $k = 1$ costs 9.0. On images, every group covers an equally informative patch on average, so there is little to borrow: $k = 1$ costs only 0.7 dB and the results are flat for $k \geq 2$. This suggests choosing the number of routed groups by how unevenly complexity is spread over the observation set, rather than by data type or modality. In practice, hard routing is nearly equivalent and cheaper on images, a small $k > 1$ helps on sparse data, and over-provisioning is safe (36.3 vs. 36.3 dB and 93.8 vs. 93.2 IoU beyond the elbow).

F Downstream Task Implementation Details

All downstream models are trained on frozen tokenizations with no gradient flow to the tokenizer or renderer.

F.1 Generation (Latent Diffusion)

Generation uses a diffusion transformer trained on the (frozen) tokenizations learned during the LH-NeF training (reconstruction stage). Training and evaluation follows three steps: (i) preprocessing (tokenization extraction and normalization), (ii) diffusion model training, and (iii) sampling and rendering for FID computation.

Tokenization extraction and normalization. Given a frozen LH-NeF model, we run the LH-NeF tokenizer on every training and validation sample, extracting the grouped token representation $\mathbf{Y}^{(L)} \in \mathbb{R}^{G_L\times K_L\times C_L}$ from the final block (for brevity we write G, K, C for G_L, K_L, C_L throughout this section). Along with each token set we store the group routing metadata (centroids $\mu_g \in \mathbb{R}^d$ and extents $\lambda_g \in \mathbb{R}^d_{>0}$). Tokenizations are cast to float16 and written to sharded archives for efficient loading. We compute running per-channel statistics (mean, std) across all training samples; during diffusion training, tokenizations are normalized as $\hat{\mathbf{Y}}^{(L)} = (\mathbf{Y}^{(L)} - \mathbf{m})/\mathbf{s}$ where $\mathbf{m}, \mathbf{s} \in \mathbb{R}^C$ are the per-channel mean and standard deviation (broadcast over groups and tokens).

Diffusion model architecture. The target of the latent diffusion model is the (normalized) tokenization $\hat{\mathbf{Y}}^{(L)} \in \mathbb{R}^{G\times K\times C}$. We use a DiT variant adapted to the grouped structure: self-attention within each group's K tokens, and regrouping across blocks via cross-attention from learnable latent queries ($G \to G/4 \to 1$), so that a single global-attention layer at the coarsest level provides full

cross-group communication. A symmetric decoder path with skip connections expands back to the original grouping (U-Net style). The grouping reduces self-attention cost from $\mathcal{O}((GK)^2)$ to $\mathcal{O}(GK^2 + G^2)$ across the hierarchy.

Following DiT (Peebles & Xie, 2023), all sublayers use adaLN-Zero conditioning from a noise-level embedding (the EDM $c_{\text{noise}}(\sigma)$ passed through random Fourier features), with zero-initialized gates. Continuous coordinate embeddings are computed from the group centroids μ_g via a random Fourier feature projection. Since all K tokens within a group share the same positional embedding (their group centroid μ_g), we add a learned per-position token-ID embedding ($j \in \{1, \ldots, K\}$) to break within-group permutation symmetry.

Training. We adopt the EDM formulation (Karras et al., 2022). Writing $\mathbf{z}_0 = \hat{\mathbf{Y}}^{(L)}$ for the clean (normalized) tokenization, we perturb it with continuous, variance-exploding noise $\mathbf{z}_\sigma = \mathbf{z}_0 + \sigma\,\epsilon$, and wrap the network with Karras preconditioning so that the denoised estimate is

$$\hat{\mathbf{z}}_0 = c_{\text{skip}}(\sigma)\,\mathbf{z}_\sigma + c_{\text{out}}(\sigma)\,F_\theta\big(c_{\text{in}}(\sigma)\,\mathbf{z}_\sigma,\; c_{\text{noise}}(\sigma)\big), \quad c_{\text{noise}}(\sigma) = \tfrac{1}{4}\ln\sigma, \tag{15}$$

with the standard $c_{\text{skip}}, c_{\text{out}}, c_{\text{in}}$ coefficients of Karras et al. (2022). Noise levels are sampled as $\ln\sigma \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$ ($P_{\text{mean}}{=}{-}1.2$, $P_{\text{std}}{=}1.2$), and we minimize the EDM-weighted denoising loss

