MLP Splatting: Object-Centric Neural Fields

Shinjeong Kim*, Yuzhou Cheng*, Xin Kong*,
Paul H. J. Kelly, and Andrew J. Davison

Department of Computing, Imperial College London
{s.kim, y.cheng25, x.kong21, p.kelly, a.davison}@imperial.ac.uk

Abstract. 3D representations are fundamental for scene rendering, understanding, and interaction. Approaches like Neural Radiation Fields and 3D Gaussian Splatting achieve impressive photorealistic novel-view synthesis, but lack the ability to easily decompose scene elements into object level, requiring additional segmentation or grouping. We present MLP-Splatting, a method that enables scene decomposition via a few expressive light-field primitives while providing photorealistic novel-view synthesis.

MLP-Splatting models each primitive as an independent compact MLP with localized spatial support that predicts radiance and opacity. In contrast to low-level Gaussian primitives or a single global radiance field, our neural primitives provide greater expressive capacity while remaining spatially localized. Rendering is performed through efficient sparse volumetric compositing over ray–primitive interactions.

Our primitives are supervised using RGB supervision alone, which yields primitives that represent local scene regions often corresponding to objects or object parts, enabling interactive object-level editing without segmentation masks by selecting a handful of primitives. Our method, augmented with optional semantic feature distillation, enables open-vocabulary scene interaction and open-set instant segmentation. Compared to state-of-the-art semantic 3DGS methods, we achieve substantially lower memory usage ($1/7\times$) and faster rendering ($5\times$).

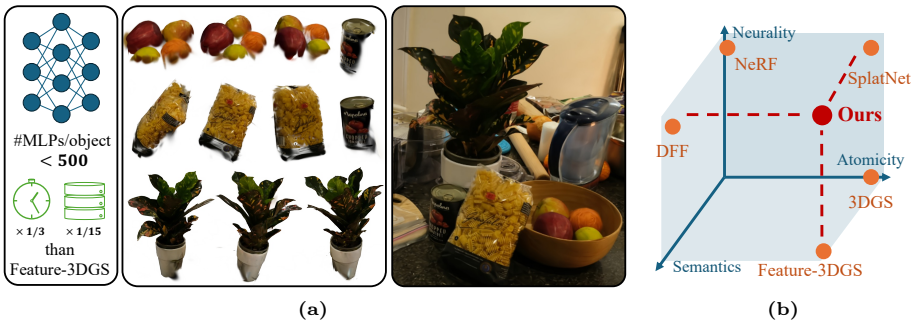
Project Page: https://shinjeongkim.com/mlp-splatting

Keywords: 3D Reconstruction · Radiance Fields · Novel View Synthesis

1 Introduction

Reconstructing a 3D scene remains a fundamental and enduring problem in computer vision, closely tied to the scene representation. Different application goals have led to diverse primitives, ranging from discrete point clouds featuring ease of obtaining and processing, point-based primitives such as 3D Gaussian Splatting (3DGS) [12] for editability, mesh representations [20] tailored for watertight modeling and physical simulation, and neural radiance fields (NeRF) [25] for photorealistic novel-view synthesis.

* Equal contribution.



(a) (b)

Fig. 1: MLP-Splatting Overview. **(a)** Example group of learned primitives corresponding to object or object parts, and the rendered scene. **(b)** Conceptual comparison of 3D scene representations along neurality, atomicity, and semantic awareness. Our method combines the expressiveness of neural fields with the atomicity of primitives, enabling representations that better align with semantic structure.

From another perspective, scene representations differ fundamentally in the level of abstraction of their primitives. Although many existing representations operate at the level of low-level geometric or radiometric elements, a natural question arises: should higher-level entities, such as objects or object parts, serve as the basic functional units of the representation?

In this work, we revisit this question from an object-level perspective. Most existing scene representations are not inherently object-centric: their basic primitives correspond to low-level geometric [5,9,12,20,23] or radiometric elements [25, 26], while objects emerge only as derived groupings or semantic overlays. By object-centric, we mean that the representation's fundamental functional units correspond to objects or meaningful parts of objects.

Although recent advances have begun to incorporate object awareness into neural scene representations, this awareness is typically introduced by distilling semantic knowledge onto pre-existing representations, such as implicit radiance fields in NeRF-based methods [14, 16, 18, 41] or 3DGS [4, 39, 43]. In both cases, semantics are imposed on representations originally designed for photometric realism rather than arising from representations whose primitive units are themselves object- or part-level entities.

Yet we argue that the structure of real scenes itself naturally gives rise to object- and part-level organization, and that a scene representation can be designed to capture this structure directly. Many object-level properties in 3D space, such as spatial occupancy, color, and density exhibit strong coherence within an object, while sharp transitions are concentrated primarily near object boundaries, despite potentially high intrinsic complexity. We therefore introduce MLP-Splatting, which assigns independent MLP-based primitives to spatially significant parts of objects. These primitives collectively form a scene-level radiance field for rendering. Each primitive is continuous and differentiable, shares no parameters with the others, and in principle has unbounded expressive capacity, making it well-suited to modeling coherent object-level structure.

To showcase this object-centric property more clearly, we optionally augment the representation with semantic guidance through feature distillation [16, 43]. This allows the learned primitives and their associated embeddings to support tasks such as segmentation and language-guided editing. With this supervision, MLP-Splatting achieves semantic metrics comparable to Feature-3DGS [43], despite exhibiting orders-of-magnitude differences in model memory due to the nature of the representation. We further analyze the theoretical memory bound required to reach the same photometric error.

