FLEXICOORD: FLEXIBLE SPATIAL REPRESENTATIONS VIA FUNCTION REPARAMETERIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Spatial Representations form the foundation of various modern Machine Learning systems in Computer Vision and Graphics, such as image encoding, data compression, and 3D reconstruction. However, these representations typically evaluate spatial functions in fixed coordinate systems, limiting their efficiency in modeling sharp boundaries, thin structures, and complex geometry. We introduce FlexiCoord, a general strategy that augments an existing spatial function with a learned coordinate reparameterization field, allowing its spatial support to adapt to the target signal. We instantiate FlexiCoord with Gaussian kernels for novel-view synthesis, multi-resolution hash grids for image encoding, and vector–matrix decompositions for volumetric medical-data compression. At fixed primitive budgets, FlexiCoord-GS improves reconstruction fidelity while maintaining rendering speeds comparable to 2DGS and approximately three times faster than texture-based Gaussian methods. Under comparable memory budgets, FlexiCoord-NGP and FlexiCoord-VM improve the reconstruction of images and medical volumes, respectively. These results demonstrate that learned coordinate reparameterization provides a broadly applicable mechanism for improving the quality–efficiency trade-off of spatial representations.

1 INTRODUCTION

Spatial representations are a core component of modern machine learning systems for computer vision and computer graphics. A wide range of tasks, including image encoding (Müller et al., 2022), 3D reconstruction (Mildenhall et al., 2020; Barron et al., 2021; 2022; Kerbl et al., 2023; Huang et al., 2024; Chen et al., 2022), and volumetric data compression (Chen et al., 2022), can be formulated as the approximation of a spatially varying signal. Depending on the application, this signal may be encoded using feature grids (Müller et al., 2022), Gaussian primitives (Kerbl et al., 2023; Huang et al., 2024), tensor decompositions (Chen et al., 2022), or other local basis functions. Despite their architectural differences, these spatial representations often share a common design principle: query points are evaluated in a fixed, uniform coordinate system. This fixed-coordinate assumption can become a bottleneck for representation efficiency. Natural spatial signals frequently contain sharp edges, thin structures, curved boundaries, and locally complex geometric patterns. When a representation is constrained to operate in a fixed coordinate system, simple basis functions must either become increasingly dense or increasingly expressive to capture these structures. For example, Gaussian splatting may require many primitives to approximate non-elliptical shapes; feature grids may require high resolution to preserve sharp image details; and tensor-factorized volume representations may need larger ranks to model complex anatomical structures. These solutions can improve expressiveness, but often introduce substantial memory or computational overhead while producing only marginal improvements in performance.

Recent work on frequency-aligned textured Gaussian (Xie et al., 2026) suggests that adaptive texture reparameterization can improve the representation efficiency of TexturedGS. However, this design is specialized to textured Gaussian splatting. In this work, we ask whether the underlying idea can be generalized to different spatial representations through local coordinate alignment. We introduce FlexiCoord, a general framework for enhancing spatial representations through learned function reparameterization. FlexiCoord augments an existing spatial representation with a lightweight reparameterization field that remaps query locations before evaluation. This allows the representation to

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.

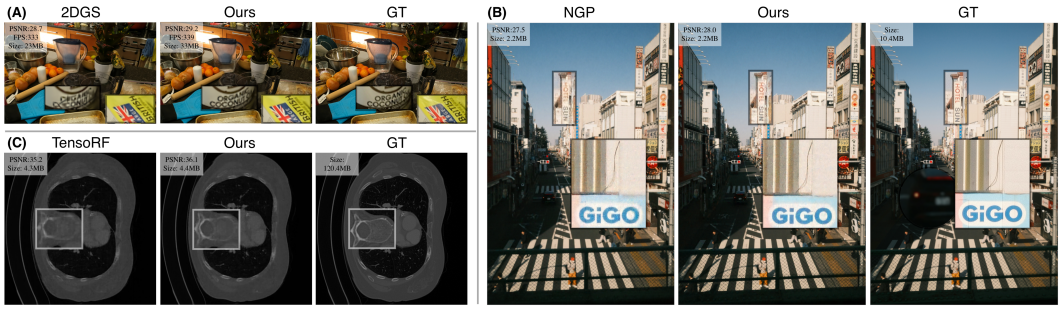

Figure 1: FlexiCoord applies learned coordinate reparameterization to three spatial representations: (A) Gaussian kernels for splatting-based novel-view synthesis, (B) multi-resolution hash grids for image encoding, and (C) vector–matrix tensor decompositions for volumetric medical-data compression. Across these applications, FlexiCoord improves the reconstruction of fine details and thin structures under comparable computational efficiency (A) or representation budgets (B and C).

adapt its local coordinates to the underlying signal, transforming fixed spatial functions into adaptive primitives that can better capture complex geometry.

The key advantage of FlexiCoord is its generality. It is not designed for a specific network architecture, primitive type, or signal modality. Instead, it acts as a representation-level enhancement that can be attached to different spatial functions. In Gaussian Splatting, FlexiCoord deforms the geometry of Gaussian kernels, enabling each primitive to model more flexible spatial support. In image encoding, it augments multi-resolution hash-grid features with coordinate remapping, improving the reconstruction quality of sharp local structures. In volumetric medical data compression, it enhances vector-matrix tensor decompositions by allowing factorized components to adapt to complex anatomical geometry. These applications cover distinct representation families, yet all can be understood through the same principle: spatial functions become more expressive when evaluated in learned, flexible coordinates. These results suggest that FlexiCoord offers a novel direction to design more powerful spatial representations other than simply increasing resolution or expanding parameter budgets.

In summary, our contributions are:

- We propose FlexiCoord, a general framework for improving spatial representations through learned coordinate reparameterization for general functions.
- We instantiate FlexiCoord in three different applications: deformable Gaussian kernels for splatting-based novel-view synthesis, coordinate-remapped hash grids for image encoding, and reparameterized vector-matrix decompositions for volumetric data compression.
- We demonstrate favorable quality–efficiency trade-offs across the three applications: improved fidelity at fixed primitive counts and comparable rendering speed for Gaussian splatting, and improved reconstruction quality under matched memory budgets for image and volumetric encoding.