$$\mathcal{L}_{\text{dm}} = \mathbb{E}_{\sigma,\epsilon}\big[\lambda(\sigma)\,\|\hat{\mathbf{z}}_0 - \mathbf{z}_0\|^2\big], \quad \lambda(\sigma) = \frac{\sigma^2 + \sigma_{\text{data}}^2}{(\sigma\,\sigma_{\text{data}})^2}, \tag{16}$$

where σ_{data} is estimated from the normalized tokenizations (≈ 1). We use AdamW and maintain an exponential moving average (EMA) of model weights for sampling.

Sampling. We generate samples with the deterministic ODE sampler of Karras et al. (2022), using 18 steps over the Karras noise schedule ($\rho{=}7$, $\sigma_{\text{min}}{=}0.002$, $\sigma_{\text{max}}{=}80$, no stochastic churn). Sampled tokenizations are denormalized ($\mathbf{Y}^{(L)} = \hat{\mathbf{Y}}^{(L)} \cdot \mathbf{s} + \mathbf{m}$), reshaped to $[B, G, K, C]$, and passed through the frozen LH-NeF renderer to produce images. The renderer receives the sampled tokens as its tokenizer block output and queries a full coordinate grid at the target resolution. For evaluation, we generate 50,000 samples and compute FID against the real training set.

Notes on generation on variable geometries. Rendering a generated sample requires not only the grouped token representation but also the routing metadata: group centroids $\mu_g \in \mathbb{R}^d$ and extents $\lambda_g \in \mathbb{R}^d_{>0}$. For uniformly sampled data, the routing metadata is constant across all samples, therefore the diffusion model only needs to generate the grouped token representation, and rendering uses the fixed-grid routing metadata. For data on variable geometries (e.g. on point clouds), the group partition depends on the distribution of observed points for each input, which differs per sample. The routing metadata must therefore be generated alongside the token representation, e.g. by concatenating it to the diffusion target.

Hyperparameters. Table 14 lists the diffusion model architecture and training configuration for each dataset.

F.2 Classification

ShapeNet16. The ShapeNet16 classifier is a transformer encoder operating on the tokenization extracted with a frozen LH-NeF tokenizer forward pass and per-channel normalized as in §F.1. We add two kinds of blocks: *intra-group* blocks self-attend within each group's K tokens, and *inter-group* blocks pool each group to a single vector via mean, self-attend across the G group representations, and broadcast back. This factorization is added for efficiency, with the added benefit of matching the grouped structure of our tokenization. Both block types alternate for a total of 8 layers. We then prepend a learnable CLS token to the per-group pooled representation and apply a 2-layer transformer encoder; the CLS output is fed to a linear classification head. Positional information is injected via random Fourier feature embeddings of the group centers μ_g and a learned per-slot token-ID embedding to break within-group permutation symmetry.

Table 14: Diffusion model hyperparameters per dataset. Input structure (G, K, C) is inherited from the tokenizer output representation.

	CIFAR-10	CelebA-HQ 64^2
Input structure (from LH-NeF tok.)		
Groups G	32	64
Tokens/group K	4	2
Token dim C	16	24
Sequence length $L = GK$	128	128
Architecture		
Hidden dim (per level)	[384, 384, 480, 576]	[384, 384, 480, 576]
Depth (enc/proc/dec blocks)	3 (7 total)	3 (7 total)
Self-attend layers/block	[1, 2, 4, 4, 4, 2, 1]	[1, 2, 4, 4, 4, 2, 1]
Attention heads	6	6
MLP widening factor	4	4
Dropout	0	0
Diffusion (EDM)		
Parameterization	precond. $\hat{\mathbf{z}}_0$	precond. $\hat{\mathbf{z}}_0$
σ_{data}	≈ 1 (measured)	≈ 1 (measured)
P_{mean} / P_{std}	-1.2 / 1.2	-1.2 / 1.2
σ_{min} / σ_{max}	0.002 / 80	0.002 / 80
ρ	7	7
EMA rate	0.99995	0.99995
Training		
Optimizer	AdamW	AdamW
Learning rate	1.0×10^{-4}	1.53×10^{-4}
Weight decay	0.01	0
Batch size	128	128
Epochs	5000	6000
Sampling		
Sampler	Heun (EDM ODE)	Heun (EDM ODE)
Steps	18	18
Stochastic churn S_{churn}	0	0
Samples for FID	50,000	50,000

Table 15: ShapeNet16 classification hyperparameters.

