SMOOTH-NGP: C^∞-SMOOTH DIFFERENTIABLE HASH GRIDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Grid-based implicit methods like Instant-NGP greatly speed up neural field training, but their default multilinear interpolation yields only piecewise-smooth functions, which can destabilize higher-order derivatives and create grid-aligned artifacts. These issues limit performance in derivative-sensitive tasks, such as precise normal computation and PDE solving. We present *Smooth-NGP*, a C^∞-smooth variant of Instant-NGP that swaps multilinear interpolation for an infinitely differentiable kernel and extends the hash grid with a sliding-window design. The smooth kernel provides stable closed-form derivatives of any order, supporting derivative-driven losses while retaining the efficiency of grid encodings. However, because the kernel is flat at cell boundaries, its derivatives go to zero there, reducing gradient flow exactly where grid artifacts tend to appear. The sliding-window mechanism resolves this by ensuring each query point falls within the interior of several shifted grid windows, preserving useful gradients across the entire domain. On image fitting, signed distance field learning, and PINN-based PDE solving, Smooth-NGP reliably improves reconstruction quality, derivative fidelity, training stability, and surpasses leading grid-based baselines. We will release the source code and experimental scripts upon publication.

1 INTRODUCTION

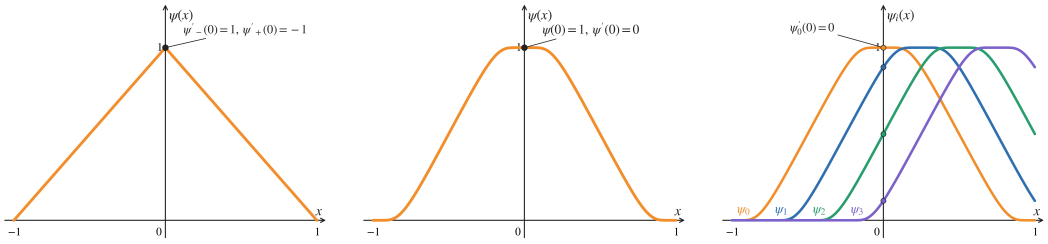
Neural fields represent a signal as a continuous function of spatial coordinates. This perspective underpins radiance fields for view synthesis, signed distance and occupancy fields for 3D geometry, and solution fields in scientific computing. Training typically consists of sampling coordinates and minimizing losses defined by images, point clouds, boundary conditions, or PDE residuals; inference then entails querying the learned field at arbitrary locations. Provided the representation is sufficiently smooth, the same model can also analytically compute spatial derivatives, which are essential for evaluating normals, curvature, and differential operators.

A central challenge, however, is efficiency. Classical neural fields often rely on coordinate-based MLPs that must be evaluated many times during both training and inference, making optimization slow and limiting the recovery of fine local detail. Instant-NGP Müller et al. (2022) addresses this bottleneck with a multi-resolution hash encoding coupled to a very small decoder network Müller et al. (2022). By storing local features in a hierarchy of hash tables and retrieving them through interpolation, Instant-NGP shifts much of the representational burden from the MLP to fast memory lookups, yielding substantial gains in both convergence speed and query efficiency.

Despite this success, standard Instant-NGP inherits a structural limitation from its interpolation scheme. In 3D, hash-grid features are obtained with trilinear interpolation, which guarantees continuity of the interpolated values but not of their derivatives. The resulting feature field is only C^0 across voxel boundaries, and its gradients change piecewise from cell to cell. This behavior is problematic in applications that depend on stable analytical derivatives. In physics-informed learning, unreliable first- and second-order derivatives make PDE optimization difficult. In geometric reconstruction, discontinuous gradients can appear as grid-aligned seams and rough surfaces. Although later variants of Instant-NGP improve aliasing, scene coverage, or reconstruction quality, the smoothness limitation imposed by trilinear interpolation remains.

Our goal is to retain the efficiency of hash-grid neural fields while endowing them with smooth higher-order differentiability. We therefore propose *Smooth-NGP*, a modified Instant-NGP architecture

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.



(a) Linear interpolation (b) Smooth cutoff interpolation (c) Smooth cutoff interpolation with sliding window

Figure 1: Comparison of derivatives at the boundaries for different interpolation functions. (a) Traditional linear interpolation: the derivative is discontinuous at the boundary, leading to gradient jumps that severely degrade geometric smoothness. (b) We adopted a smooth cutoff interpolation function in Equation 4 with $\psi'(0) = 0$: although it eliminates gradient jumps and ensures geometric smoothness, the zero derivative causes learning stagnation (vanishing gradient) at the boundary, resulting in noticeable artifacts. (c) The smooth cutoff interpolation combined with sliding-window hash grids: while the derivative at the first window is zero ($\psi_0'(0) = 0$), the derivatives from overlapping windows ($\psi_1'(0), \psi_2'(0), \psi_3'(0)$) remain non-zero. This architectural design provides continuous gradient flow, enabling the hash grids to continue learning and substantially reducing boundary-aligned artifacts.

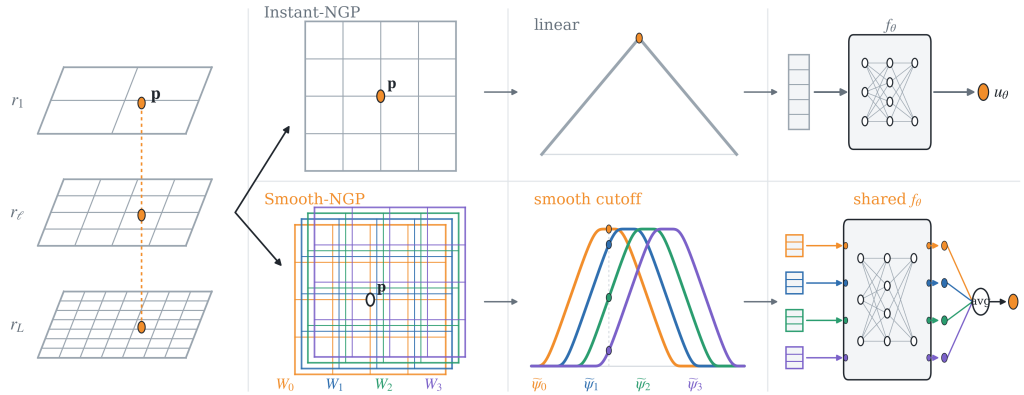
built around two changes. First, we adapt the classical compactly supported C^∞ partition-of-unity kernel of Bruno & Kunyansky (2001); Ying & Zorin (2004) to multiresolution hash encoding, replacing trilinear interpolation while preserving the standard interpolation constraints and yielding smooth transitions across cell boundaries. This gives the encoded feature field well-behaved higher-order analytical derivatives and makes hash-grid feature encodings better suited to derivative-based objectives.

A smooth interpolation kernel alone, however, introduces a new optimization issue. Because the adopted C^∞ kernel is flat at cell boundaries, its derivative vanishes there, which weakens gradient flow exactly where boundary artifacts are most likely to occur. To address this issue, we introduce a sliding-window hash-grid architecture. We evaluate several shifted versions of the grid and aggregate their predictions, so that a point lying on the boundary of one window is represented in the interior of others. This preserves informative gradients throughout the domain, stabilizes training, and further suppresses grid-aligned artifacts. Together, these changes turn Instant-NGP from a fast but piecewise-smooth feature encoding into a C^∞-smooth neural-field representation that remains efficient and supports higher-order derivative objectives.

We show that Smooth-NGP is effective for image fitting, 3D SDF learning, and PDE solving. In image fitting, Smooth-NGP achieves better edge fidelity than Instant-NGP Müller et al. (2022) and ∂^∞-Grid Kairanda et al. (2026). For 3D reconstruction, it markedly mitigates grid-aligned artifacts, improving both geometric accuracy, normal accuracy, and visual quality. For various types of PDEs, we benchmark against Instant-NGP and ∂^∞-Grid, and Smooth-NGP delivers higher-quality solutions than these grid-based baselines. Overall, the results suggest that Instant-NGP-style acceleration can be applied beyond fast reconstruction to a wider range of neural-field tasks where accurate derivatives matter.