2 Related Work

2.1 Novel View Synthesis (NVS)

Novel view synthesis (NVS) seeks to generate photorealistic renderings of a 3D scene from unseen viewpoints, typically from a set of posed multi-view images. Early progress in this direction was closely tied to classical structure-from-motion (SfM) and multi-view stereo (MVS) pipelines (Snavely et al., 2006; 2008; Agarwal et al., 2011; Schonberger & Frahm, 2016; Seitz et al., 2006; Furukawa & Ponce, 2009; Furukawa et al., 2010), which recover scene geometry through multi-view correspondences, whereas they are sensitive to occlusions and often struggle to extrapolate to viewpoints that are insufficiently covered by the input images.

Neural radiance fields (NeRF) (Mildenhall et al., 2021) have significantly advanced NVS by representing scenes as implicit neural functions optimized with photometric supervision. Follow-up

works have further improved anti-aliasing (Barron et al., 2021), large-scale reconstruction (Tancik et al., 2022; Barron et al., 2022), reflection modeling (Verbin et al., 2022), and acceleration via grid-based encodings (Zhao et al., 2023). Nevertheless, these methods remain constrained by the need for dense ray sampling and costly MLP inference.

These limitations have motivated the development of explicit representations such as 3D Gaussian Splatting (Kerbl et al., 2023), which replaces neural volumetric fields with differentiable Gaussian primitives for real-time rendering. Based on this foundation, subsequent methods improve geometric accuracy and rendering stability through refined kernel formulations and more effective densification strategies (Fu et al., 2024; Lei et al., 2025; Li et al., 2025). To further improve scene geometry (Huang et al., 2024), 2D Gaussian Splatting introduces planar Gaussian primitives for more geometrically accurate and efficient radiance field representation, enabling fine-grained geometric reconstruction while preserving high appearance quality. However, both 3D and 2D Gaussian Splatting model appearance primarily using view-dependent spherical harmonic coefficients, without capturing spatial color variation within each primitive. This limits their expressiveness, particularly for representing fine-grained textures and high-frequency appearance details.

To address this limitation, recent methods introduce spatially varying Gaussian primitives. Some approaches augment each primitive with RGB or RGBA textures to represent spatially varying appearance (Chao et al., 2025; Xu et al., 2024a; Rong et al., 2025; Huang & Gong, 2024; Song et al., 2024; Xie et al., 2026) or with normal textures to enable relighting (Sun et al., 2025). Others employ neural texture fields to decouple geometry from appearance (Xu et al., 2024b; Zhang et al., 2025), achieving high visual fidelity but leading to slower training and rendering because of costly MLP inference.

Texture-based GS provides a favorable trade-off between quality and efficiency, enhancing appearance modeling while preserving real-time rendering performance. In particular, the most recent FACT-GS (Xie et al., 2026) introduced an adaptive texture parameterization for textured Gaussian Splatting using a deformation field. While effective for improving texture capacity allocation, FACT-GS is specifically designed for texture-based Gaussian representations and does not study function reparameterization as a general mechanism for spatial representations.

2.2 Positional Encoding for Neural Spatial Representations

Neural spatial representations model signals as continuous functions of spatial coordinates. Early implicit neural representations directly regress signals from coordinates using MLPs; however, they failed to capture high-frequency details because of the MLP's low-frequency bias (Rahaman et al., 2019). To address this limitation, NeRF (Mildenhall et al., 2020) adopts sinusoidal positional encoding to map 3D positions and viewing directions into a higher-dimensional Fourier feature space, enabling the network to represent fine geometric and appearance details. SIREN (Sitzmann et al., 2020) instead uses periodic activation functions to represent complex signals such as images, shapes, and wavefields. Instant-NGP (Müller et al., 2022) introduces a multi-resolution hash-grid encoding, where coordinates query learnable features at multiple spatial resolutions and a lightweight MLP decodes the final signal. This design enables efficient high-frequency modeling with manageable memory usage compared with dense multi-resolution grids. Hash-grid encodings have since become a common choice for neural fields and image encoding because they provide fast optimization and compact storage. Follow-up works extended the instant-NGP for different applications, such as neural SDF surface reconstruction (Li et al., 2023), and compact Gaussian Splatting field (Lee et al., 2024). Another line of work reduces the memory cost of volumetric representations through tensor factorization. TensoRF (Chen et al., 2022) factorizes radiance fields using low-rank tensor decompositions, including CP and vector-matrix (VM) decompositions, where 3D feature volumes are represented by compact vector and matrix factors. More recent works introduced more factorized representations, such as K-Planes (Fridovich-Keil et al., 2023), HexPlane (Cao & Johnson, 2023), and Tensor4D (Shao et al., 2023), which extend tensor-factorized fields to dynamic and high-dimensional scenes.

Beyond Neural radiance fields, positional encoding has also been widely used for image representation and implicit neural compression, where an image is represented as a continuous function from 2D coordinates to RGB values. Fourier features and periodic activations allow compact networks to fit high-frequency image details (Tancik et al., 2020; Sitzmann et al., 2020). Other works design

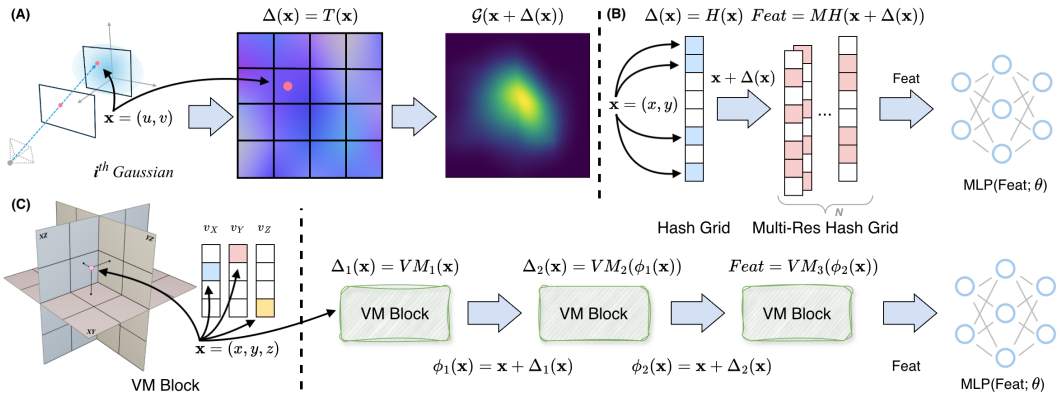

Figure 2: Overview of the three FlexiCoord instantiations. (A) FlexiCoord-GS: a per-primitive displacement texture reparameterizes the local coordinates of each Gaussian kernel. (B) FlexiCoord-NGP: a single-resolution hash grid predicts coordinate offsets used to query the multi-resolution feature grid, whose features are decoded into RGB values by an MLP. (C) FlexiCoord-VM: VM blocks represent both the coordinate-offset fields and the volumetric feature field. Two nested offset fields are used to improve the representation of fine anatomical structures.