In summary, our contributions are fourfold:

1. *Object-centric neural scene representation.* We introduce *MLP-Splatting*, a neural scene representation in which each primitive is modeled as an independent perceptron. This formulation treats objects or parts of objects as the fundamental functional units of the representation.
2. *Emergent object-level structure.* Under pure RGB supervision, primitives naturally specialize to coherent scene regions, enabling object-level editing **without segmentation**. With feature distillation, we achieve state-of-the-art rendering quality and competitive semantics in the Replica [34] and ScanNet [6] datasets.
3. *Efficient neural-primitive rendering.* Our scene paradigm and efficient tile-based compositing pipeline enable real-time rendering at $\sim$**10** FPS (vs. $\sim$**2** FPS in [43]) with only **120 MB** of memory (vs. **880 MB** in [43]) on average.
4. *Theoretical capacity analysis.* We derive the *Equal-Quality Memory Law*, showing that achieving the same pixel error requires $\Omega(\varepsilon^{-2})$ parameters for Gaussian splatting but only $\Theta(\varepsilon^{-3/s})$ for MLP-Splatting, where s is the local smoothness order of the scenes.

2 Related Work

2.1 3D Scene Representations

NeRFs [25] represent a scene as a continuous function that maps 3D positions and viewing directions to densities and radiance, optimized by differentiating the volumetric rendering of posed images. A large body of work has since improved the fidelity–efficiency trade-off [1, 3, 7, 26, 31] and broadened the operating regime [2, 24]. Instant-NGP [26, 35] introduces multiresolution hash encodings and fused CUDA kernels to accelerate training and inference drastically. For large, unbounded outdoor captures, mip-NeRF 360 [2] addresses aliasing and scale imbalance via cone-based integration and scene parameterization, while Zip-NeRF [3] further improves anti-aliasing by combining rendering/signal-processing ideas with grid-based representations.

In parallel, methods based on optimizable explicit primitives, such as 3D/2D Gaussian Splatting [11, 12], ellipsoid rendering [23, 44], polyhedral element rendering [8, 9, 22], etc., replace dense ray marching on a volumetric neural field with a fast visibility-aware rasterization or efficient closed-form volumetric rendering on a set of independently-optimizable unit geometric primitives, achieving

high-quality real-time rendering. Our work seeks to combine the representational flexibility of neural fields with the computational efficiency of splatting-style pipelines, while pursuing a distinct representational goal: making object- and part-level structure emerge naturally from the scene representation.

2.2 Object-level Scene Representations

A growing line of research seeks representations that explicitly expose objects for editing, tracking, and compositional reasoning. iLabel [42] revealed that a single MLP trained for scene representation automatically has a certain level of object awareness, such that 2D semantic masks can be predicted by a very small number of interactive annotations. Various works have explored object-centric scene representations that model a scene as a composition of per-object components [10, 17, 27, 36, 37]. Such decomposition enables object-level manipulations such as re-arrangement, relighting, editable rendering, and tracking [10].

Within radiance-field methods, object-compositional formulations introduce an explicit object branch, often conditioned on learnable codes, to support editable scene rendering [37], while real-time object-level mapping methods model each instance with a separate compact neural field to build object-level neural 3D scene representations online [17]. In Panoptic Neural Fields [18], dynamic scenes are decomposed into object instances, each of which is represented by an MLP, and background regions, which are modeled separately. [18] relies on external pose estimation, tracking, and 2D panoptic predictions as pseudo-supervision.

For 3DGS, most extensions remain Gaussian-centric, with object-level structure introduced through grouping or auxiliary embeddings attached to individual Gaussians. Gaussian Grouping [39], for example, learns per-Gaussian-grouping features guided by multiview mask predictions from a promptable segmentation model such as SAM [15], allowing open-world 3D segmentation and editing. These works demonstrate the value of object awareness; however, object-level entities are typically recovered through supervision, clustering, or auxiliary feature fields rather than serving as intrinsic functional primitives. By contrast, our approach explicitly treats objects and parts as independent functional units, each learned by a compact MLP, and composes these local models to render the scene.

2.3 Open World Semantic Understanding

Open-world semantics aims to recognize and localize concepts beyond a fixed label set, often by leveraging vision and vision-language foundation models. CLIP [30] provides a shared vision-language embedding space that supports zero-shot recognition and retrieval via text prompts. Promptable segmentation models such as SAM [15] enable class-agnostic mask generation at scale, while open-set object detectors such as Grounding DINO [19] perform phrase grounding and text-conditioned detection by integrating language into transformer-based detection. Prompt-based segmentation methods such as CLIPSeg [21] further bridge text or image prompts to pixel-level masks.

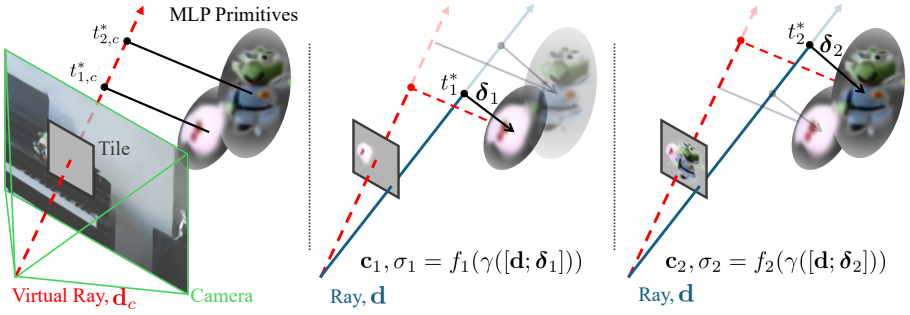


Fig. 2: Procedure of MLP-Splatting pipeline. **(Left)** For each tile, sorting for MLPs is performed per tile, based on the depth calculated with respect to a virtual ray passing through the center of the tile; only MLPs for which the support region is splatted to the image plane that overlap with the tile are considered. **(Center)** Based on the sorted order, we perform the MLP weight loading and forward pass cooperatively per GPU thread block responsible for every pixel in the tile to obtain the color and density. **(Right)** Once finishing the MLP forward pass, the tile's threads proceed to alpha blend and move to the next MLP.

Recent work has brought open-vocabulary, open-set, and prompt-driven semantics into 3D and neural rendering by lifting foundation-model signals into 3D-consistent fields. OpenScene [28] predicts dense 3D features aligned with text and image embeddings to support open-vocabulary 3D understanding. LERF [13] embeds language features into radiance fields by supervising multi-view-consistent CLIP embeddings along rays, enabling open-ended language queries in 3D. GARField [14] learns hierarchical 3D groups from SAM masks for open-ended and interactive scene-level grouping. In the Gaussian-splatting family, LangSplat [29] constructs a 3D language field over Gaussians to support efficient open-vocabulary querying. Unlike these methods, we represent scenes with independent MLPs, each designed to represent an object or part of it, so semantics attach directly to the representation's native granularity.