2 Related Work

The advent of coordinate-based Multi-Layer Perceptrons (MLPs) has profoundly transformed the landscape of continuous signal parameterization. However, pure MLPs are often bottlenecked by their slow convergence and intensive computational overhead. To address these efficiency concerns, grid-based implicit representations have emerged as a widely adopted alternative. Instant-NGP Müller et al. (2022) popularized this direction by introducing multi-resolution hash encoding. It maps



(a) Multi-res. grids (b) Sliding windows (c) Interpolation (d) Decoder

Figure 2: Comparison between standard Instant-NGP and Smooth-NGP. (a) The same spatial domain is represented by grids of increasing resolutions $r_1, \ldots, r_L$. A representative level is processed by the two pipelines shown in the top and bottom rows of (b)–(d), respectively. (b) Instant-NGP uses a single grid, whereas Smooth-NGP uses four diagonally shifted windows. (c) Instant-NGP applies linear interpolation, while each shifted window evaluates the smooth cutoff kernel at a different local coordinate. (d) The shared decoder f_θ is applied independently to every window, and the resulting predictions are averaged to produce the final output.

coordinates to hierarchical grids and extracts continuous features via multi-linear interpolation, dramatically accelerating training from days to minutes. Following this paradigm, a multitude of grid-based variants have been developed to optimize grid structures and memory efficiency. For example, Plenoxels Fridovich-Keil et al. (2022) and PlenOctrees Yu et al. (2021) employ explicit dense and sparse hierarchical voxel grids, respectively, to achieve extremely fast forward queries. To further mitigate the cubic memory footprint of 3D grids, tensor factorization approaches such as TensoRF Chen et al. (2022), EG3D Chan et al. (2022), and K-Planes Sara Fridovich-Keil and Giacomo Meanti et al. (2023) have been proposed, which cleverly decompose volumetric features into compact, lower-dimensional planes or vectors.

While Instant-NGP established hash grids as highly efficient, its multi-linear encoder is only globally C^0: within each cell, its Hessian has zero diagonal entries (pure second derivatives), whereas its off-diagonal entries (mixed derivatives) are not structurally constrained to vanish. This structural constraint can cause second-derivative errors relative to a target field. A nonlinear decoder can induce nonzero pure second derivatives through the chain rule, but does not restore encoder smoothness across cell boundaries. This limits derivative-sensitive applications such as PDE solving and high-fidelity surface reconstruction. To circumvent this, SDF modeling methods such as iSDF Ortiz et al. (2022), NeuS2 Wang et al. (2023), PermutoSDF Rosu & Behnke (2023), and Neuralangelo Li et al. (2023) resort to numerical gradients, inevitably introducing truncation errors and computational overhead. Similarly, novel view synthesis variants like Zip-NeRF Barron et al. (2023) and Tri-MipRF Hu et al. (2023) employ spatial pre-filtering to reduce aliasing. Although these adaptations improve macroscopic smoothness, the underlying C^0 interpolation remains unchanged. Therefore, fundamentally replacing linear interpolants with analytically smooth kernels is essential to natively support derivative-based objectives without compromising grid speed.

The integration of deep learning with scientific computing has been largely catalyzed by the pioneering work on Physics-Informed Neural Networks (PINNs) Raissi et al. (2019). By embedding governing partial differential equations (PDEs) directly into the loss function as regularization terms, PINNs enforce physical consistency during the training process. While this paradigm demonstrates remarkable versatility, standard PINNs formulated with coordinate-based MLPs face severe optimization challenges, particularly for high-frequency or highly non-linear physical fields Krishnapriyan et al. (2021). Recent research has rigorously analyzed these failure modes, highlighting how spectral

bias, causal violations, and conflicting gradient landscapes significantly impede convergence in non-linear PDE regimes Wang et al. (2024); Daw et al. (2023). To mitigate these gradient pathologies, modern approaches employ adaptive sampling strategies or introduce trust-region calibration, as seen in RoPINN Wu et al. (2024), to better balance optimization and generalization errors. These optimization-focused observations further highlight the essential importance of designing foundational architectures that inherently yield stable, higher-order analytical derivatives, thereby intrinsically regularizing the training dynamics.

In search of more efficient solvers, researchers have explored diverse neural architectures beyond traditional MLPs. A prominent direction involves grid-based representations that aim to combine the efficiency of explicit structures with the smoothness of neural fields. A notable example is the recently proposed ∂^∞-Grid Kairanda et al. (2026), which utilizes differentiable feature grids to ensure infinite analytical differentiability, addressing the derivative breakdown prevalent in standard multi-linear grids. Other innovations include the Latent Neural Operator (LNO) Wang & Wang (2024) and the Newton Informed Neural Operator Hao et al. (2024a), which integrate latent space mapping or traditional numerical solvers into the neural pipeline. As these architectural variants proliferate, the PINNacle benchmark Hao et al. (2024b) provides a crucial standardized framework for evaluating their performance in complex geometries and high-dimensional tasks. These advances collectively drive the development of robust, higher-order differentiable solvers.

3 METHOD

We propose *Smooth-NGP*, a smooth multi-resolution hash representation for coordinate-based neural fields. Let $\Omega \subset \mathbb{R}^d$ denote the spatial domain, and let $u_\theta : \Omega \to \mathbb{R}^m$ be the field predicted by the network. In our experiments, u_θ represents an RGB image, a signed distance field, or a scalar PDE solution. As in Instant-NGP Müller et al. (2022), the model combines a multi-resolution hash-grid encoder with a lightweight MLP decoder. As illustrated in Figure 2, our method modifies this pipeline in two ways: we replace multilinear interpolation with a boundary-flat C^∞ kernel, and we aggregate several shifted grids to avoid the boundary-induced gradient degeneration caused by flat cell boundaries.

3.1 REVIEW OF INSTANT NGP

In standard multi-resolution hash encoding, like Instant NGP Müller et al. (2022), for each level ℓ with resolution r_ℓ, a query point $\mathbf{p} \in [0,1]^d$ yields a scaled coordinate $\mathbf{x}_\ell = r_\ell \mathbf{p}$, which determines the cell index $\mathbf{n}_\ell = \lfloor \mathbf{x}_\ell \rfloor$ and the local coordinate $\mathbf{t}_\ell = \mathbf{x}_\ell - \mathbf{n}_\ell$. The features at the 2^d cell corners, retrieved via a hash function H_ℓ and denoted as $\mathbf{g}_{\ell,\mathbf{v}} = \mathbf{g}_\ell(H_\ell(\mathbf{n}_\ell + \mathbf{v}))$ for each corner offset $\mathbf{v} \in \{0,1\}^d$, are aggregated using tensor-product multi-linear weights defined as:

$$w_{\mathbf{v}}(\mathbf{t}_\ell) = \prod_{j=1}^{d} \phi_{v_j}(t_{\ell,j}), \qquad \phi_0(t) = 1 - t, \quad \phi_1(t) = t. \tag{1}$$

With these weights, the combined level feature is obtained by interpolating across all the features located at the corners of the cube as follows,

$$h_\ell(\mathbf{p}) = \sum_{\mathbf{v} \in \{0,1\}^d} w_{\mathbf{v}}(\mathbf{t}_\ell)\, \mathbf{g}_{\ell,\mathbf{v}}. \tag{2}$$

The final field value is predicted by a lightweight MLP decoder f_θ using the concatenated features from all L levels $u_\theta(\mathbf{p}) = f_\theta(h_1(\mathbf{p}) \oplus \cdots \oplus h_L(\mathbf{p}))$.

The key limitation of multilinear interpolation is that h_ℓ is affine along each coordinate. Within each cell, the encoder Hessian therefore has zero diagonal entries, i.e., $\partial_{x_j x_j} h_\ell = 0$ and $\Delta h_\ell(\mathbf{p}) = 0$, whereas its off-diagonal mixed derivatives $\partial_{x_j x_k} h_\ell$, $j \neq k$, are not structurally constrained to vanish. Thus, the Hessian is not identically zero but is structurally constrained, which can cause second-derivative errors for target fields with nonzero pure derivatives. First-order derivatives are also only piecewise continuous, causing gradient jumps across cell boundaries.

3.2 C^∞ INTERPOLATION

To endow the hash encoding with smooth higher-order analytical derivatives, we adopt a smooth cutoff function $\tilde{\psi}(t)$ that is C^∞-continuous on $\mathbb{R}$ to replace ϕ_0 and ϕ_1. Moreover, $\tilde{\psi}(t)$ should satisfy,

$$\tilde{\psi}(t) = 1, t \in (-\infty, 0] \qquad \tilde{\psi}(t) = 0, t \in [1, +\infty) \qquad \tilde{\psi}(t) + \tilde{\psi}(1-t) = 1, t \in [0, 1] \tag{3}$$

The first two conditions enforce interpolation at the grid vertices, and the third condition yields a partition of unity. The C^∞-continuity ensures the smoothness of the hash encoding.

Several functions satisfy the interpolation conditions in Equation 3. We do not introduce a new cutoff function; instead, we adopt the classical compactly supported C^∞ construction used by Bruno & Kunyansky (2001); Ying & Zorin (2004). Let $L(t) = \frac{2\exp(-1/t)}{t-1}$, $0 < t < 1$, and define $\tilde{\psi}(t)$ as follows:

$$\tilde{\psi}(t) = \begin{cases} 1, & t \le 0, \\ \dfrac{1}{1 + \exp(L(1-t) - L(t))}, & 0 < t < 1, \\ 0, & t \ge 1. \end{cases} \tag{4}$$

This construction of $\tilde{\psi}(t)$ is C^∞-continuous and satisfies all three conditions in Equation 3.