band-limited or multiscale coordinate networks to better control aliasing and frequency allocation, such as BACON (Lindell et al., 2022).

These methods demonstrate the importance of coordinate encoding for representing images and other dense signals, but they still rely on fixed coordinate systems and often require careful frequency design to balance smoothness and detail preservation.

3 Method

Existing spatial representations improve expressiveness by increasing the capacity of the function itself, e.g., increasing the number of Gaussian primitives in Gaussian Splatting or increasing the (hash) grid resolution. In contrast, FlexiCoord improves expressiveness by learning a lightweight function reparameterization field that better aligns the local coordinate domain with the target signal.

3.1 FlexiCoord

We define FlexiCoord as a function composition of an existing spatial function $f_\theta(\mathbf{x})$ and a reparameterization field $\Delta_\phi(\mathbf{x})$,

$$\hat{f}_\theta(\mathbf{x}) = f_\theta(\mathbf{x} + \lambda\Delta_\phi(\mathbf{x})), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n,\ n \in \mathbb{R}^+$ is the input spatial coordinate, λ is the reparameterization weight, θ and ϕ are the learnable parameters of $f_\theta(\mathbf{x})$ and $\Delta_\phi(\mathbf{x})$, respectively. When $\Delta_\phi(\mathbf{x}) = \mathbf{0}$, Eq. 1 is exactly equal to the original representation. Therefore, FlexiCoord is a conservative extension that preserves the baseline as a special case.

3.2 FlexiCoord-GS

3.2.1 Deformable Gaussian

Given an input coordinates $\mathbf{x} = (x, y)$, a Gaussian kernel $\mathcal{G}$ in the normalized coordinate space is defined as

$$\mathcal{G}(x, y) = e^{-0.5*(x^2+y^2)}. \tag{2}$$

Applying FlexiCoord by a reparameterization field $\Delta_\phi(\mathbf{x})$ to $\mathcal{G}$ leads to a reparameterized normalized Gaussian $\hat{\mathcal{G}}$, where

$$\hat{\mathcal{G}}(\mathbf{x}) = \mathcal{G}(\mathbf{x} + \lambda\Delta_\phi(\mathbf{x})). \tag{3}$$

This reparameterization deforms the geometry of the normalized Gaussian kernel $\mathcal{G}$, allowing the resulting kernel $\hat{\mathcal{G}}$ to represent more complex spatial structures, as shown in 6.

Such improved expressiveness is particularly useful for Gaussian-splatting-based novel-view synthesis, where reconstructing fine-grained details, such as text, patterns, and thin structures, remains challenging. By deforming the kernel domain rather than only modulating its opacity or appearance, the proposed formulation enables each Gaussian primitive to capture complex local geometry more efficiently.

3.2.2 2D GAUSSIAN SPLATTING

We adopt the 2D Gaussian Splatting representation introduced in (Huang et al., 2024). Each primitive lies on a local tangent plane with center p_k, orthonormal basis (t_β, t_γ, n_k), and scales (s_β, s_γ). Local coordinates on the plane are (β, γ):

$$P(\beta, \gamma) = p_k + s_\beta t_\beta \beta + s_\gamma t_\gamma \gamma. \tag{4}$$

The spatial weight is $\mathcal{G}(\beta, \gamma) = \exp[-(\beta^2 + \gamma^2)/2]$, and splats are projected and composited in image space. Appearance is modeled using spherical harmonics, producing one view-dependent color per primitive, without spatial variation across the tangent plane.

3.2.3 FLEXICOORD FOR 2DGS

We represent the reparameterization field for each Gaussian with a per-Gaussian texture $\mathbf{T}_i$, as shown in Fig. 2. Given the local intersection coordinates $\mathbf{s}(\mathbf{x}) = (\beta, \gamma)$ on the tangent plane, the corresponding texture coordinates (u, v) are computed as

$$u = \frac{\beta + \xi}{2\xi}\tau, \quad v = \frac{\gamma + \xi}{2\xi}\tau, \tag{5}$$

where ξ controls the effective reparameterization support, and τ is the texture resolution. The offset $\Delta_\phi(\mathbf{x})$ are bilinearly sampled from the texture map $\mathbf{T}_i(u, v)$. Thus, the deformed Gaussian value $\hat{\mathcal{G}}_i$ at $s(\mathbf{x})$ was evaluated as

$$\hat{\mathcal{G}}_i(s(\mathbf{x})) = \mathcal{G}_i(s(\mathbf{x}) + \lambda \mathbf{T}_i(u, v)), \tag{6}$$

and composite the SH encoded color $c_i(\mathbf{p})$ using volumetric alpha blending weighted by the reparameterized Gaussian $\hat{\mathcal{G}}_i(s(x))$:

$$c(\mathbf{p}) = \sum_{i=1}^{N} c_i(\mathbf{p})\, \hat{\mathcal{G}}_i(\mathbf{s}(\mathbf{x}))\, o_i \times \prod_{j<i} \Big[1 - \hat{\mathcal{G}}_j(\mathbf{s}(\mathbf{x}))\, o_j\Big]. \tag{7}$$

3.3 FLEXICOORD-NGP

3.3.1 NEURAL GRAPHICS PRIMITIVE (NGP)

NGP introduced Multi-resolution Hash grids as a fast and powerful representation for spatial positional encoding, which was applied to neural radiance fields and image encoding. Given an input coordinate $\mathbf{x} \in \mathbb{R}^d$ for the ith layer, the grid corners that are closest to $\mathbf{x}$ were found, and the feature vectors of these corners were queried by spatial hashing (Teschner et al., 2003),

$$h(\mathbf{x}) = (\oplus_{i=1}^{d} \mathbf{x}_i \pi_i)\ mod\ T, \tag{8}$$

where $\oplus$ denotes the bit-wise XOR operation, T is the hash table size, and π_i are unique, large prime numbers. Then the queried feature vectors were d-linearly interpolated, where $d = 2\ or\ 3$ is the spatial dimension of the hash grid. This process was independent for all layers' hash grids with different resolutions. The interpolated feature vectors for each level were concatenated and fed to the MLP.