	ShapeNet16
d_{model}	256
Depth (blocks)	8
Attention heads	8
MLP ratio	4.0
Dropout	0.1
Label smoothing	0.0
Pooling	CLS token
Optimizer	AdamW
Learning rate	1.5e-3
Weight decay	0.29
Batch size	256
Epochs	100

CIFAR-10. Following the protocol of Spatial Functa (Bauer et al., 2023) and ENF (Wessels et al., 2025), we extract frozen tokenizations on a $50\times$-augmented training set (random 32×32 crops of 40×40 zero-padded images, horizontal flips) and an unaugmented val/test set, apply per-channel normalization as in §F.1, and train a classifier on the resulting tokenizations.

The tokenization $\mathbf{Y}^{(L)} \in \mathbb{R}^{G\times K\times C}$ is reshaped into a 2D feature map for convolutions: we order the G groups in row-major order by their centroids μ_g, arrange the K slots within each group as an

$H_k \times W_k$ sub-grid, and concatenate to obtain $\mathbf{X} \in \mathbb{R}^{C\times H\times W}$ with $H = H_g H_k$, $W = W_g W_k$ (so $G = H_g W_g$ and $K = H_k W_k$). We then apply a 2-stage ConvNeXt (Liu et al., 2022) classifier: a 1×1 stem to projection dim D_0 (no patchify, since $\mathbf{X}$ is already low-resolution), ConvNeXt blocks with stochastic depth, a strided downsampling between stages, global-average pooling, and a linear head. We additionally apply two latent-space augmentations during classifier training: Mixup (Zhang et al., 2018) on the feature map and random zeroing of spatial cells. Both are disabled at evaluation.

Table 16: CIFAR-10 and ImageNet-1k classification hyperparameters (frozen-latent ConvNeXt classifier).

	CIFAR-10	ImageNet-1k
D_0, D_1, D_2 (per-stage widths)	192, 384	192, 384, 768
Blocks per stage	4, 4	3, 3, 6
Depthwise kernel size	3×3	5×5
MLP ratio	4	4
Stochastic depth (drop_path)	0.2	0.3
LayerScale init	10^{-6}	10^{-6}
Head dropout	0.1	0.1
Group grid (H_g, W_g)	$(4, 8)$	$(8, 16)$
Slot grid (H_k, W_k)	$(2, 2)$	$(2, 2)$
Mixup α	0.5	0.2
CutMix α	–	1.0
Token drop p	0.1	0
Label smoothing	0.1	0.1
Optimizer	AdamW	AdamW
Learning rate (peak)	1.24×10^{-3}	2×10^{-3}
Weight decay	4.15×10^{-2}	5×10^{-2}
LR schedule	cosine, 1000-iter warmup	cosine, 12,400-iter warmup
Batch size	1024	1024
Epochs	200	48

ImageNet-1k. We follow the same protocol with the ImageNet-1k 256^2 tokenizer. We hold out 1% of the training images for validation (12,812 images) and report accuracy on the official validation set (50,000 images) as the test set. Each remaining training image is tokenized 10 times under different augmentations (random resized crops with scale in $[0.08, 1]$, horizontal flips, and RandAugment with 2 operations of magnitude 5), and the validation and test images once without augmentation. The classifier is the ConvNeXt classifier described above with three stages instead of two (35.2M parameters), and applies CutMix (Yun et al., 2019) on the latent feature map in addition to Mixup, choosing between the two with probability 0.5 per batch (Table 16). We keep an exponential moving average (EMA) of the classifier weights with decay 0.9999, select the epoch with the highest validation accuracy of the EMA weights, and report the test accuracy of the weights of that epoch.