3 MLP-Splatting

We represent a scene as a collection of independent primitives, each modeled by a compact MLP that predicts radiance and opacity within a local region. Altogether, these primitives form a piecewise-continuous approximation of the light-field. Given a camera ray, rendering reduces to evaluating the interaction between the ray and primitives in the vicinity, followed by standard volumetric compositing.

3.1 Neural Primitive Representation

Each primitive i is parameterized by a center $\boldsymbol{\mu}_i \in \mathbb{R}^3$, a scale vector $\mathbf{s}_i \in \mathbb{R}^3$, and a rotation represented by a unit quaternion $\mathbf{q}_i \in S^3$, where S^3 denotes the

unit 3-sphere in $\mathbb{R}^4$. The scale and rotation together define an anisotropic Gaussian support region that models a soft spatial boundary in $\mathbb{R}^3$. Associated with each primitive is an independent neural function $f_i(\cdot;\theta_i)$ mapping a 6D input to $(\mathbf{c}_i, \sigma_i)$, where $\mathbf{c}_i \in [0,1]^3$ denotes RGB radiance and $\sigma_i \geq 0$ denotes opacity density. No parameters are shared across primitives, allowing discontinuities between objects while preserving intra-object continuity.

3.2 Ray Modeling and Interaction with Primitives

Let a camera ray be defined as $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, where $\mathbf{o}$ is the origin and $\mathbf{d}$ is the normalized direction. For the primitive i, we compute an *optimal soft contact point* by minimizing a Gaussian-weighted distance from the ray to the primitive center:

$$t_i^\star = \arg\min_{t \geq 0} \ \left\| \mathbf{o} + t\mathbf{d} - \boldsymbol{\mu}_i \right\|_{\boldsymbol{\Lambda}_i}^2 , \tag{1}$$

where $\boldsymbol{\Lambda}_i = \mathbf{R}(\mathbf{q}_i)\,\mathrm{diag}(s_{ix}^2, s_{iy}^2, s_{iz}^2)^{-1}\,\mathbf{R}(\mathbf{q}_i)^\top \in \mathbb{R}^{3\times 3}$ ($\mathbf{R}(\cdot)$ is a rotation matrix of a given quaternion) is a positive definite matrix induced by the primitive's anisotropic Gaussian support. This objective admits a closed-form solution:

$$t_i^\star \ = \ \max\!\left(0,\ \frac{\mathbf{d}^\top \boldsymbol{\Lambda}_i (\boldsymbol{\mu}_i - \mathbf{o})}{\mathbf{d}^\top \boldsymbol{\Lambda}_i \mathbf{d}}\right), \qquad \mathbf{p}_i^\star \ = \ \mathbf{o} + t_i^\star \mathbf{d}. \tag{2}$$

We define the residual vector from the soft contact point to the primitive center as:

$$\boldsymbol{\delta}_i \ = \ \boldsymbol{\mu}_i - \mathbf{p}_i^\star, \tag{3}$$

which, together with the ray direction, establishes a one-to-one mapping between viewing configuration and primitive-local spatial interaction. The 6D geometric input to the primitive MLP is constructed as:

$$\mathbf{x}_i = [\mathbf{d}\,;\,\boldsymbol{\delta}_i] \in \mathbb{R}^6, \tag{4}$$

coupling directional dependence (viewing direction) with local geometric context at the contact location. This directly connects our formulation to the light-field parameterization, in which radiance depends jointly on viewing direction and spatial location, expressed here in primitive-centered coordinates.

To enhance high-frequency expressivity, we apply multi-resolution positional encoding to each scalar component of $\mathbf{x}_i$. For a scalar p, the encoding is defined as:

$$\gamma(p) = \left(\sin(2^0\pi p), \cos(2^0\pi p), \ldots, \sin(2^{L-1}\pi p), \cos(2^{L-1}\pi p)\right), \tag{5}$$

and extended component-wise to vectors. The encoded feature $\gamma(\mathbf{x}_i)$ is then used as the actual input to the primitive MLP.

The primitive predicts raw outputs $(\tilde{\mathbf{c}}_i, \tilde{\sigma}_i) = f_i(\gamma(\mathbf{x}_i))$, which are mapped to valid ranges via $\mathbf{c}_i = \mathrm{sigmoid}(\tilde{\mathbf{c}}_i)$ and $\sigma_i = \mathrm{softplus}(\tilde{\sigma}_i)$. To enforce spatial locality, opacity is further modulated by a Gaussian falloff induced by the primitive's anisotropic support.

3.3 Rendering and Compositing

Sorting by Eq. (2) leads to a geometrically correct order of primitives, requiring the thread responsible for each pixel to independently sort all primitives whose support region splatted to the image plane intersect the ray for the pixel. As an alternative for efficiency (see Fig.2), we can sort with a $t^*_{i,c}$ obtained by solving Eq. (1) with $\mathbf{d}_c$, which is a virtual ray direction passing through the center of the tile the pixel belongs to. The pixel plane is divided into square tiles, as in [12].

For a ray intersecting N primitives sorted by increasing $t^\star_i$, the rendered color is computed by front-to-back compositing:

$$\mathbf{C} = \sum_{i=1}^{N} T_i \alpha_i \mathbf{c}_i, \tag{6}$$

with $\alpha_i = 1 - \exp(-\sigma_i \ell_i)$, $T_i = \exp\Big(-\sum_{j=1}^{i-1} \sigma_j \ell_j\Big)$, where ℓ_i denotes the effective integration length associated with the i-th ray–primitive interaction.

This formulation resembles not only alpha blending of 3DGS, but also the discrete approximation of the volume rendering integral used in NeRF, while replacing dense sampling with sparse per-primitive interactions.