3.3.2 FLEXICOORD FOR IMAGE ENCODING

NGP was used as a positional encoding for image representation (Müller et al., 2022). The multi-resolution hash grid allows NGP to capture fine-scale details in the images at high resolutions while maintaining manageable memory usage compared to a dense multi-resolution grid. However, we observe that the reconstructed images can still suffer from over-smoothing in high-frequency regions and thin structures, such as text boundaries, wires, and cracks, as shown in Fig. 1 (B) and Fig. 4.

Thus, we propose FlexiCoord-NGP, where a reparameterization field is added to the multi-resolution hash grid to increase its ability to encode high-frequency details. Instead of adopting a multi-resolution hash grid, we encode the reparameterization field with a single-level hash grid. This design is motivated by the observation that the frequency distribution of the learned coordinate offsets remains spatially consistent across resolution levels.

Hence, as shown in Fig. 2 (B), the pipeline of FlexiCoord-NGP for image encoding is as follows: 1) Sampling random coordinates $\mathbf{x} = (x, y)$ on the image, the $\Delta_\phi(\mathbf{x})$ is encoded by a single-level hash grid $H(\mathbf{x})$. 2) The feature vector is queried by the reparameterized coordinates on the multi-resolution hash grid $Feat = MH(\mathbf{x} + \lambda H(\mathbf{x}))$. 3) The RGB colors are predicted by the MLP with $Feat$ as features.

3.4 FLEXICOORD-VM

3.4.1 VECTOR-MATRIX (VM) DECOMPOSITION

Low-rank decomposition is commonly used to compactly represent high-dimensional tensors. A representative variant is the vector-matrix (VM) decomposition for $3D$ tensor introduced in (Chen et al., 2022), where each VM component consists of three matrix factors, corresponding to the XY, XZ, and YZ planes, together with three vector factors along the X, Y, and Z axes, as shown in Fig. 2 (C). Considering a $3D$ tensor $\tau_M \in \mathbb{R}^{I \times J \times K}$, which represents a volumetric medical scan (CT or MRI), we can factorize it as

$$\tau_M = \sum_{r=1}^{R} (v_X \circ M_{YZ} + v_Y \circ M_{XZ} + v_Z \circ M_{XY}). \tag{9}$$

3.4.2 FLEXICOORD FOR VOLUMETRIC DATA COMPRESSION

The reparameterization Field in the FlexiCoord-VM was represented by nested two-layer offset VM blocks VM_1 and VM_2 to handle the fine-grained structures, such as bones and vessels, as shown in Fig. 2 (C). The pipeline of the FlexiCoord-VM for volumetric data encoding is as follows: 1) Sampling random coordinates $\mathbf{x} = (x, y, z)$, the first-layer reparameterization field Δ_{ϕ_1} is encoded by the first VM block $VM_1(\mathbf{x}) = \Delta_1(\mathbf{x})$. 2) The second-layer reparameterization field Δ_{ϕ_2} is encoded by second VM block $VM_2(\mathbf{x} + \lambda_1 \Delta_{\phi_1}(\mathbf{x})) = \Delta_{\phi_2}(\mathbf{x})$. 3) The feature vector is queried by the reparameterized coordinates on the feature VM block, $Feat = VM_3(\mathbf{x} + \lambda_2 \Delta_{\phi_2}(\mathbf{x}))$. 4) The densities are predicted by the MLP with $Feat$ as input. The feature VM block VM_3 factorizes the tensor according to Eq. 9. To better integrate spatial information across different axes, we use a modified structure for the offset VM blocks VM_1 and VM_2 as detailed in the appendix.

4 EXPERIMENTS

4.1 IMPLEMENTATION DETAILS

FlexiCoord-GS We developed a lightweight CUDA kernel for deformable Gaussian kernels described in section 3.2, which was built on gsplat (Ye et al., 2025). To reduce the storage memory, we conduct a post-process to filter out the reparameterization field that makes ignorable deformations. If the L^2 difference between the deformed and undeformed Gaussian is less than 0.02, the reparameterization field for this Gaussian will be discarded. For optimization, we followed Textured Gaussians (Chao et al., 2025) with a two-stage optimization for stability. In the first stage, the 2D Gaussian primitives are trained using the standard losses from 3D Gaussian Splatting (Kerbl et al., 2023), combining $\mathcal{L}_1$ photometric and structural similarity ($\mathcal{L}_{\text{SSIM}}$) terms, augmented with a mask

loss $\mathcal{L}_\alpha$ when ground-truth masks are available:

$$\mathcal{L} = \eta\mathcal{L}_1 + (1-\eta)\mathcal{L}_{\text{SSIM}} + \mathcal{L}_\alpha, \quad \eta = 0.2. \tag{10}$$

This stage runs for 30,000 iterations. In the second stage, we optimize the per-Gaussian reparameterization field while fine-tuning the Gaussian parameters $\{p_k, s_\beta, s_\gamma, t_\beta, t_\gamma, o, \text{SH}\}$ without pruning or densification, for another 30,000 iterations. Learning rates for the reperameterization field is set to 1×10^{-3}, and reparameterization weight $\lambda = 1$.

FlexiCoord-NGP We adapt tiny-cuda-nn (Müller, 2021) as the base hash grid and MLP implementation. The hash table size for all experiments is 2^{16}, the resolution for the reparameterization field is 512, and a fully fused MLP that has 2 hidden layers with 64 neurons was used. The reparameterization weight $\lambda = 0.001$, learning rates for MLP and Tensor-VM are 0.001 and 0.01, respectively.