F.3 ERA5 FORECASTING

We follow the ENF protocol (Wessels et al., 2025), where consecutive hourly ERA5 temperature snapshots are tokenized and per-channel normalized (as in §F.1), and a temporal predictor F_ψ maps $\mathbf{Y}_t \mapsto \mathbf{Y}_{t+1}$ in tokenization space. Training uses a function-space loss: the predicted tokenization $\hat{\mathbf{Y}}_{t+1} = F_\psi(\mathbf{Y}_t)$ and the ground-truth tokenization $\mathbf{Y}_{t+1}$ are both decoded through the frozen LH-NeF renderer, and the loss is the MSE between the decoded temperature fields.

Our temporal predictor uses the DiT architecture used for generation (§F.1), with the only difference being the conditioning mechanism: the diffusion model uses adaLN-Zero from a noise-level embedding, while the forecaster uses standard pre-norm (no timestep conditioning, as this is a deterministic prediction task). The model predicts a *residual* $\Delta\mathbf{Y} = F_\psi(\mathbf{Y}_t)$, and the forecast is $\hat{\mathbf{Y}}_{t+1} = \mathbf{Y}_t + \Delta\mathbf{Y}$. ENF uses a PΘNITA MPNN (3 layers, hidden dim 256) as their temporal predictor.

As in ENF, ERA5 results are evaluated through mean squared errors on temperatures normalized to $[0, 1]$ with the global minimum and maximum of the dataset of Dupont et al. (2022b) (202.66 K

and 320.93 K), computed over the full 46×90 grid. Our models are trained on temperatures mapped to $[-1, 1]$ while Functa and ENF predict temperatures in a $[0, 1]$ scale; at evaluation time, we re-normalize our predictions back to $[0, 1]$ for MSE calculation. T_t-MSE is averaged over all $2{,}420$ test snapshots and T_{t+1}-MSE over the 443 pairs of consecutive hourly test snapshots.

Table 17: ERA5 forecasting model hyperparameters.

	LH-NeF	ENF
Hidden dim	256	256
Depth (blocks)	3	3 (layers)
Self-attention layers per block	4	–
Attention heads	4	–
Predict residual	yes	yes
Optimizer	AdamW	Adam
Learning rate	8.4e-5	–
Weight decay	3.9e-3	–
Batch size	128	32
Epochs	500	1,000

G Baseline Reproduction

Parameter counts (marked with ‡ in Table 2) and configurations for the efficiency benchmark are reproduced from reported hyperparameters using the authors' public code where possible.

Functa (Dupont et al., 2022a). We use the authors' official JAX/Haiku implementation directly, with the per-dataset hyperparameters reported in their paper. All Functa models use a SIREN with hidden width 512 and ω_0=30.

Table 18: ENF reproduction hyperparameters per dataset. CIFAR-10 hyperparameters and parameter count (522K) are reported by the authors. For all other datasets, parameter counts are computed from the authors' official JAX implementation using reported hyperparameters.

	CelebA-HQ 64^2	ImageNet1k 256^2	ShapeNet16	ERA5
Hidden dim	256	128	128	128
Latent dim	64	64	32	64
Num. latents	36	169	27	36
Num. heads	3*	3	3	3
k (nearest)	4	4	4	4
σ_q, σ_v	2.0, 10.0	2.0, 10.0	2.0, 10.0	2.0, 8.0
Params	3.2M	817K	813K	817K

*Not reported for CelebA-HQ; we use the same number of heads as in all other datasets (3).

Spatial Functa (Bauer et al., 2023). Spatial Functa does not provide publicly available code. We use an unofficial public JAX reimplementation (Papa, 2024) to compute parameter counts. Because this reimplementation is not validated against the authors' original and to avoid reporting inaccurate results, we exclude Spatial Functa from the efficiency benchmark and report only parameter counts.

Table 19: Functa reproduction hyperparameters per dataset.

	CIFAR-10	CelebA-HQ 64^2	ShapeNet16	ERA5
Hidden width	512	512	512	512
Hidden layers	10	13	15	16
ω_0	30	30	30	30
Params	2.6M	3.4M	4.0M	4.1M

ENF (Wessels et al., 2025). For CIFAR-10, the authors report 522K parameters. For all other datasets, we instantiate ENF's official JAX implementation with the hyperparameters reported in their paper and count parameters directly.