To enable efficient full-resolution rendering, primitives are rasterized into screen-space tiles using conservative overlap tests based on their projected support regions. Within each tile, only spatially relevant primitives are evaluated, allowing scalable rendering without global ray marching.

3.4 Feature Embedding

We attach semantic features to each primitive using the same ray–primitive interaction used for radiance. Let $\boldsymbol{\delta}_i = \boldsymbol{\mu}_i - \mathbf{p}^\star_i$ and $\mathbf{u}_i = \mathrm{clip}_{[-1,1]^3}\big(\mathbf{R}(\mathbf{q}_i)^\top \boldsymbol{\delta}_i \oslash \mathbf{s}_i\big)$ be the scale-normalized local offset. This local offset enables regions with similar appearance but different local orientation to be separated in the semantic space, as follows:

$$g_{ik} = \mathrm{softmax}_k\big(\mathbf{a}_k^\top \mathbf{u}_i\big), \qquad \mathbf{f}_i = \sum_{k=1}^{4} g_{ik} \mathbf{e}_{ik}, \tag{7}$$

where $\mathbf{e}_{ik} \in \mathbb{R}^{C_f}$ are learnable slot embeddings, and $\mathbf{a}_k$ are directional conditioning defined by a canonical tetrahedral frame $\mathcal{A} = \{\mathbf{a}_k\}_{k=1}^4 \subset \mathbb{S}^2$. This is the minimal affinely independent basis in 3D and yields an isotropic angular partition. This per-primitive feature embedding is rendered with the same volumetric weights as color: $\mathbf{F} = \sum_{i=1}^{N} T_i \alpha_i \, \mathbf{f}_i$.

3.5 Optimization

Given ground-truth image I with color channels $\mathbf{C}^*$, we supervise rendered colors $\mathbf{C}$ using a photometric objective,

$$\mathcal{L}_{rgb} = (1-\lambda)\,\|\mathbf{C} - \mathbf{C}^*\|_1 + \lambda\,\big(1 - \mathrm{SSIM}(\mathbf{C}, \mathbf{C}^*)\big), \tag{8}$$

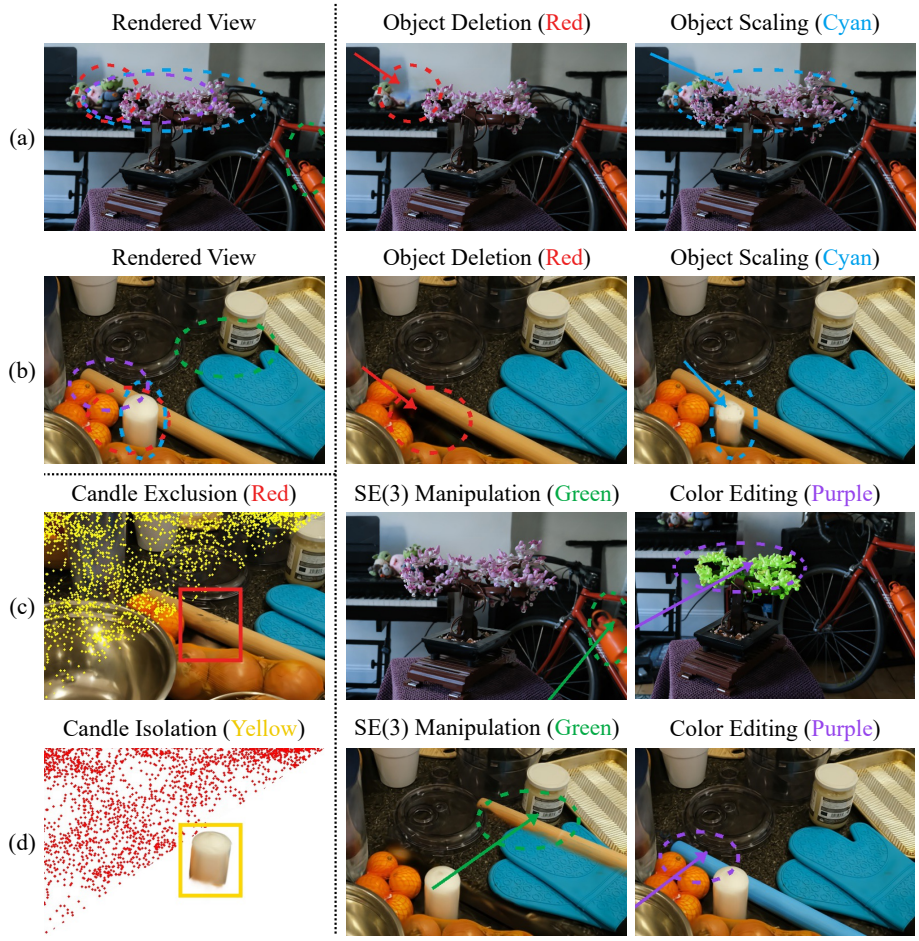


Fig. 3: Interactive scene editing demonstrations. Row (a) and (b) show object scaling and deletion demonstrated on a bonsai tree and a candle. Leftmost column of row (c) and (d) shows exclusion and isolation of the candle. Rest parts of the row (c) and (d) show color editing and rigid body transformations applied to the bonsai tree's and the rolling pin's primitives, and how they are rendered from the new location.

which balances per-pixel fidelity and structural similarity [12]. When jointly training semantics, we align the rendered feature map with the latent embedding from a vision foundation model:

$$\mathcal{L} = \mathcal{L}_{rgb} + \mathcal{L}_f, \qquad \mathcal{L}_f = \|\mathbf{F} - \mathbf{F}_t\|_1, \tag{9}$$

where $\mathbf{F}_t$ denotes the teacher latent embedding of image I.

4 Experiments

4.1 Implementation Details

The 6D input is positional encoded into a 72-dimensional feature vector. Each MLP architecture is 72–32–32–4, producing RGB color and raw density. The feature embedding has a dimension of 512, and its rendered feature is directly guided by 2D semantic features. We implement a custom tile-based CUDA renderer with a 8×8 tile size, which is critical for efficient rendering. For each tile, we first perform an AABB–tile-frustum intersection test to identify candidate primitives, and then apply a block-level radix sort using their depths along a representative virtual ray passing through the tile center. This virtual ray is used only to determine the primitive ordering within the tile and is independent of actual per-pixel rendering rays.