FlexiCoord-VM Although we adapt the VM structure introduced by (Chen et al., 2022) as the representation, we adapt the implementation from tiny-cuda-nn (Müller, 2021) for fast GPU-based bilinear interpolation of the grid feature query. The reparameterization weight $\lambda = 0.01$, learning rates for MLP and Tensor-VM are 0.001 and 0.01, respectively. All experiments are run on a single NVIDIA RTX A6000 GPU.

4.2 Datasets and Evaluation Metrics

Dataset. We evaluated FlexiCoord-GS on three standard benchmarks for novel view synthesis: Blender (Mildenhall et al., 2021) (all scenes), MipNeRF 360 v2 (Barron et al., 2022) (publicly available scenes including Bicycle, Counter, Garden, Room, Bonsai, Kitchen, and Stump), Tanks & Temples (Knapitsch et al., 2017) (Caterpillar, Train, Truck, Church, Courthouse, Family, Ignatius, M60, Palace, and Panther).

FlexiCoord-NGP was evaluated on five ultra-resolution images obtained from the internet with more than 20 million pixels, and FlexiCoord-VM was evaluated on a subset of the Medical Segmentation Decathlon (Antonelli et al., 2022), including MRIs of the heart, CTs of the lungs and hepatic vessels.

Evaluation Protocol For FlexiCoord-GS, we compare methods at fixed primitive budgets (50% and 10%), as primitive count governs representation granularity and rendering workload; FPS and model size are also reported to account for per-primitive overhead. For FlexiCoord-NGP and VM, we compare reconstruction quality under comparable memory budgets.

Metrics. We measured the Novel-view synthesis quality quantitatively by three well-known image metrics: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al., 2018). For image and volumetric data encoding, we report Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM).

4.3 Experimetal results

4.3.1 FlexiCoord-GS

Quantitative Results. We evaluated our method against 2DGS (Huang et al., 2024), FACT-GS (Xie et al., 2026) and Textured GS (Chao et al., 2025) under varying primitive budgets (50%, 10%). All methods follow the same 2DGS training protocol (Huang et al., 2024). By directly deforming the geometry of Gaussian kernels, our method gains greater flexibility in representing complex structures, including thin geometry and text boundaries. This improved geometric expressiveness leads to consistently higher reconstruction fidelity across all datasets and parameter settings, as shown in Tab. 1. More than that, the learned reparameterization reshapes the Gaussian density distribution to concentrate more on the target geometry, which avoids unnecessary density spread in regions that contribute little to the final rendering and reduces the number of effective ray-Gaussian contributions compared with 2DGS. As a result, although this reparameterization requires additional computation during each ray-Gaussian intersection, its overall inference speed remains comparable or even faster than 2DGS and is approximately 3 times faster than texture-based GS (Chao et al., 2025; Xie et al., 2026).

Qualitative Results The Qualitative results of our method compared to 2DGS (Huang et al., 2024), Textured GS (Chao et al., 2025), and FACT-GS (Xie et al., 2026) under a few primitives (10%)

	Blender				MipNeRF360				Tanks and Temples			
	PSNR↑	SSIM↑	LPIPS↓	FPS↑	PSNR↑	SSIM↑	LPIPS↓	FPS↑	PSNR↑	SSIM↑	LPIPS↓	FPS↑
2DGS(50%)	33.00	0.964	0.031	**400**	28.85	0.863	0.108	**125**	23.65	0.819	0.188	**172**
TexturedGS(50%)	33.29	0.965	0.029	139	29.04	0.866	0.104	39	23.94	0.824	0.182	52
FACT-GS(50%)	33.50	0.966	0.027	139	**29.07**	0.869	**0.100**	44	23.90	0.826	**0.176**	57
Ours(50%)	**33.53**	**0.967**	**0.026**	385	29.03	**0.870**	0.101	123	**24.06**	**0.832**	**0.176**	170
2DGS(10%)	30.34	0.945	0.059	435	27.52	0.815	0.180	222	22.93	0.783	0.257	270
TexturedGS(10%)	30.72	0.950	0.057	213	27.73	0.822	0.173	83	23.20	0.789	0.252	92
FACT-GS(10%)	31.24	0.953	0.047	208	27.89	0.832	0.154	94	23.23	0.796	0.236	99
Ours(10%)	**31.47**	**0.954**	**0.043**	**455**	**28.04**	**0.837**	**0.153**	**238**	**23.24**	**0.799**	**0.234**	**278**

Table 1: Quantitative comparison with 2DGS (Huang et al., 2024), TexturedGS (Chao et al., 2025), and FACT-GS (Xie et al., 2026) using 50% and 10% of the original Gaussian primitives. FlexiCoord-GS provides a favorable quality–speed trade-off, achieving particularly strong reconstruction quality at the 10% budget while achieving faster rendering speeds than 2DGS and approximately three times higher than texture-based methods. (Best scores are shown in **bold**, second-best are underlined.)