Following Instant NGP, we define $\tilde{\phi}_0(t) = \tilde{\psi}(t), \qquad \tilde{\phi}_1(t) = 1 - \tilde{\psi}(t), \qquad \tilde{w}_{\mathbf{v}}(\mathbf{t}) = \prod_{j=1}^{d} \tilde{\phi}_{v_j}(t_j)$, and replace h_ℓ with the smooth aggregated level-wise feature encoding

$$\tilde{h}_\ell(\mathbf{p}) = \sum_{\mathbf{v} \in \{0,1\}^d} \tilde{w}_{\mathbf{v}}(\mathbf{t}_\ell)\, \mathbf{g}_{\ell,\mathbf{v}}.$$

The resulting cell-local basis functions are compactly supported and form a partition of unity. The composed features $\tilde{h}_\ell$ are smooth across cell boundaries. For any multi-index α, the partial derivative can be explicitly expressed as:

$$\frac{\partial^{|\alpha|} \tilde{h}_\ell(\mathbf{p})}{\partial \mathbf{p}^\alpha} = r_\ell^{|\alpha|} \sum_{\mathbf{v} \in \{0,1\}^d} \frac{\partial^{|\alpha|} \tilde{w}_{\mathbf{v}}(\mathbf{t}_\ell)}{\partial \mathbf{t}_\ell^\alpha}\, \mathbf{g}_{\ell,\mathbf{v}}. \tag{5}$$

This equation clearly shows that the smoothness of the n-th order partial derivative of the hash grid feature encoding with respect to the input coordinates $\mathbf{p}$ (with $n = |\alpha|$) is fundamentally determined by the smoothness of the n-th order partial derivative of $\tilde{w}_{\mathbf{v}}$ with respect to the local coordinates $\mathbf{t}_\ell$.

Consequently, the adopted interpolation function $\tilde{\psi}(t)$ fulfills the fundamental criteria of grid interpolation, and its infinite differentiability equips the hash grid with higher-order analytical derivatives. In contrast, the Gaussian function employed by ∂^∞-Grid Kairanda et al. (2026) lacks compact support and therefore must be truncated, which introduces discontinuities.

3.3 SLIDING WINDOWS

As illustrated in Figure 1, the globally C^∞ constant extension in Equation 4 satisfies $\tilde{\psi}^{(k)}(0) = \tilde{\psi}^{(k)}(1) = 0$ for all $k \ge 1$, causing the gradients of $u_\theta(\mathbf{p})$ to vanish along grid cell boundaries. Standard interpolation does not have this boundary-flat property and can introduce grid-aligned artifacts.

We address this issue by shifting the feature grids, ensuring that each space point is covered by multiple smooth cutoff functions $\tilde{\psi}(t)$ with different values of t. Specifically, we introduce N sub-grid shifts $\mathcal{D} = \{\boldsymbol{\delta}_i\}_{i=0}^{N-1}$ with $\boldsymbol{\delta}_i \in [0, 1)^d$. Crucially, each window maintains an independent feature table $\mathbf{g}_{\ell,i}$. For a query point $\mathbf{p}$, its shifted coordinate at level ℓ is $\mathbf{x}_{\ell,i} = r_\ell \mathbf{p} + \boldsymbol{\delta}_i$. Following the standard encoding process (described in Section 3.1) to derive the cell index $\mathbf{n}_{\ell,i}$ and local coordinate $\mathbf{t}_{\ell,i}$, the corner features $\mathbf{g}_{\ell,i,\mathbf{v}} = \mathbf{g}_{\ell,i}(H_\ell(\mathbf{n}_{\ell,i} + \mathbf{v}))$ are interpolated to compute the representation for the i-th window:

$$\tilde{h}_{\ell,i}(\mathbf{p}) = \sum_{\mathbf{v} \in \{0,1\}^d} \tilde{w}_{\mathbf{v}}(\mathbf{t}_{\ell,i})\, \mathbf{g}_{\ell,i,\mathbf{v}}.$$

As illustrated in Figure 2b, we demonstrate this operation using $N = 4$ windows. Starting from the standard hash grid, we shift it along the diagonal direction by an offset of $\boldsymbol{\delta}_i$ (where $i = 0, \ldots, N-1$) to form N sliding-window hash grids. The complete feature for the i-th window is concatenated by $E_i(\mathbf{p}) = \tilde{h}_{1,i}(\mathbf{p}) \oplus \cdots \oplus \tilde{h}_{L,i}(\mathbf{p})$, and the corresponding prediction is $u_{\theta,i}(\mathbf{p}) = f_\theta(E_i(\mathbf{p}))$. The final output is the average over all windows:

$$u_\theta(\mathbf{p}) = \frac{1}{N} \sum_{i=0}^{N-1} u_{\theta,i}(\mathbf{p}). \tag{6}$$

In all experiments, we use diagonal shifts $\boldsymbol{\delta}_i = \frac{i}{N}\mathbf{1}$, $i = 0, \ldots, N-1$. This construction ensures that any point lying on the boundary of one window is evaluated as an interior point in other windows, thereby providing arbitrary gradients near cell boundaries. The selection of the number of windows N introduces an inherent trade-off between representational fidelity and computational cost: increasing N enhances accuracy but also raises computational demands. In practice, we fix $N = 4$ for both 2D and 3D settings to strike a favorable compromise between precision and efficiency.

3.4 Smooth decoder and analytical derivatives

We use lightweight MLP decoders with C^∞-smooth activation functions, such as $\tanh$. Combined with the smooth NGP feature encoder, these activations avoid introducing derivative discontinuities and preserve the arbitrary-order differentiability of u_θ. The decoder architecture is configured per task as described in each experimental setup.

For a scalar output field and a single window i, the Laplacian admits the decomposition

$$\Delta u_{\theta,i}(\mathbf{p}) = \mathrm{tr}\big(J_{E_i}(\mathbf{p})^\top H_{f_\theta}(E_i(\mathbf{p})) J_{E_i}(\mathbf{p})\big) + \nabla f_\theta(E_i(\mathbf{p}))^\top \Delta E_i(\mathbf{p}), \tag{7}$$

where J_{E_i} denotes the Jacobian of the feature encoder and H_{f_θ} denotes the decoder Hessian with respect to its input. The complete model combines these per-window terms by averaging them as in Eq. (6). Under multilinear interpolation, the encoder Hessian has zero diagonal entries within each cell, so $\Delta E_i(\mathbf{p}) = 0$, whereas its off-diagonal entries are not constrained to vanish. Hence, pure second-order terms in the decoded output arise only from the decoder-Hessian term. Smooth-NGP instead allows the encoder to contribute continuous pure second derivatives through $\Delta E_i(\mathbf{p})$, while ensuring that J_{E_i} remains continuous. This enables Smooth-NGP to handle the strong form of a second-order PDE.

4 Experiments

To evaluate the representation quality and high-order differentiability of our architecture, we consider three applications: image fitting, neural signed distance field learning for 3D surface reconstruction, and PDE solving with Physics-Informed Neural Networks (PINNs) Raissi et al. (2019).

CUDA implementation. We implement Smooth-NGP in CUDA by extending NVIDIA's tiny-cuda-nn framework Müller (2021), which provides the CUDA implementation underlying Instant-NGP. Specifically, we add CUDA kernels for the proposed smooth cutoff kernel and shifted multi-window hash-grid evaluation, and extend the framework to support second- and higher-order analytical derivatives, allowing the C^∞ continuity and smoothness of the proposed representation to be evaluated directly. All results reported in this paper are obtained with this CUDA implementation. All reported training times are measured on a single 24-GB NVIDIA GeForce RTX 4090 GPU.

4.1 Image fitting

Datasets. We evaluate image fitting on all 100 RGB images from the validation split of the DIV2K dataset Agustsson & Timofte (2017). Each image is resized so that its shorter side is 512 pixels while preserving its aspect ratio, and then center-cropped to 512×512 pixels. Pixel intensities are linearly normalized to $[-1, 1]$, and each image is fitted independently as a continuous 2D signal $I_\theta(\mathbf{p})$, where $\mathbf{p} \in [0, 1]^2$ denotes the normalized image coordinate. We use PSNR and SSIM to evaluate reconstruction quality.

Baselines and experimental setup. The models are trained using full-image pixel supervision with $\mathcal{L}_{\text{pix}} = \frac{1}{3|\Omega_I|} \sum_{\mathbf{p}\in\Omega_I} \|I_\theta(\mathbf{p}) - I_{\text{gt}}(\mathbf{p})\|_2^2$, where Ω_I is the set of image pixels and $I_\theta(\mathbf{p}), I_{\text{gt}}(\mathbf{p}) \in [-1, 1]^3$ are the predicted and ground-truth RGB vectors at pixel $\mathbf{p}$.