We initialize the MLP primitives using an octree-based sampling of the COLMAP [32] point cloud, ensuring sufficient spatial coverage and geometric fidelity. During training, the primitives in the vicinity naturally compete with each other: some specialize in representing parts of objects, while others gradually vanish as their scale and opacity approach zero. Exploiting this property to keep the proposed method simple to analyze, no explicit density control mechanism is employed.

Optimization is done via the Adam optimizer [40] with learning rates of 10^{-2} for log-scales, 10^{-4} for positions, and 10^{-3} for all other parameters. The model is trained for 60,000 iterations with a single NVIDIA RTX 6000 Ada GPU. All evaluations are conducted on an NVIDIA RTX 4090 GPU, following the evaluation setting of [43].

Evaluation metrics. We report two groups of metrics following the Replica protocol of Feature-3DGS [43]. **Photometric quality** is evaluated on held-out novel views using PSNR, SSIM, and LPIPS [12]. **Semantic quality** is evaluated by rendering per-pixel semantic predictions and computing mean intersection-over-union (mIoU) and pixel accuracy: $\mathrm{IoU}_c = \frac{\mathrm{TP}_c}{\mathrm{TP}_c+\mathrm{FP}_c+\mathrm{FN}_c}$, $\mathrm{mIoU} = \frac{1}{|\mathcal{C}|}\sum_{c\in\mathcal{C}} \mathrm{IoU}_c$, and $\mathrm{Acc} = \frac{1}{|\Omega|}\sum_{p\in\Omega} \mathbf{1}[\hat{y}_p = y_p]$, where $\mathcal{C}$ is the evaluated label set and Ω denotes image pixels.

Baselines. We compare against (i) NeRF-DFF [16], which distills CLIP-LSeg features into a NeRF-style feature field, and (ii) Feature-3DGS [43], which distills the same teacher features into 3D Gaussian primitives with an efficient rasterization pipeline. Both baselines are evaluated on the Replica [34] dataset, and Feature-3DGS is evaluated on the ScanNet [6] dataset, where sequences are sampled following [43].

4.2 MLP as an Implicit Primitive for Explicit Representation

Although the primary goal of MLP-Splatting is to construct an object-centric scene representation, it retains strong photometric rendering performance. De-

Table 1: Novel-view synthesis evaluation on Replica and ScanNet datasets.

Dataset	Replica		ScanNet	
Metric|Method	Feature-3DGS	Ours	Feature-3DGS	Ours
PSNR ↑	36.18	**36.25**	23.32	**25.35**
SSIM ↑	0.964	**0.971**	0.817	**0.830**
LPIPS ↓	**0.079**	0.090	**0.362**	0.403

Table 2: Top-5 object counts per scene. MLP and 3DGS are shown side-by-side for direct comparison.

Scene	floor		rug		table		chair		bag		wall		Total	
	MLP	3DGS	MLP	3DGS	MLP	3DGS	MLP	3DGS	MLP	3DGS	MLP	3DGS	MLP	3DGS
room0	253	48490	485	21621	511	98197	782	43149	76	17484	-	-	2107	228941
room1	91	21525	116	32751	1961	332611	-	-	44	10389	428	140025	2640	537301
office3	174	50671	-	-	198	24242	290	36867	49	9102	351	62102	1062	182984
office4	171	29803	-	-	178	35771	203	49684	38	9425	296	51922	886	176605

spite decomposing the scene into independent neural primitives, the resulting model achieves faithful view synthesis comparable to scene-centric approaches.

Statistics shown in Tab. 1 illustrate that our formulation preserves high-frequency appearance details while maintaining spatial coherence across viewpoints. Through a comparison of the modeling behaviors of 3DGS and MLP-Splatting, as illustrated in the second column of Fig. 4, we observe clear differences. 3DGS employs a large number of Gaussian ellipsoids to represent the fine structures of the blinds, yet the learned result ultimately collapses into blurry, coarse regions. In contrast, MLP-Splatting successfully represents each slat with one or a few elongated MLP primitives, producing the correct striped structure. This demonstrates that modeling objects as independent functional units does not compromise photometric fidelity; rather, it offers a structurally aligned alternative to monolithic scene-wide fields.

We further analyze the joint training of appearance and semantics. In Tab. 2, we present a side-by-side comparison between MLP-Splatting and Feature-3DGS in terms of the parameter count required to represent the same object in the same scene. For each scene, we select the top-5 object groups (covering about 95% of all primitives), and estimate their counts using a softmax-based threshold ($\tau = 0,15$). We observe that MLP-Splatting typically requires fewer than 1% of the Gaussian ellipsoids required by Feature-3DGS. A single MLP has fewer parameters than ten semantic Gaussian ellipsoids. This demonstrates an order-of-magnitude gap in representational capacity.

4.3 Photometry-Driven Interactive Scene Editing

Notably, even when trained solely with RGB supervision, MLP-Splatting exhibits emergent "object-level" structure. Without any segmentation masks, language priors, or identity supervision, each primitive naturally specializes to represent a single object or a coherent part of an object. For example, in the lower-left

of Fig. 3, the candle object is jointly represented by 11 MLP primitives. They are largely distributed along the edges of the cylindrical structure. By selecting these primitives and masking them out, the object can be removed from the scene.

We attribute this emergent behavior to the inductive bias of our representation. Since primitives are independent functional predictors with localized spatial support, optimization favors configurations in which each MLP exclusively models a coherent and confined subset of the scene. As a result, semantic grouping naturally emerges from photometric reconstruction alone.

This emergent object-level decomposition enables interactive scene editing directly at the primitive level with ease. By manipulating or removing selected primitives, users can edit individual objects without retraining or auxiliary segmentation models. Fig. 3 demonstrates object removal, recoloring, and partial editing achieved purely through primitive-level operations.