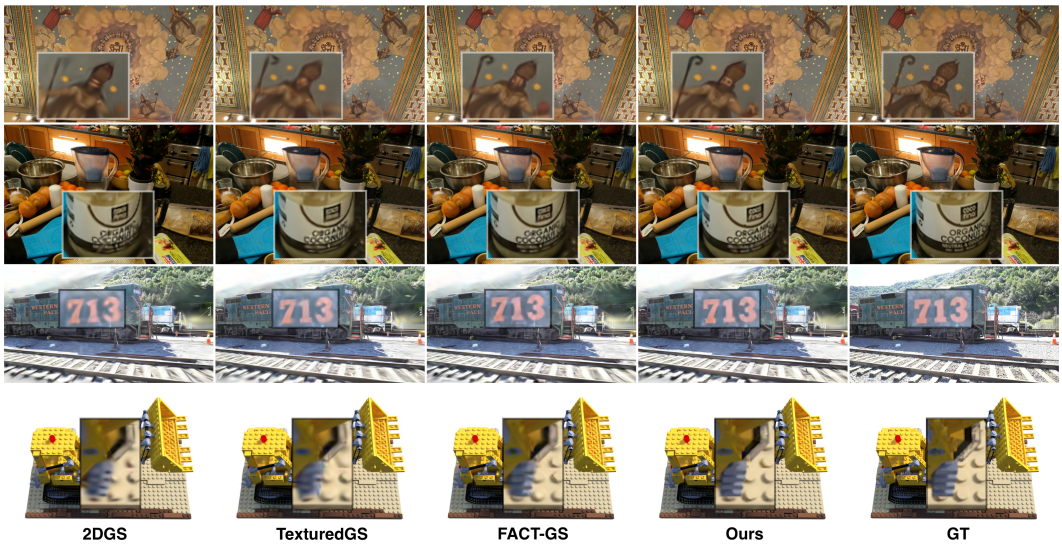

Figure 3: Qualitative comparison under a limited primitive budget of 10%. Compared with 2DGS (Huang et al., 2024), FlexiCoord-GS more accurately reconstructs thin structures, text boundaries, and fine patterns. Compared with TexturedGS and FACT-GS (Chao et al., 2025; Xie et al., 2026), directly deforming the Gaussian kernels produces sharper boundaries while retaining more efficient rendering.

is shown in Fig. 3. The results demonstrate the expected advantages of our method. First, fine-scale details can be captured more efficiently with a limited numer of primitives, as shown by the character's eyes in the painting and the thin text on the jar. Second, sharp and thin structures are reconstructed more faithfully, such as the thin square bricks in the LEGO scene.

4.3.2 FLEXICOORD-NGP

Quantitative Results. We validate the efficiency of our method in image encoding on five ultra-resolution images collected from online sources. As shown in Tab. 2, our method achieves higher encoding fidelity compared to the NGP method under a similar memory budget, where both methods have a model size of around 2.2MB.

Qualitative Results. The qualitative results in Fig. 4 and Fig. 1 further demonstrate the effectiveness of FlexiCoord for image encoding. Compared with the baseline representation, our method better preserves high-frequency details that are easily over-smoothed or corrupted by noise under limited representation capacity. For example, the text in Fig. 1(B) and the cracks at the left of Fig. 4 and the tower tip at the right are reconstructed with clearer boundaries. These results indicate that coordinate reparameterization helps the encoding allocate effective representational capacity toward locally complex regions, where small spatial misalignments can lead to noticeable loss of detail.

Ultra-resolution Image				Volumetric Medical Data			
	PSNR	SSIM	Mem (MB)		PSNR	SSIM	Mem (MB)
NGP	25.67	0.598	2.2	Tensor-VM	37.29	0.933	4.3
Ours	**26.07**	**0.616**	2.2	Ours	**38.23**	**0.941**	4.4

Table 2: Quantitative results for FlexiCoord-NGP and VM. (Best scores are shown in **bold**.)

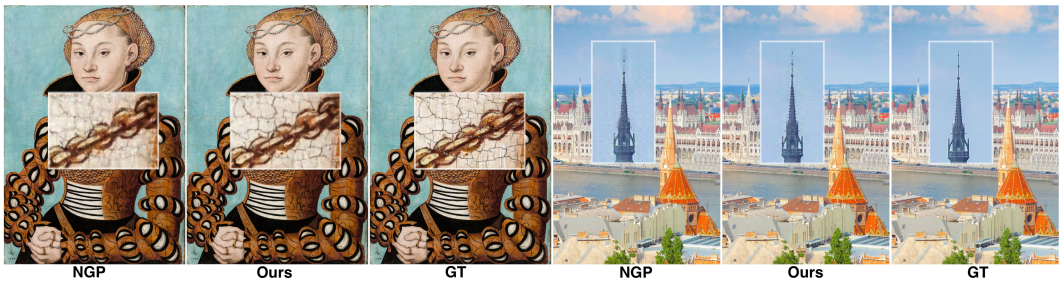

Figure 4: Qualitative comparison of ultra-high-resolution image encoding. FlexiCoord-NGP better preserves high-frequency details, including cracks, text, and the tip of the tower.

In addition to improving fine-scale structures, our method also produces cleaner reconstructions in smooth regions, such as the yellow screen and the red arrow shown in the first row of Fig. 5. This suggests that FlexiCoord does not merely increase high-frequency response uniformly; instead, it adapts the coordinate domain according to local signal complexity. As a result, capacity is concentrated around detailed regions while smooth areas remain stable and less noisy. This behavior leads to a better balance between detail preservation and noise suppression, improving the overall fidelity of ultra-resolution image encoding.

4.3.3 FLEXICOORD-VM

Quantitative Results. We evaluate the encoding efficiency of our method on 10 volumetric medical images, including MRIs of the heart, CTs of the lungs and hepatic vessels. As shown in Tab. 2, our method achieves higher PSNR and SSIM metrics, with a maximum PSNR improvement of nearly 1dB. The comparison was conducted under a similar memory budget, where Tensor-VM has a model size of 4.3MB, and ours has a model size of 4.4MB.

Qualitative Results. We demonstrate the improved efficiency of FlexiCoord-VM in volumetric medical data compression via quantitative results in Fig. 1 (C) and Fig. 5 in the appendix. FlexiCoord-VM achieves higher encoding fidelity by more clearly preserving thin anatomical structures, including bones and vessels, while reducing noise in large homogeneous regions. This behavior further demonstrates that coordinate reparameterization adaptively allocates representation capacity according to local signal complexity.

5 CONCLUSION

In this work, we presented FlexiCoord, a general strategy that augments existing spatial functions with learned coordinate reparameterization. By adapting the coordinate domain to the target signal, FlexiCoord improves the representation of sharp boundaries, fine details, and thin structures without redesigning the underlying representation.

We instantiated FlexiCoord with Gaussian kernels, multi-resolution hash grids, and vector–matrix decompositions. At fixed primitive budgets, FlexiCoord-GS improves reconstruction fidelity while maintaining rendering speeds comparable to 2DGS and substantially higher than texture-based Gaussian methods. Under comparable memory budgets, FlexiCoord-NGP and FlexiCoord-VM improve image and volumetric reconstruction, preserving fine structures while reducing noise in homogeneous regions. These results demonstrate that learned coordinate reparameterization provides a broadly applicable mechanism for improving the quality–efficiency trade-off of spatial representations.