We compare Smooth-NGP with Instant-NGP Müller et al. (2022), SIREN Sitzmann et al. (2020), and ∂^∞-Grid Kairanda et al. (2026). Instant-NGP and ∂^∞-Grid use maximum resolutions of 256, whereas Smooth-NGP uses four shifted windows with a maximum per-window resolution of 119. Instant-NGP, ∂^∞-Grid, and Smooth-NGP contain approximately 0.175M, 0.175M, and 0.183M grid parameters, respectively. Complete configurations and parameter-count calculations are provided in Appendices A and A.1.



(a) Instant-NGP PSNR: 31.58 dB (b) SIREN PSNR: 35.26 dB (c) ∂^∞-Grid PSNR: 32.03 dB (d) Ours PSNR: 36.05 dB

Figure 3: Qualitative comparison of image-fitting results using Instant-NGP, SIREN, ∂^∞-Grid, and our method. The main panels show absolute residual maps, while the insets show the corresponding reconstructed images. All residual maps share the color scale shown on the right, and the PSNR of each reconstruction is reported below the corresponding panel.

Table 1: Quantitative comparison for image fitting on DIV2K. PSNR and SSIM measure image quality, while T reports training time in minutes. All three metrics are reported as medians over the 100 validation images.

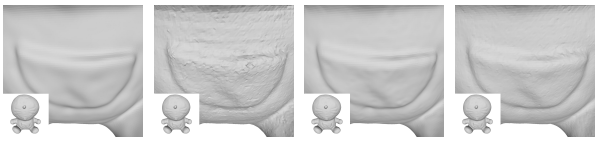
Method	**PSNR** ↑	**SSIM** ↑	T **(min)** ↓
Instant-NGP	30.42	0.93	2.83
SIREN	33.01	0.93	3.58
∂^∞-Grid	29.26	0.91	84.61
Ours	**33.04**	**0.96**	3.58

Results. Figure 3 presents a typical example, indicating that Smooth-NGP yields residuals that are both smaller in magnitude and less prevalent across the image than the tested baselines. Owing to the smooth cutoff kernel and sliding-window aggregation, Smooth-NGP reconstructs edge transitions more faithfully, resulting in lower residuals in these areas than Instant-NGP and ∂^∞-Grid. Because image edges involve large, rapidly changing spatial gradients, they are particularly difficult for grid interpolation; the observed reduction in residuals aligns with a better modeling of spatial derivatives in high-gradient regions. Relative to SIREN, the local nature of our grid-based representation further reduces error in regions with smooth colors. Summary results on the complete DIV2K validation set are provided in Table 1. Additional qualitative results for a subset of images are discussed in Appendix B.

4.2 Neural Signed Distance Fields

Datasets. We further evaluate our method on high-fidelity 3D surface reconstruction to assess the gradient smoothness of the learned representations. We evaluate 72 objects from OmniObject3D Wu et al. (2023), comprising 24 randomly selected categories and three randomly selected models per category. Each object is normalized to the coordinate range $[-1, 1]^3$. We use the symmetric Chamfer-L_1 Distance (CD) and Normal Error (NE) following the Occupancy Networks evaluation protocol Mescheder et al. (2019) to quantify geometric fidelity and normal accuracy.

Baselines and experimental setup. The networks learn an implicit continuous 3D surface via a Signed Distance Field (SDF) and minimize $\mathcal{L} = \mathbb{E}_{\mathbf{p}} \left[(u_\theta(\mathbf{p}) - \text{SDF}_{\text{gt}}(\mathbf{p}))^2\right]$, where $\mathbf{p} \in [0, 1]^3$ is a normalized 3D coordinate, $u_\theta(\mathbf{p})$ is predicted by the hash-grid encoder and MLP decoder, and $\text{SDF}_{\text{gt}}(\mathbf{p})$ is the ground truth. After training, we apply the Marching Cubes algorithm Lorensen & Cline (1987) to the zero level set of u_θ at a grid resolution of 256^3 to extract the surface mesh. We compare Smooth-NGP with SIREN Sitzmann et al. (2020), Instant-NGP Müller et al. (2022), and ∂^∞-Grid Kairanda et al. (2026). Instant-NGP and ∂^∞-Grid use the same eight-level resolution schedule with a maximum resolution of 128, whereas Smooth-NGP uses four shifted windows with a maximum per-window resolution of 64. Instant-NGP, ∂^∞-Grid, and Smooth-NGP contain approximately 7.14M, 7.14M, and 3.53M grid parameters, respectively. Complete configurations and parameter-count calculations are provided in Appendices A and A.1.



(a) SIREN (b) Instant-NGP (c) ∂^∞-Grid (d) Ours

Figure 4: Qualitative comparison of 3D reconstruction results obtained by SIREN, Instant-NGP, ∂^∞-Grid, and our method. The main panels show magnified surface details, while the insets show the corresponding full reconstructions.

Table 2: Median quantitative metrics over all 72 evaluated objects. CD and NE are scaled by 10^{-3} and 10^{-2}, respectively, and T denotes the total runtime in minutes.

Method	**CD** $\downarrow$	**NE** $\downarrow$	T **(min)** $\downarrow$
SIREN	2.87	2.73	26.9
Instant-NGP	2.70	1.34	**20.4**
∂^∞-Grid	2.68	1.12	144.2
Ours	**2.67**	**0.90**	35.6

Results. As shown in Figure 4 and Table 2, we evaluate our method on 3D surface reconstruction by extracting zero-level set meshes from learned SDFs. Qualitatively, Smooth-NGP suppresses the surface noise visible in Instant-NGP while retaining more geometric detail than SIREN, which tends to over-smooth fine structures. Compared with Instant-NGP Müller et al. (2022), our method substantially reduces NE from 1.338 to 0.898. Since the normal to an SDF level set is determined by the direction of its spatial gradient, this reduction is consistent with a more accurate representation of spatial derivatives. Furthermore, the strong locality of our grid-based representation preserves high-frequency details and yields a lower CD than SIREN. The improvement over Instant-NGP entails additional computation: Smooth-NGP takes 35.6 minutes vs. 20.4 minutes for Instant-NGP, reflecting the overhead of smooth interpolation and four shifted windows. Compared with ∂^∞-Grid, Smooth-NGP achieves slightly better reconstruction quality while reducing the runtime from 144.2 to 35.6 minutes. This substantial speed advantage is primarily attributable to our CUDA implementation and to the extensive preprocessing required by ∂^∞-Grid to construct interpolation-neighborhood indices. Additional visual results are discussed in Appendix B.

4.3 Solving PDEs via PINN

In this section, we perform experiments in the PINN framework Raissi et al. (2019) to assess the accuracy of our proposed architecture for adapting higher-order derivatives.

Datasets. We adopt the PINNacle benchmark Hao et al. (2024b) and, in light of the well-known convergence difficulties of PINNs, we remove equations for which the default PINNacle baseline yields relative errors greater than 0.5 or collapses to the trivial zero solution, since PINNs fail to converge in these settings. As a result, we assess six instances covering four types of PDEs: Burgers1D (1D Burgers, nonlinear advection diffusion), Poisson2D-Classic (2D Poisson, elliptic source-driven) and PoissonBoltzmann2D (2D Poisson-Boltzmann, electrostatic potential in ionic media), NS2D-LidDriven and NS2D-BackStep (2D incompressible Navier-Stokes in cavity and back-step flows), and Wave1D (1D wave propagation).

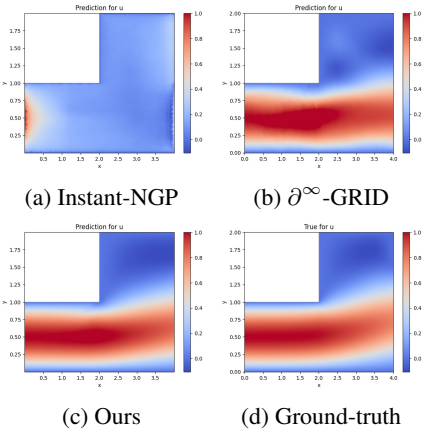
Baselines and experimental setup. We compare Smooth-NGP with Instant-NGP Müller et al. (2022) and ∂^∞-GRID Kairanda et al. (2026). Instant-NGP and ∂^∞-GRID both use a maximum resolution of 256, whereas Smooth-NGP uses four shifted windows with a maximum per-window resolution of 64. Instant-NGP, ∂^∞-GRID, and Smooth-NGP contain approximately 0.175M, 0.197M, and 0.073M grid parameters, respectively. Complete configurations and parameter-count calculations are provided in Appendices A and A.1.

Following the benchmark settings established in PINNacle Hao et al. (2024b), the overall loss function is generally formulated as $\mathcal{L} = \mathcal{L}_{pde} + \lambda\mathcal{L}_{bc} + \lambda\mathcal{L}_{ic}$, where $\mathcal{L}_{pde}$, $\mathcal{L}_{bc}$, and $\mathcal{L}_{ic}$ denote the loss terms for the governing PDE residuals, boundary conditions, and initial conditions, respectively. The specific PDEs and detailed loss formulations for each experiment follow the settings established in PINNacle.