Table 20: Spatial Functa hyperparameters per dataset.

	CIFAR-10	ImageNet1k 256^2
Latent grid	8×8	32×32
Latent channels	16	64
Conditioning dims	1,024	65,536
SIREN width	256	256
SIREN layers	6	8
ω_0	10	10
Modulation	shift-only	shift-only
Lat.-to-mod. map	1×1 Conv	3×3 Conv
Interpolation	1-NN	1-NN
MetaSGD lrs	Yes	Yes
Params	425K	1.4M

H Efficiency Benchmark Protocol

Table 5 benchmarks the cost of a single training step on CelebA-HQ 64^2. Measurements are performed on a single NVIDIA H100 NVL GPU (47 GB MIG partition). We report parameter count, peak memory at batch size 1 and maximum batch size.

Functa. Benchmarked using the authors' official JAX/Haiku implementation (Dupont et al., 2022a) with the reported CelebA-HQ 64^2 hyperparameters (Table 19). The training step is a 3-step MAML inner loop ($\eta{=}10^{-2}$ inner-SGD on the per-sample shift modulations) followed by a final SIREN forward pass and an outer-loop backward pass through the SIREN weights, fully fused as a single JIT-compiled XLA program (matching the ENF setup above). Maximum batch size is found via the same binary-search protocol as ENF.

ENF. Benchmarked using the authors' official JAX implementation, with JAX 0.9.2 and XLA JIT compilation. The training step is a 3-step MAML inner loop followed by an outer-loop backward pass through all model parameters (second-order MAML), compiled as a single fused XLA program. This setup is favorable to ENF: XLA fuses the full inner-loop and outer-gradient computation, eliminating the overhead of retained computation graphs that arises in eager-mode frameworks such as PyTorch. Maximum batch size is found via binary search over $[1, 4096]$ with 3 validation iterations per candidate.

LH-NeF. Benchmarked in PyTorch 2.10 (CUDA 12.8). The training step is a single forward pass through the tokenizer and renderer followed by a backward pass through all parameters. Maximum batch size is found via binary search over $[1, 2048]$ with 5 validation iterations per candidate.

I Qualitative Results

I.1 CelebA-HQ 64^2 generation samples



Figure 13: Additional unconditional generation samples on CelebA-HQ 64^2 from a diffusion transformer trained on the frozen LH-NeF tokenizations.

I.2 SHAPENET16 RECONSTRUCTIONS

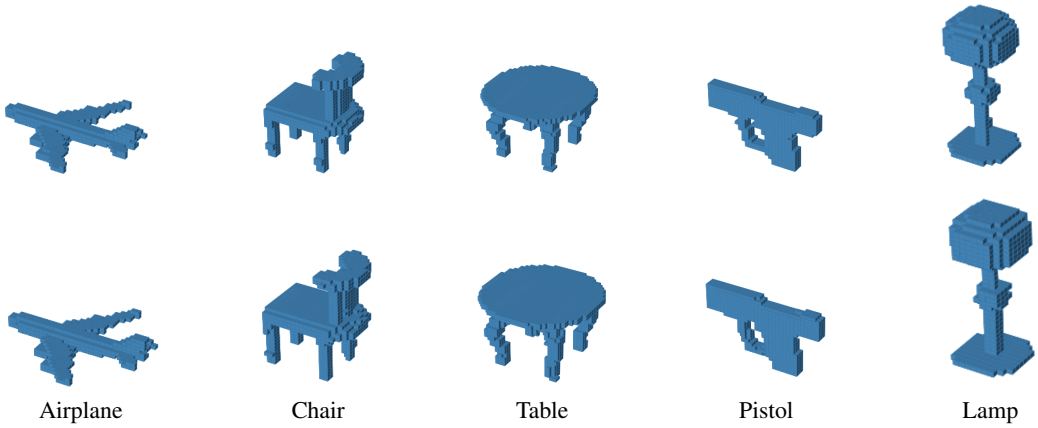


Figure 14: ShapeNet16 voxel occupancy reconstructions at 32^3 resolution. Top: ground-truth occupancy grids. Bottom: LH-NeF reconstructions.