6 AI DISCLOSURE

In this work, we used generative AI tools to summarize or analyze the existing literature, edit the research paper to improve readability, and edit software code. We have not used generative AI tools for concept formalization and experiment design or execution. We have reviewed all AI-assisted work. We reviewed the LLM-generated literature analysis for correct citations. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54(10):105–112, 2011.

Michela Antonelli, Annika Reinke, Spyridon Bakas, Keyvan Farahani, Annette Kopp-Schneider, Bennett A Landman, Geert Litjens, Bjoern Menze, Olaf Ronneberger, Ronald M Summers, et al. The medical segmentation decathlon. *Nature communications*, 13(1):4128, 2022.

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 *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5855–5864, 2021.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5470–5479, 2022.

Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 130–141, 2023.

Brian Chao, Hung-Yu Tseng, Lorenzo Porzi, Chen Gao, Tuotuo Li, Qinbo Li, Ayush Saraf, Jia-Bin Huang, Johannes Kopf, Gordon Wetzstein, et al. Textured gaussians for enhanced 3d scene appearance modeling. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 8964–8974, 2025.

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.

Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12479–12488, 2023.

Yang Fu, Sifei Liu, Amey Kulkarni, Jan Kautz, Alexei A Efros, and Xiaolong Wang. Colmap-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20796–20805, 2024.

Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2009.

Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. Towards internet-scale multi-view stereo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pp. 1434–1441. IEEE, 2010.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, pp. 1–11, 2024.

Zhentao Huang and Minglun Gong. Textured-gs: Gaussian splatting with spatially defined color and opacity. *arXiv preprint arXiv:2407.09733*, 2024.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017.

Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3d gaussian representation for radiance field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21719–21728, 2024.

Jiahui Lei, Yijia Weng, Adam W Harley, Leonidas Guibas, and Kostas Daniilidis. Mosca: Dynamic gaussian fusion from casual videos via 4d motion scaffolds. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 6165–6177, 2025.

Haolin Li, Jinyang Liu, Mario Sznaier, and Octavia Camps. 3d-hgs: 3d half-gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 10996–11005, 2025.

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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8456–8465, 2023.

David B Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16252–16262, 2022.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, pp. 405–421. Springer, 2020.

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. tiny-cuda-nn, 4 2021. URL `https://github.com/NVlabs/tiny-cuda-nn`.

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.

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

Victor Rong, Jingxiang Chen, Sherwin Bahmani, Kiriakos N Kutulakos, and David B Lindell. Gstex: Per-primitive texturing of 2d gaussian splatting for decoupled appearance and geometry modeling. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 3508–3518. IEEE, 2025.

Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4104–4113, 2016.

Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 1, pp. 519–528. IEEE, 2006.

Ruizhi Shao, Zerong Zheng, Hanzhang Tu, Boning Liu, Hongwen Zhang, and Yebin Liu. Tensor4d: Efficient neural 4d decomposition for high-fidelity dynamic reconstruction and rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16632–16642, 2023.

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.

Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pp. 835–846. 2006.

Noah Snavely, Steven M Seitz, and Richard Szeliski. Modeling the world from internet photo collections. *International journal of computer vision*, 80(2):189–210, 2008.

Yunzhou Song, Heguang Lin, Jiahui Lei, Lingjie Liu, and Kostas Daniilidis. Hdgs: Textured 2d gaussian splatting for enhanced scene rendering. *arXiv preprint arXiv:2412.01823*, 2024.

Hanxiao Sun, Yupeng Gao, Jin Xie, Jian Yang, and Beibei Wang. Svg-ir: Spatially-varying gaussian splatting for inverse rendering. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 16143–16152, 2025.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020.

Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8248–8258, 2022.

Matthias Teschner, Bruno Heidelberger, Matthias Müller, Danat Pomerantes, and Markus H Gross. Optimized spatial hashing for collision detection of deformable objects. In *Vmv*, volume 3, pp. 47–54, 2003.

Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5481–5490. IEEE, 2022.

Tianhao Xie, Linlian Jiang, Xinxin Zuo, Yang Wang, and Tiberiu Popa. Fact-gs: Frequency-aligned complexity-aware texture reparameterization for 2d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, pp. 202–212, June 2026.

Rui Xu, Wenyue Chen, Jiepeng Wang, Yuan Liu, Peng Wang, Lin Gao, Shiqing Xin, Taku Komura, Xin Li, and Wenping Wang. Supergaussians: Enhancing gaussian splatting using primitives with spatially varying colors. *arXiv preprint arXiv:2411.18966*, 2024a.

Tian-Xing Xu, Wenbo Hu, Yu-Kun Lai, Ying Shan, and Song-Hai Zhang. Texture-gs: Disentangling the geometry and texture for 3d gaussian splatting editing. In *European Conference on Computer Vision*, pp. 37–53. Springer, 2024b.

Vickie Ye, Ruilong Li, Justin Kerr, Matias Turkulainen, Brent Yi, Zhuoyang Pan, Otto Seiskari, Jianbo Ye, Jeffrey Hu, Matthew Tancik, et al. gsplat: An open-source library for gaussian splatting. *Journal of Machine Learning Research*, 26(34):1–17, 2025.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Xin Zhang, Anpei Chen, Jincheng Xiong, Pinxuan Dai, Yujun Shen, and Weiwei Xu. Neural shell texture splatting: More details and fewer primitives. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 25229–25238, 2025.

Yang Katie Zhao, Shang Wu, Jingqun Zhang, Sixu Li, Chaojian Li, and Yingyan Celine Lin. Instant-nerf: Instant on-device neural radiance field training via algorithm-accelerator co-designed near-memory processing. In *2023 60th ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6. IEEE, 2023.