Results. As shown in Figure 5 and Table 3, Smooth-NGP achieves the lowest relative error among the evaluated grid-based methods across all six cases. In the NS-BS example, Instant-NGP produces a substantially less accurate flow field, while ∂^∞-GRID yields a broadly converged solution but retains

Table 3: Quantitative comparison and ablation study on the PINNacle benchmark. We report the l_2 relative error $\left(\frac{\|u_\theta - u^*\|_2}{\|u^*\|_2}\right)$ against the ground-truth field u^*; lower values are better, and the best results are highlighted in bold. The upper block compares alternative architectures, while the lower block reports ablations followed by the complete model. Abbreviations: SW (Sliding Window), Burg-1D (Burgers 1D), Pois-2DC (Poisson 2D_Classic), Pois-B2D (Poisson Boltzmann2D), NS-LD (Navier-Stokes LidDriven), NS-BS (Navier-Stokes BackStep), and Wave-1D (Wave 1D).

Method	**Burg-1D**	**Pois-2DC**	**Pois-B2D**	**NS-LD**	**NS-BS**	**Wave-1D**
Instant-NGP	2.86e-1	4.23e-1	1.01e0	8.09e-1	7.67e-1	8.92e-1
∂^∞-GRID	1.04e-1	8.64e-2	3.07e-2	1.07e0	2.68e-1	7.90e-1
Linear + SW	2.09e-1	3.29e-1	9.29e-1	8.71e-1	2.64e-1	9.26e-1
Ours w/o SW	1.97e-1	4.29e-1	7.32e-2	5.42e-1	3.61e-1	8.87e-1
Ours	**9.53e-2**	**1.06e-3**	**1.37e-3**	**1.93e-1**	**1.31e-1**	**6.10e-2**



(a) Instant-NGP (b) ∂^∞-GRID

(c) Ours (d) Ground-truth

Figure 5: Qualitative comparison of the fitted horizontal velocity field for the NS-BS backward-step flow. Results from Instant-NGP, ∂^∞-GRID, Smooth-NGP, and the FDM reference are shown.

noticeable numerical error; Smooth-NGP more closely recovers the principal flow structure shown in the FDM reference. These lower solution errors, particularly on PDEs involving second-order operators, are consistent with a more accurate representation of higher-order spatial derivatives while retaining an efficient hash-grid architecture. Within each cell, the multilinear encoder Hessian has a zero diagonal, whereas its off-diagonal entries are not forced to vanish; although the nonlinear decoder can supply pure second derivatives, this constraint can increase error relative to the target Hessian. These results suggest that smooth hash-grid representations can be effectively applied to higher-order PDEs, such as the Poisson equation, within the PINN framework. Additional visual results are discussed in Appendix B.

4.4 Ablation Studies

To disentangle the roles of the smooth kernel and the sliding-window architecture, the lower block of Table 3 reports two ablated configurations. Linear + SW retains the shifted-window architecture but replaces the smooth kernel with multilinear interpolation. Its encoder Hessian has a zero diagonal within each cell, whereas its mixed derivatives are not forced to vanish; it therefore contributes no pure second-order terms to the Laplacian in Equation (7). Its consistently higher errors demonstrate that shifted windows alone cannot provide the smooth pure second derivatives required by the evaluated PDE objectives. Conversely, Ours w/o SW retains the smooth interpolation kernel but removes the multi-window aggregation in Equation (6). Although its encoding remains smooth, the boundary-flat kernel produces vanishing derivatives at cell boundaries, weakening optimization. Its higher errors therefore confirm that shifted windows are necessary to compensate for this boundary degeneration. Together, these results demonstrate the complementary roles of the two components: the smooth kernel provides higher-order continuity, while the shifted windows maintain informative gradients near cell boundaries.

5 Conclusions

We present Smooth-NGP, combining a classical boundary-flat C^∞ kernel and sliding windows in multiresolution hash encodings. Instant-NGP's multilinear encoder Hessian has a zero diagonal within cells, whereas its off-diagonal entries are not forced to vanish. Smooth-NGP provides continuous arbitrary-order derivatives, improves cross-cell gradient smoothness, and resolves boundary-induced gradient vanishing. Image and SDF results indicate improved gradient fitting; PINN results validate smooth second derivatives. Limitations include higher peak memory and slower execution than Instant-NGP due to independent windows and derivative computation. Future work targets fused CUDA kernels and adaptive windows.

AI USE STATEMENT

In this work, we used OpenAI's Codex to assist with manuscript preparation and revision, including translation, organization, improvement of clarity and wording, refinement of mathematical exposition, and preparation and layout of figures and tables. The tool was also used to help summarize and articulate observations from experimental results provided by the authors. We did not use generative AI tools to generate or preprocess experimental data, implement the core method, conduct experiments, or produce any of the reported numerical results; the remaining required-disclosure tasks are not applicable to this work.

All AI-assisted material was critically reviewed, verified, and revised by the authors where necessary. In particular, the authors independently checked all mathematical statements, experimental values, citations, figures, and conclusions. We take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work develops a smooth multiresolution hash-grid representation for continuous signal fitting, three-dimensional surface reconstruction, and physics-informed PDE solving. The experiments use established public benchmarks and do not involve human participants, personal or sensitive data, or automated decisions concerning individuals. We are not aware of ethical risks specific to the proposed method beyond those generally associated with neural-field modeling, three-dimensional reconstruction, and scientific machine learning.

REPRODUCIBILITY STATEMENT

Section 3 specifies the multiresolution hash encoding, the adopted C^∞ cutoff interpolation function, the shifted-window aggregation scheme, the smooth decoder, and the analytical derivative formulation. Sections 4.1, 4.2, and 4.3 describe the datasets, training objectives, compared methods, and evaluation protocols for image fitting, signed distance field reconstruction, and PDE solving, respectively. Appendices A and A.1 provide the complete experimental configurations and detailed grid-parameter calculations. We will release the source code and experiment scripts upon publication.

REFERENCES

Eirikur Agustsson and Radu Timofte. NTIRE 2017 challenge on single image super-resolution: Dataset and study. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1122–1131, 2017. doi: 10.1109/CVPRW.2017.150. URL https://doi.org/10.1109/CVPRW.2017.150.

Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Oscar P. Bruno and Leonid A. Kunyansky. A fast, high-order algorithm for the solution of surface scattering problems: Basic implementation, tests, and applications. *Journal of Computational Physics*, 169(1):80–110, 2001. ISSN 0021-9991. doi: https://doi.org/10.1006/jcph.2001.6714.

Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022.

Arka Daw, Jie Bu, Sifan Wang, Paris Perdikaris, and Anuj Karpatne. Mitigating propagation failures in physics-informed neural networks using retain-resample-release (r3) sampling. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Wenrui Hao, Xinliang Liu, and Yahong Yang. Newton informed neural operator for solving nonlinear partial differential equations. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, 2024a.

Zhongkai Hao, Jiachen Yao, Chang Su, Hang Su, Ziao Wang, Fanzhi Lu, Zeyu Xia, Yichi Zhang, Songming Liu, Lu Lu, and Jun Zhu. Pinnacle: A comprehensive benchmark of physics-informed neural networks for solving pdes. In *Advances in Neural Information Processing Systems*, 2024b.

Wenbo Hu, Yuling Wang, Lin Ma, Bangbang Yang, Lin Gao, Xiao Liu, and Yuewen Ma. Tri-miprf: Tri-mip representation for efficient anti-aliasing neural radiance fields. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Navami Kairanda, Shanthika Naik, Marc Habermann, Avinash Sharma, Christian Theobalt, and Vladislav Golyanik. ∂^∞-grid: A neural differential equation solver with differentiable feature grids. *International Conference on Learning Representations*, 2026.

Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael Mahoney. Characterizing possible failure modes in physics-informed neural networks. In *Advances in Neural Information Processing Systems*, 2021.

Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proc. of ACM SIGGRAPH*, pp. 163–169, 1987.

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

Thomas Müller. tiny-cuda-nn. `https://github.com/NVlabs/tiny-cuda-nn`, 2021.

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

Joseph Ortiz, Alexander Clegg, Jing Dong, Edgar Sucar, David Novotny, Michael Zollhoefer, and Mustafa Mukadam. isdf: Real-time neural signed distance fields for robot perception. In *Robotics: Science and Systems*, 2022.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Radu Alexandru Rosu and Sven Behnke. Permutosdf: Fast multi-view reconstruction with implicit surfaces using permutohedral lattices. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Sara Fridovich-Keil and Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems*, Red Hook, NY, USA, 2020. Curran Associates Inc.

Sifan Wang, Shyam Sankaran, and Paris Perdikaris. Respecting causality for training physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 2024.

Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse pde problems. In *Advances in Neural Information Processing Systems*, 2024.