The mathematical implementation is tightly coupled with the MLP primitives. Each primitive has an explicit spatial position and an orientation parameterized by a quaternion. Therefore, rigid-body manipulation can be implemented by applying an SE(3) transformation to the poses of a single primitive or a selected group of primitives, updating their positions and orientations consistently. Furthermore, object-level scaling can be achieved by modifying the scale parameters of primitives.

At a deeper level, we can directly operate on the MLP's parameters. In particular, global color and opacity adjustments can be implemented by modifying the bias of the final layer. The color transformations shown in Fig. 3 are obtained by adding small offsets along target color directions in the last bias vectors of the MLPs. More fine-grained texture manipulation can be achieved by post-optimizing the internal MLP parameters, similar to DFF [16].

4.4 Multi-view Consistent 3D Segmentation

As discussed in the previous section, the proposed MLP primitives enable easy editing on objects, stemming from the property that each learned radiance field is typically constrained to object boundaries. To better demonstrate this characteristic, arising from the combination of both explicit and implicit scene representation properties, we augment our primitives with semantic embeddings and conduct language-guided editing and segment-anything experiments in the style of [43]. Through experiments in this section, we show that MLPs can still effectively preserve fine-grained semantic details despite covering volumes that are significantly larger than those of conventional primitives.

Language-guided Segmentation We follow the approach of [43], which distills the feature map of image encoder of LSeg-CLIP into embeddings that are independent for each primitive, and learns to represent semantic feature field. These independent embeddings are trained jointly with our primitives under supervision from the feature map and images.

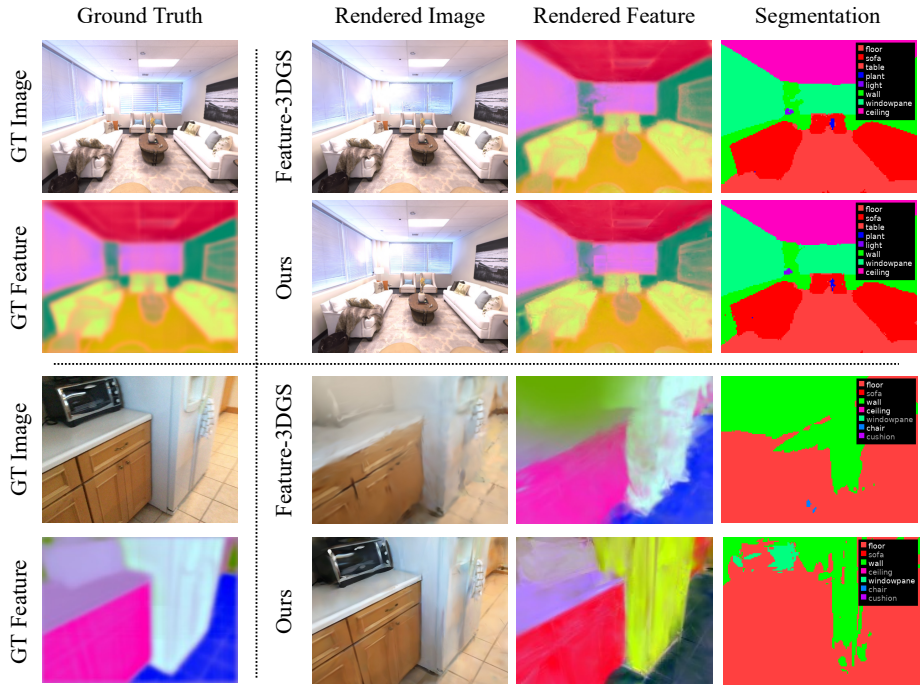


Fig. 4: 3D semantic segmentation with LSeg-guided embeddings rendered on novel views. The top two rows are from the Replica dataset [34], and the bottom two rows are from the ScanNet [6] dataset.

As shown in Tab. 3, evaluated under the protocol proposed in [16, 43], our method outperforms [16] across all semantic segmentation metrics, despite each primitive modeling a light-field-like radiance field, and performs competitively with Feature 3DGS. This strong performance reflects the explicit representation property of our approach: by decomposing the scene into multiple primitives that partition space, our representation encourages semantic regions to better align with object boundaries.

At the same time, our method is significantly more efficient than [43] in both memory usage and rendering speed, demonstrating the high compression capability of neural scene representations. The speed improvement arises because this high compression allows the scene to be represented with fewer primitives, thereby effectively avoiding the large alpha-blending overhead of numerous semantic embeddings encountered by [43].

Furthermore, as illustrated in Fig. 4, our method not only is quantitatively performant, but also qualitatively produces semantic segmentation results that are comparable to [43].

Segment Anything In Fig. 5, we perform instance segmentation across multiple views and scenes using our primitives and embeddings trained with the

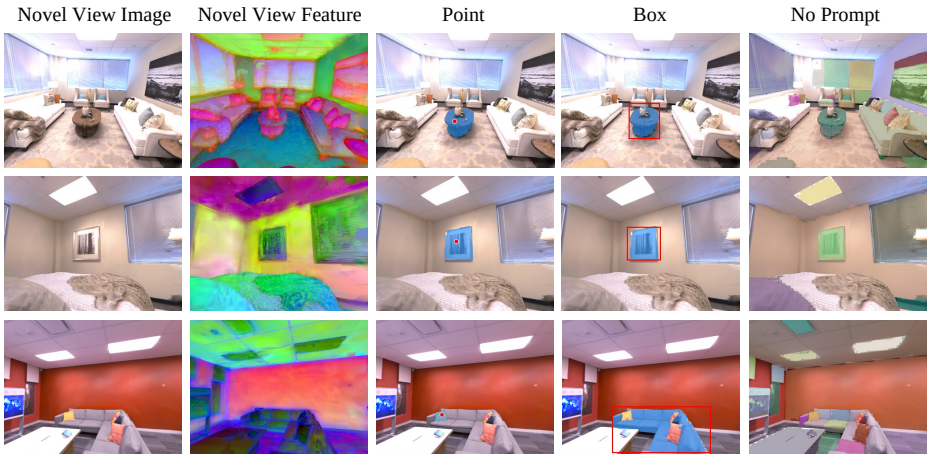


Fig. 5: Novel view semantic segmentation with SAM embeddings.