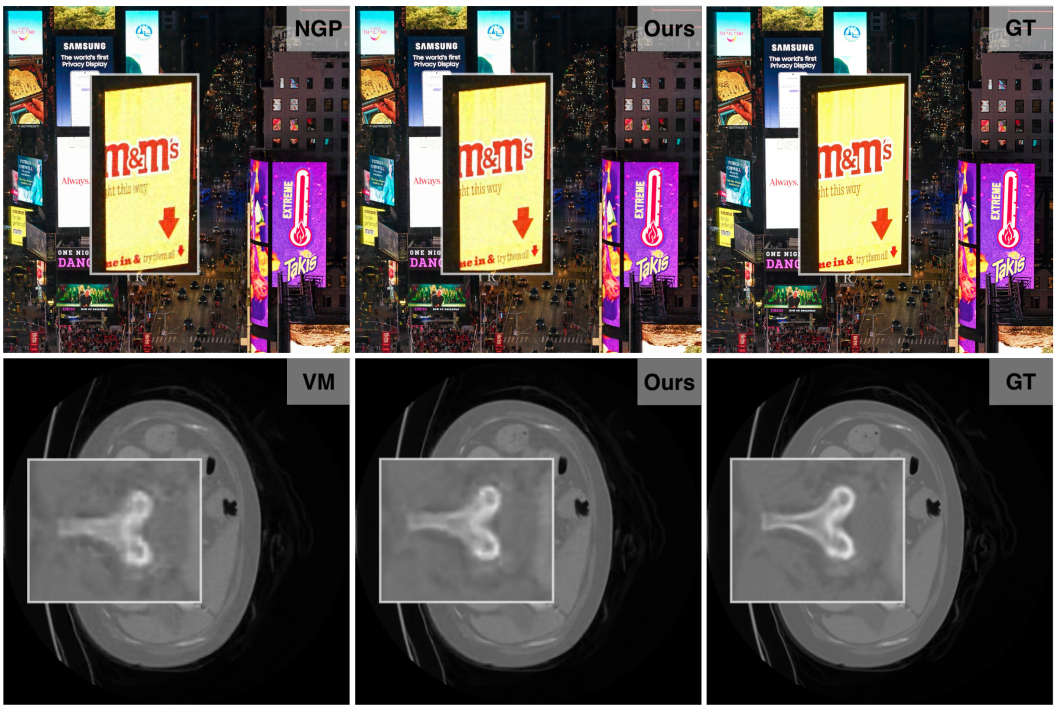

Figure 5: More qualitative results for FlexiCoord-NGP and VM.

A Appendix

A.1 More Qualitative Results

More qualitative results of the FlexiCoord-NGP and FlexiCoord-VM is shown in Figure. 5. In addition to improving fine-scale structures, our method also produces cleaner reconstructions in smooth regions in both ultra-resolution images and volumetric medical data. As shown in Fig. 5, our method reduces the noise in the yellow screen and the red arrow of the ultra-resolution image in the first row and in large homogeneous regions in the medical scan. This behavior further demonstrates that coordinate reparameterization adaptively allocates representation capacity according to local signal complexity.

A.2 Design of the offset VM blocks

While the feature VM block VM_3 factorizes the tensor according to Eq. 9, we design the offset VM blocks VM_1 and VM_2 with a slightly different structure. Given input coordinates $\mathbf{x} \in \mathbb{R}^3$, the offset VM blocks query the reparameterization field Δ_{ϕ_i} for each axis using the corresponding element in each plane and vector

$$\begin{aligned}
\Delta_\phi^x &= \frac{1}{2}(M_{XY}(\mathbf{x}).x + M_{XZ}(\mathbf{x}).x) * v_X(\mathbf{x}) \\
\Delta_\phi^y &= \frac{1}{2}(M_{XY}(\mathbf{x}).y + M_{YZ}(\mathbf{x}).y) * v_Y(\mathbf{x}) \\
\Delta_\phi^z &= \frac{1}{2}(M_{XZ}(\mathbf{x}).z + M_{YZ}(\mathbf{x}).z) * v_Z(\mathbf{x})
\end{aligned} \tag{11}$$

where $M_{XY}(\mathbf{x})$, $M_{XZ}(\mathbf{x})$, $M_{YZ}(\mathbf{x})$, $v_X(\mathbf{x})$, $v_X(\mathbf{x})$, $v_X(\mathbf{x})$ are calculated by bilinear/linear interpolation of the corresponding axis, and $M_{XY}(\mathbf{x}).x$ represents the x axis of the $M_{XY}(\mathbf{x})$.

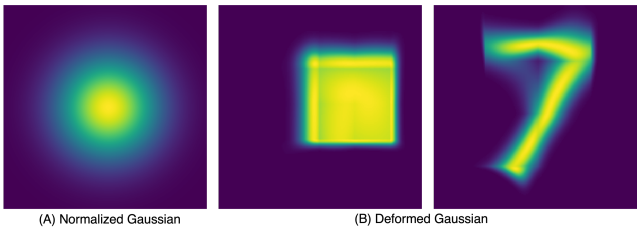

Figure 6: FlexiCoord transforms a conventional elliptical Gaussian kernel into more flexible spatial supports. (A) Original normalized Gaussian kernel. (B–C) Reparameterized kernels fitted to a square and the digit "7."

	Blender	MipNeRF360	Tanks and Temples
2DGS	2.1MB	61.5MB	30.8MB
TexturedGS	4.3MB	128.2MB	64.2MB
FACT-GS	3.9MB	117.8MB	59.0MB
Ours	3.7MB	102.1MB	49.8MB

Table 3: Model sizes at the 10% primitive budget. Post-processing (described in 4.1.) removes displacement fields associated with negligible deformations, making FlexiCoord-GS more compact than the texture-based Gaussian methods, although it remains larger than the original 2DGS representation.

A.3 Deformable Gaussian

As shown in Fig. 6, a normalized Gaussian (A) can be transformed by Eq. 3 into shapes such as a square and a "seven". This process not only changes the kernel contour, but also redistributes its density to better match the target shape.

A.4 Model Size for FlexiCoord-GS

The model size of all methods with 10% primitives is shown in Tab. 3. Compared with texture-based GS, our method achieves the smallest model size after applying the post-processing strategy described in 4.1. Compared with 2DGS, our method introduces additional memory overhead due to the learned reparameterization field. However, this overhead leads to a favorable trade-off: our method achieves comparable or faster inference speed while producing significantly better visual details. This is particularly important for applications that require both real-time performance and high-fidelity reconstruction, such as interactive scene visualization, editing, and rendering of fine-scale structures.