Yiming Wang, Qin Han, Marc Habermann, Kostas Daniilidis, Christian Theobalt, and Lingjie Liu. Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Haixu Wu, Huakun Luo, Yuezhou Ma, Jianmin Wang, and Mingsheng Long. Ropinn: Region optimized physics-informed neural networks. In *Advances in Neural Information Processing Systems*, 2024.

Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, Dahua Lin, and Ziwei Liu. OmniObject3D: Large-vocabulary 3D object dataset for realistic perception, reconstruction and generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 803–814, 2023. doi: 10.1109/CVPR52729.2023.00084.

Lexing Ying and Denis Zorin. A simple manifold-based construction of surfaces of arbitrary smoothness. *ACM SIGGRAPH 2004 Papers*, 2004.

Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. PlenOctrees for real-time rendering of neural radiance fields. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

A Experimental Configurations

Image fitting. Instant-NGP uses eight levels, two features per level, a base resolution of 2, a growth factor of 2, and a maximum resolution of 256. Smooth-NGP uses eight levels, two features per level, a base resolution of 5, a growth factor of 1.6223, four shifted windows, and a maximum per-window resolution of 119. ∂^∞-Grid uses the same eight-level resolution schedule as Instant-NGP, with two features per scale and a maximum resolution of 256. All three grid-based models therefore produce 16-dimensional encodings. Each encoding is mapped through one 64-dimensional $\tanh$ hidden layer to a linear RGB output layer. All models are optimized with Adam for 10,000 full-image updates using cosine learning-rate annealing from an initial learning rate of 10^{-3}.

Neural signed distance fields. Smooth-NGP uses eight levels, two features per level, a base resolution of 8, a growth factor of 1.34, four shifted windows, and a maximum per-window resolution of 64. Instant-NGP uses eight levels, two features per level, a base resolution of 17, a growth factor of 1.33, and a maximum resolution of 128. ∂^∞-Grid uses the same grid configuration as Instant-NGP, including eight levels and a maximum resolution of 128. The three grid-based configurations use an identical three-layer MLP decoder with width 64 and Softplus ($\beta = 100$) activations. As in image fitting, we optimize all methods with Adam using an initial learning rate of 10^{-3} and cosine learning-rate annealing. For mesh evaluation, we sample 200,000 points uniformly from the surfaces of both the predicted and reference meshes. Following Occupancy Networks Mescheder et al. (2019), CD is the symmetric Chamfer-L_1 distance, and NE is one minus the symmetric normal consistency computed from nearest-neighbor correspondences. We report the median of each metric across all 72 evaluated objects.

PINN. Instant-NGP uses eight levels, two features per level, a base resolution of 2, a growth factor of 2, and a maximum resolution of 256. ∂^∞-GRID uses its official single-level configuration with resolution 256 and three features. In preliminary experiments, this official 1×3 configuration outperformed an 8×2 multilevel alternative; we therefore use the official configuration for all reported ∂^∞-GRID results. Smooth-NGP uses eight levels, two features per level, a base resolution of 8, a growth factor of 1.343, four shifted windows, and a maximum per-window resolution of 64; this configuration has fewer trainable parameters than Instant-NGP. All three methods use two width-64 linear layers separated by a Tanh activation. We use NTK-based loss-weight adaptation and cosine annealing to 0.01 times the initial learning rate of 5×10^{-3}.

A.1 Grid Parameter Counts

We count only the trainable grid features, excluding the decoder parameters shared by the compared grid-based models. Let W denote the number of independently parameterized windows, F the feature dimension at each grid point, r_ℓ the per-axis resolution at level ℓ, and d the input dimension. Because the hash-table capacity exceeds the number of grid points at every level in these configurations, and because each resolution level has independent parameters, the grid parameter count is

$$P_{\text{grid}} = WF \sum_{\ell=0}^{L-1} r_\ell^d. \tag{8}$$

Image fitting. Instant-NGP uses $r_\ell \in [2, 4, 8, 16, 32, 64, 128, 256]$, and therefore

$$P_{\text{INGP}} = 2 \sum_{\ell} r_\ell^2 = 174{,}760.$$

For Smooth-NGP, the base resolution of 5 grid points corresponds to 4 cells, and the growth factor is applied to the cell resolution. The resulting grid-point resolutions are $[5, 7, 11, 18, 28, 45, 73, 119]$. Thus,

$$P_{\text{Smooth}} = 4 \times 2 \times (5^2 + 7^2 + 11^2 + 18^2 + 28^2 + 45^2 + 73^2 + 119^2) = 182{,}544.$$

With two features at each independently parameterized resolution level, ∂^∞-Grid has

$$P_{\partial^\infty} = 2 \sum_{r \in [2,4,8,16,32,64,128,256]} r^2 = 174{,}760.$$

Neural signed distance fields. The Instant-NGP resolutions are $[17, 22, 30, 40, 53, 71, 95, 128]$, whereas the per-window Smooth-NGP resolutions are $[8, 11, 15, 20, 27, 35, 47, 64]$. Hence,

$$P_{\mathrm{INGP}} = 2 \times (17^3 + 22^3 + 30^3 + 40^3 + 53^3 + 71^3 + 95^3 + 128^3) = 7{,}135{,}752,$$
$$P_{\mathrm{Smooth}} = 4 \times 2 \times (8^3 + 11^3 + 15^3 + 20^3 + 27^3 + 35^3 + 47^3 + 64^3) = 3{,}533{,}944.$$

Because ∂^∞-Grid uses the same independently parameterized resolution levels as Instant-NGP,

$$P_{\partial^\infty} = 2 \times (17^3 + 22^3 + 30^3 + 40^3 + 53^3 + 71^3 + 95^3 + 128^3) = 7{,}135{,}752.$$

PINN. All six evaluated PINN problems have two-dimensional input domains. Instant-NGP again uses resolutions $[2, 4, 8, 16, 32, 64, 128, 256]$, while the per-window Smooth-NGP resolutions are $[8, 11, 15, 20, 27, 35, 47, 64]$. Their parameter counts are

$$P_{\mathrm{INGP}} = 2 \sum_{r \in [2,4,8,16,32,64,128,256]} r^2 = 174{,}760,$$
$$P_{\mathrm{Smooth}} = 4 \times 2 \sum_{r \in [8,11,15,20,27,35,47,64]} r^2 = 72{,}552,$$
$$P_{\partial^\infty} = 3 \times 256^2 = 196{,}608.$$

B Extended Qualitative Results

Image-fitting results. Figure 6 provides additional qualitative comparisons on eight randomly selected images from the DIV2K validation set, complementing the aggregate results in Table 1. Following Figure 3, each main panel shows the absolute-error map, with the corresponding reconstruction shown in the lower-left inset. Across images with different content and texture complexity, Instant-NGP and ∂^∞-Grid retain structured residuals around object contours and fine-scale patterns. Smooth-NGP generally reduces the extent and magnitude of these residuals relative to both grid-based baselines, particularly near rapid color transitions.

This behavior is consistent with the complementary roles of the two proposed components: the smooth cutoff kernel removes the gradient discontinuities of multilinear interpolation, while the shifted windows eliminate the derivative vanishing induced by boundary-flat interpolation. Their combination, therefore, improves the representation of high-gradient image regions without discarding the locality of the grid encoding. SIREN remains competitive and can produce lower residuals on individual images, whereas Smooth-NGP preserves localized detail while achieving comparable overall reconstruction quality. Together with the aggregate PSNR and SSIM results in Table 1, these examples indicate that the improvement over the grid-based baselines is not confined to the single case shown in the main paper.

SDF Results. As comprehensively illustrated in Figure 8 and Figure 9, we evaluate the visual quality of the reconstructed meshes on a selection of objects from OmniObject3D Wu et al. (2023). Due to the inherent gradient discontinuities at cell boundaries, the standard Instant-NGP baseline suffers from a loss of overall surface smoothness, occasionally introducing visible grid-like artifacts on certain extracted meshes. In contrast, our proposed method substantially reduces these grid artifacts, yielding cleaner and geometrically smoother reconstructions. On the other hand, while pure-MLP architectures like SIREN generate smooth surfaces, they frequently suffer from an over-smoothing phenomenon that washes out fine geometric structures. Benefiting from the strong locality of the grid-based architecture, our approach successfully preserves high-frequency local details while maintaining high-order continuity, thereby achieving superior overall reconstruction quality compared to both baselines.

It is worth noting that while some existing studies mitigate SIREN's over-smoothing or Instant-NGP's gradient discontinuities by introducing auxiliary loss terms, we deliberately exclude such regularizations. To ensure a strictly fair comparison and explicitly demonstrate the intrinsic architectural improvements of our method, all models are trained using only the most fundamental loss function: $\mathcal{L} = \mathbb{E}_{\mathbf{p}} \left[(u_\theta(\mathbf{p}) - \mathrm{SDF}_{\mathrm{gt}}(\mathbf{p}))^2 \right]$.