Table 3: Semantic and efficiency comparison on Replica and ScanNet datasets.

Dataset	Replica			ScanNet	
Metric|Method	Nerf-DFF	Feature-3DGS	Ours	Feature-3DGS	Ours
mIoU (%)↑	63.6	**78.9**	77.7	60.0	**64.3**
Accuracy (%)↑	86.4	**94.3**	93.9	87.0	**89.4**
Memory (MB)↓	86.5	629.7	**32.0**	1120	**206**
FPS↑	< 3	$\sim$3	$\sim$**8**	< 1	$\sim$**12**

embedding guidance of [15], following the protocol of [43]. For either the point or the box prompt, it is shown that the obtained instance segmentation has a clear boundary and covers most of our intended objects, demonstrating that the 2D feature field rendered from our per-primitive semantic embedding is not only well aligned with the SAM feature from which it is guided, but also multi-view consistent, given that Fig. 5 is rendered from a novel viewpoint.

5 Discussion

Lastly, we compare MLP-Splatting and Gaussian Splatting at the level of *density field approximation under volumetric rendering*. We found that Gaussian splatting is fundamentally limited by geometric surface coverage: analytic primitives cannot reproduce sharp discontinuities without allocating $\Theta(h^{-2})$ components along object boundaries. In contrast, MLP-Splatting performs local functional approximation within smooth regions and preserves discontinuities via partition-of-unity, achieving optimal Sobolev rates [33, 38].

Our analysis is concentrated into a theorem we call the Equal-Quality Memory Law. This theorem not only explains the bounds on the number of parameters

but also indicates why our representation can naturally learn object-level entities, whereas Gaussian Slatting struggles. We briefly illustrate the mathematical formulation (Sec. 5.1) and list important theorems (Sec. 5.2) while leaving the proofs for the supplementary material. For ease of exposition, we do not use bold roman notation for vectors in this section.

5.1 Function-Space Perspective

Let $\sigma : \mathbb{R}^3 \to \mathbb{R}_{\geq 0}$ denote the volume density and $c : \mathbb{R}^3 \times \mathbb{S}^2 \to \mathbb{R}^3$ the color field. Given a ray $r(t) = o + td$, volumetric rendering produces a pixel value $\mathcal{R}_r[\sigma, c] = \int_0^L T_\sigma(t)\, \sigma(r(t))\, c(r(t), d)\, dt$, where $T_\sigma(t) = \exp\Big(-\int_0^t \sigma(r(s))\, ds\Big)$ is the accumulated transmittance. Given a set of rays $\mathcal{U}$, we measure pixel error by

$$\mathcal{E}_p((\sigma, c), (\tilde{\sigma}, \tilde{c})) = \left(\frac{1}{|\mathcal{U}|} \sum_{r \in \mathcal{U}} \|\mathcal{R}_r[\sigma, c] - \mathcal{R}_r[\tilde{\sigma}, \tilde{c}]\|^p \right)^{1/p}. \tag{10}$$

Model classes. Gaussian splatting constructs density as a K-term Gaussian mixture $\sigma_K(x) = \sum_{k=1}^K a_k \exp(-\frac{1}{2}(x - \mu_k)^\top \Sigma_k^{-1}(x - \mu_k))$, with x 3D position whereas MLP-Splatting represents density as a sum of neural primitives $\sigma_N(x) = \sum_{i=1}^N \rho(f_i(x, \theta_i))$, with 6D input (x), width-m networks (f_i) and ρ enforcing nonnegativity.

The following shows how many parameters are required to approximate a piecewise-smooth target scene (σ^*, c^*) up to pixel error ε.

5.2 Main Theoretical Results

Theorem 1 (Lower Bound for Gaussian Splatting). *For any nonnegative K-Gaussian mixture σ_K,*

$$\inf_{\sigma_K} \mathcal{E}_1\big((\sigma^*, c^*), (\sigma_K, c^*)\big) \gtrsim CK^{-1/2}. \tag{11}$$

The proof combines the analyticity of Gaussian mixtures, a geometric tube argument near the discontinuity set, and stability of the rendering operator.

Theorem 2 (Upper Bound for MLP-Splatting). *Assume (σ^*, c^*) is piecewise C^s with jump set Γ. There exists a partition-of-unity construction such that*

$$\mathcal{E}_2\big((\sigma^*, c^*), (\sigma_N, c_N)\big) \leq C\big(h^s + m^{-s/3}\big), \tag{12}$$

where h is patch size and m network width. Balancing the terms yields parameter cost $\Theta(\varepsilon^{-3/s})$ to achieve pixel error ε.

The proof combines local neural approximation rates, geometric partitioning along Γ, and stability of volumetric rendering.

Corollary 1 (Equal-Quality Memory Law). *Given pixel error ε in $d = 3$ we aim to achieve,*

- *Gaussian splatting requires $\Omega(\varepsilon^{-2})$ parameters;*
- *MLP-Splatting admits a construction with $\Theta(\varepsilon^{-3/s})$ parameters.*

For $s \geq 2$ (i.e., when the target scene is piecewise twice differentiable), $\varepsilon^{-3/s} \leq \varepsilon^{-1.5}$, strictly improving the Gaussian rate.

While real-world scenes are globally non-smooth, they often exhibit smooth structure piecewise: discontinuities are concentrated near object boundaries, whereas the fields within each region remain locally regular. This suggests that, in practice, MLP-Splatting can achieve a comparable or lower rendering error with a comparable or smaller number of parameters than 3DGS.

6 Limitations

We do not explicitly design a density control strategy for MLP-Splatting. The positional-gradient–based densification used in 3DGS does not transfer well, as the gradients are often weak and noisy. Moreover, MLP primitives do not expose an explicit opacity parameter for pruning.

However, under RGB supervision we observe that primitives in regions with consistent appearance tend to rapidly expand their scales, causing neighboring primitives to become effectively inactive, while primitives in appearance-complex regions exhibit parameter oscillation to fit details. Both behaviors can reduce efficiency or reconstruction quality, suggesting that a suitable density control mechanism could further improve the framework.