PINN Results. Figure 7 presents a comprehensive qualitative comparison of the predicted fields across the six evaluated PDE benchmarks. As observed, the standard Instant-NGP struggles to accurately resolve complex physical fields. Within each cell, its multilinear encoder Hessian has a zero diagonal, whereas its off-diagonal entries are not forced to vanish; although the decoder can supply pure second derivatives, this constraint can increase error relative to the target Hessian. Its first derivatives are also discontinuous across cell boundaries. These limitations can contribute to large approximation errors and visible artifacts, particularly in higher-order problems such as the Poisson and Navier-Stokes equations.

Although the ∂^∞-GRID method manages to yield converged solutions, its visualizations reveal pronounced numerical errors and localized artifacts, particularly when solving complex PDE problems. This indicates limitations in maintaining high-order smoothness under challenging conditions.

Our proposed method significantly enhances the expressive capacity of grid-based representations. By utilizing a high-order smooth interpolation function, our approach leverages its grid structure to capture fine-grained local details of the physical fields. A visual comparison between our full model and the ablation study (Ours w/o SW) illustrates the role of the sliding-window mechanism. Without this mechanism, the representation still suffers from localized structural artifacts and checkerboard-like patterns. The smooth interpolation provides high-order continuity, while the sliding windows improve optimization near cell boundaries and suppress these artifacts. Among the evaluated grid-based methods, our method yields the lowest reported error across all six cases.

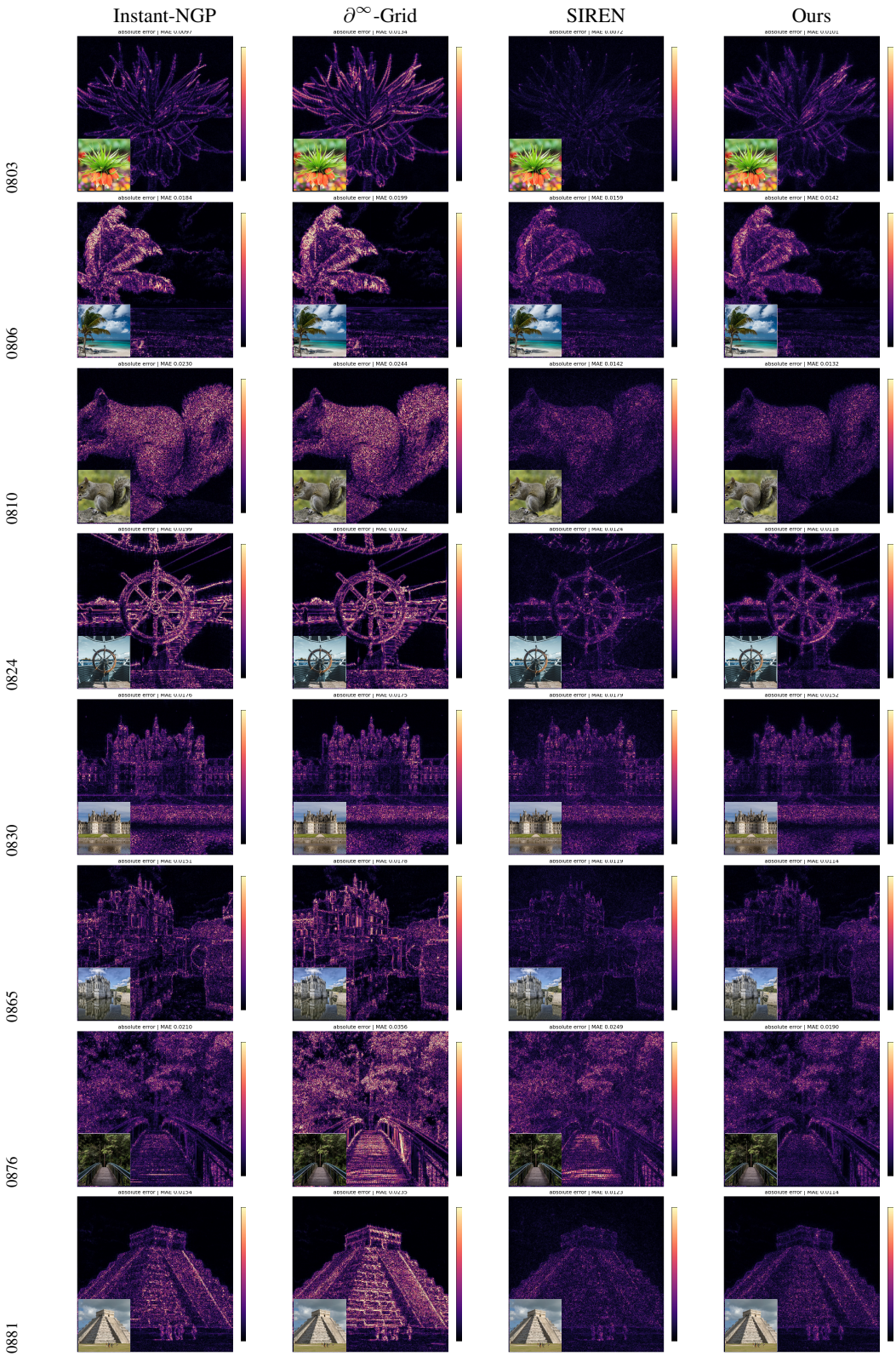


Figure 6: Additional qualitative image-fitting results on eight randomly selected DIV2K validation images. The rows are identified by DIV2K image index. All absolute-error maps use the same color scale as Figure 3, and the corresponding reconstructions are shown in the lower-left insets.

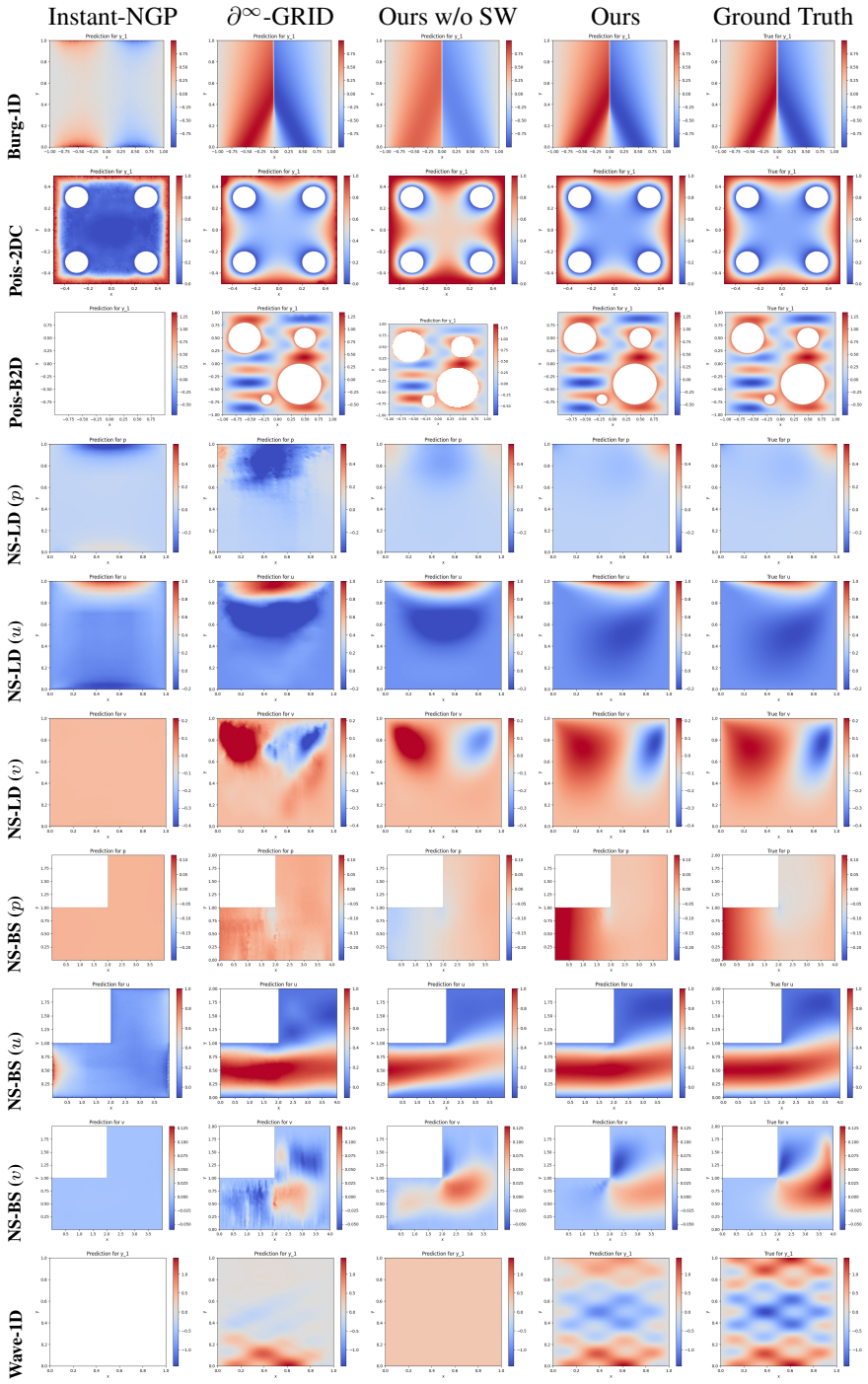


Figure 7: Qualitative comparison of different architectures within the PINN framework across six PDE benchmarks. For the Navier-Stokes experiments (NS-LD and NS-BS), the pressure (p) and velocity components (u, v) are displayed in separate rows. Smooth-NGP suppresses grid-based artifacts and yields reconstructions closer to the ground-truth fields than the evaluated grid-based baselines.

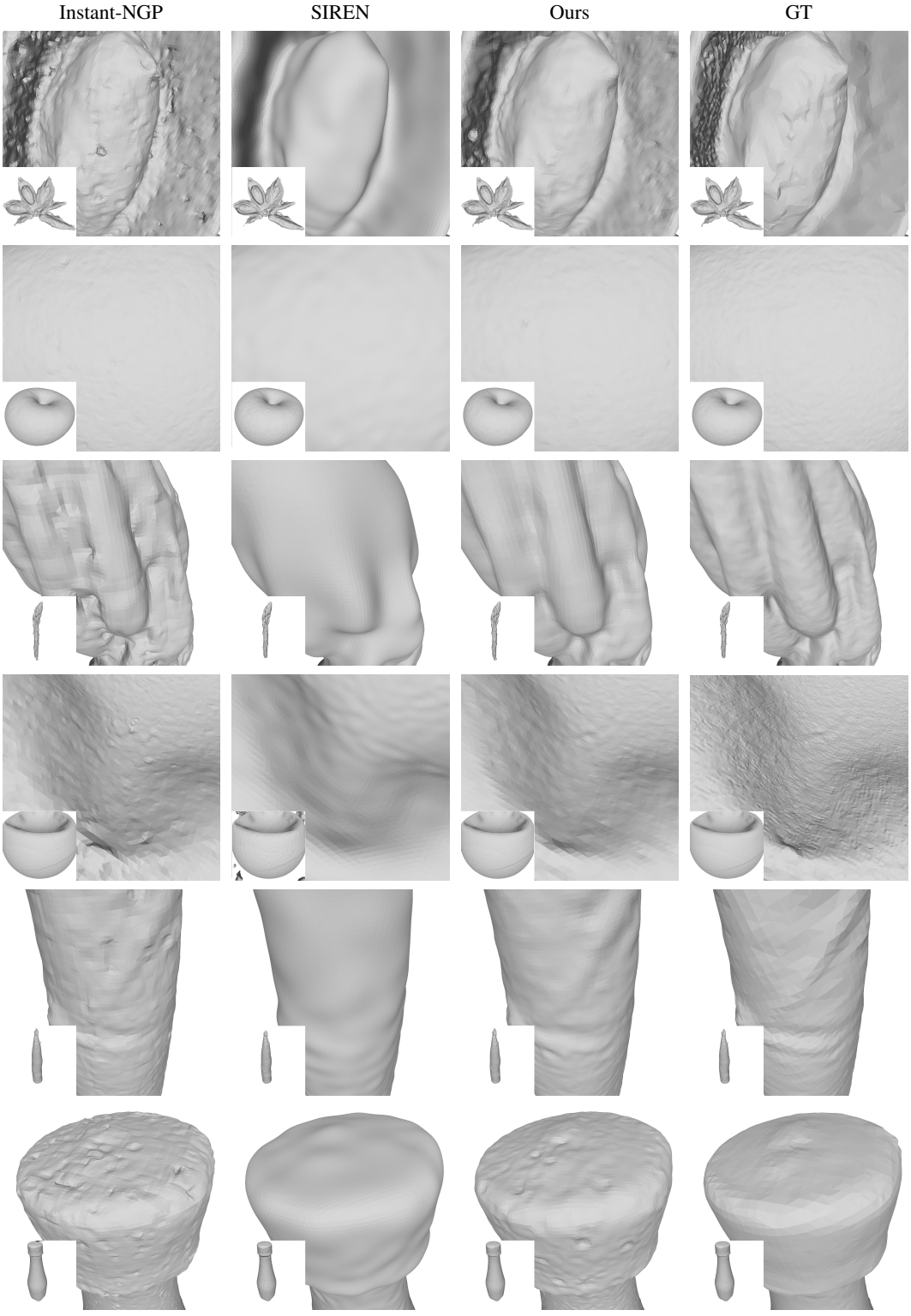


Figure 8: Extensive qualitative comparison across different scenes (Part 1/2). For each scene, we provide a zoomed-in detail view (background) and the full overall view (bottom-left inset). SIREN tends to produce over-smoothed surfaces, while Instant-NGP exhibits noticeable bumpy artifacts. In contrast, our method yields smoother surfaces while preserving high-fidelity geometric details.

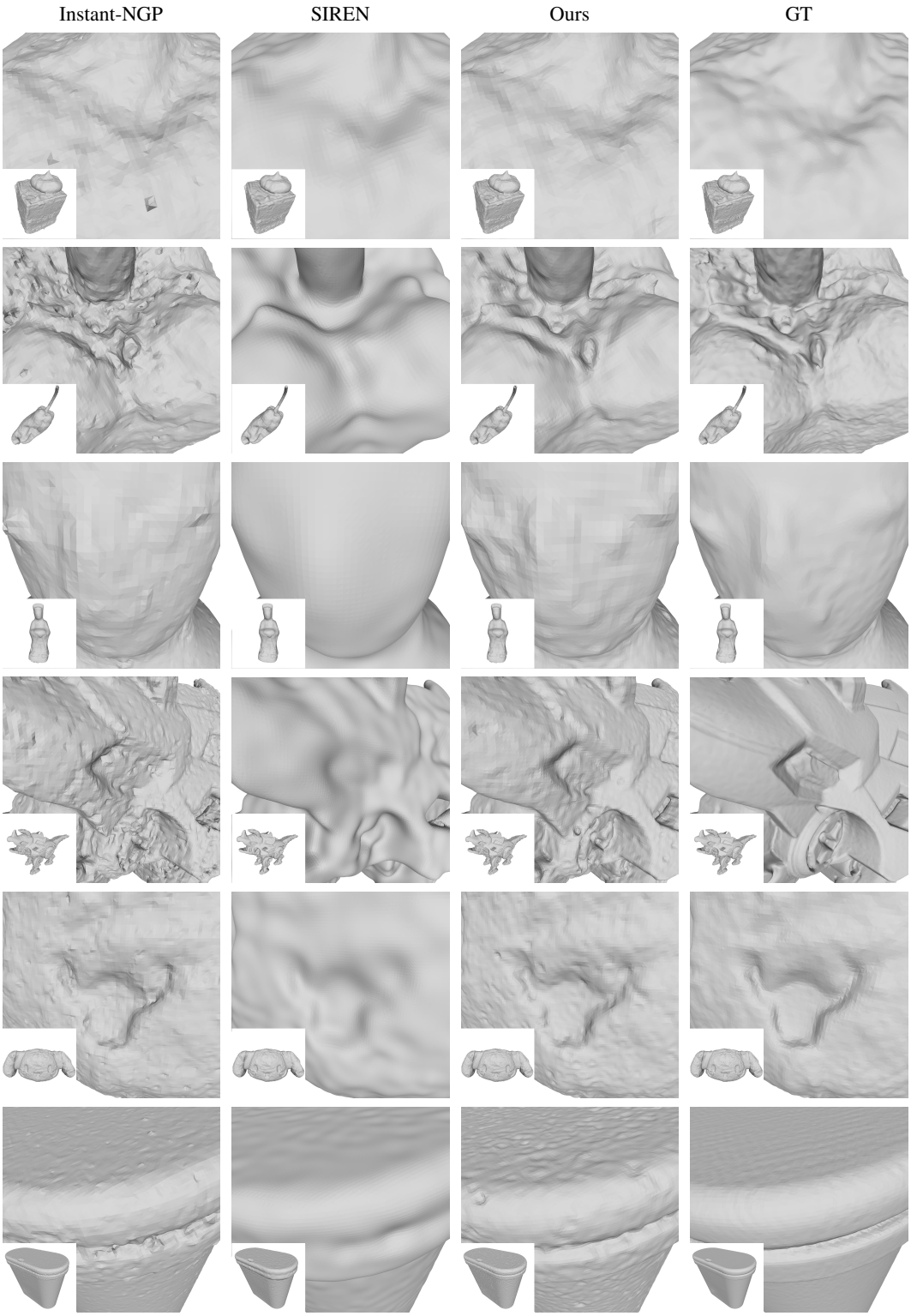


Figure 9: Extensive qualitative comparison across different scenes (Part 2/2). Continued from Figure 8.