7 Conclusion

We introduced *MLP-Splatting*, a neural scene representation in which each primitive is modeled as an independent compact MLP. Unlike existing approaches built on geometric primitives, our formulation treats neural predictors themselves as the fundamental units of the scene representation.

This design encourages primitives to specialize to coherent scene regions, leading to emergent object-level structure under RGB supervision and enabling straightforward integration with 2D semantic feature distillation. We show that compact neural primitives can represent scenes with strong photometric fidelity, competitive semantics, and substantially improved efficiency. These results suggest an alternative paradigm for neural scene representations. We hope this work motivates a shift from attaching semantics to low-level primitives toward designing representations whose inductive bias makes object-level structure emerge as a first-class property.

Acknowledgements

This research was supported by the EPSRC grant EP/Y020499/1 and the ARIA seed project NACB-SE02-P05. We appreciate all members of the Robot Vision Group for insightful and valuable discussions.

References

1. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In: Int. Conf. Comput. Vis. pp. 5855–5864 (October 2021)
2. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5470–5479 (2022)
3. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Zip-nerf: Anti-aliased grid-based neural radiance fields. In: Int. Conf. Comput. Vis. pp. 19697–19705 (2023)
4. Cen, J., Zhou, X., Fang, J., Wen, C., Xie, L., Zhang, X., Shen, W., Tian, Q.: Tackling view-dependent semantics in 3d language gaussian splatting. In: Int. Conf. Mach. Learn. (2025)
5. Cheng, Y., Jiao, J., Wang, Y., Kanoulas, D.: Logs: Visual localization via gaussian splatting with fewer training images. In: 2025 IEEE International Conference on Robotics and Automation (ICRA). pp. 15029–15036. IEEE (2025)
6. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: ScanNet: Richly-annotated 3d reconstructions of indoor scenes. In: IEEE Conf. Comput. Vis. Pattern Recog. (July 2017)
7. Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5501–5510 (June 2022)
8. Fry, N., Dexheimer, E., Mazur, K., Kelly, P.H., Davison, A.J.: Triangle splatting slam. arXiv preprint arXiv:2605.31419 (2026)
9. Govindarajan, S., Rebain, D., Yi, K.M., Tagliasacchi, A.: Radiant foam: Real-time differentiable ray tracing. In: Int. Conf. Comput. Vis. pp. 4135–4145 (2025)
10. Guo, M., Fathi, A., Wu, J., Funkhouser, T.: Object-centric neural scene rendering. arXiv preprint arXiv:2012.08503 (2020)
11. Huang, B., Yu, Z., Chen, A., Geiger, A., Gao, S.: 2d gaussian splatting for geometrically accurate radiance fields. In: ACM SIGGRAPH. pp. 1–11 (2024)
12. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G., et al.: 3d gaussian splatting for real-time radiance field rendering. ACM Trans. Graph. **42**(4), 139–1 (2023)
13. Kerr, J., Kim, C.M., Goldberg, K., Kanazawa, A., Tancik, M.: Lerf: Language embedded radiance fields. In: Int. Conf. Comput. Vis. pp. 19729–19739 (2023)
14. Kim, C.M., Wu, M., Kerr, J., Goldberg, K., Tancik, M., Kanazawa, A.: Garfield: Group anything with radiance fields. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 21530–21539 (2024)
15. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. In: Int. Conf. Comput. Vis. pp. 4015–4026 (2023)
16. Kobayashi, S., Matsumoto, E., Sitzmann, V.: Decomposing nerf for editing via feature field distillation. Adv. Neural Inform. Process. Syst. **35**, 23311–23330 (2022)
17. Kong, X., Liu, S., Taher, M., Davison, A.J.: vmap: Vectorised object mapping for neural field slam. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 952–961 (2023)
18. Kundu, A., Genova, K., Yin, X., Fathi, A., Pantofaru, C., Guibas, L.J., Tagliasacchi, A., Dellaert, F., Funkhouser, T.: Panoptic neural fields: A semantic object-aware neural scene representation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 12871–12881 (2022)

19. Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Jiang, Q., Li, C., Yang, J., Su, H., et al.: Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In: Eur. Conf. Comput. Vis. pp. 38–55. Springer (2024)
20. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. In: Seminal graphics: pioneering efforts that shaped the field, pp. 347–353. ACM New York, NY, USA (1998)
21. Lüddecke, T., Ecker, A.: Image segmentation using text and image prompts. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7086–7096 (2022)
22. von Lützow, N., Nießner, M.: Linprim: Linear primitives for differentiable volumetric rendering. In: Adv. Neural Inform. Process. Syst. (2025)
23. Mai, A., Hedman, P., Kopanas, G., Verbin, D., Futschik, D., Xu, Q., Kuester, F., Barron, J.T., Zhang, Y.: Ever: Exact volumetric ellipsoid rendering for real-time view synthesis. In: Int. Conf. Comput. Vis. pp. 4930–4939 (2025)
24. Martin-Brualla, R., Radwan, N., Sajjadi, M.S.M., Barron, J.T., Dosovitskiy, A., Duckworth, D.: Nerf in the wild: Neural radiance fields for unconstrained photo collections. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7210–7219 (June 2021)
25. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: Eur. Conf. Comput. Vis. (2020)
26. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph. **41**(4), 1–15 (2022)
27. Niemeyer, M., Geiger, A.: Giraffe: Representing scenes as compositional generative neural feature fields. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 11453–11464 (June 2021)
28. Peng, S., Genova, K., Jiang, C., Tagliasacchi, A., Pollefeys, M., Funkhouser, T., et al.: Openscene: 3d scene understanding with open vocabularies. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 815–824 (2023)
29. Qin, M., Li, W., Zhou, J., Wang, H., Pfister, H.: Langsplat: 3d language gaussian splatting. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 20051–20060 (2024)
30. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: Int. Conf. Mach. Learn. pp. 8748–8763. PmLR (2021)
31. Reiser, C., Peng, S., Liao, Y., Geiger, A.: Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In: Int. Conf. Comput. Vis. pp. 14335–14345 (October 2021)
32. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: IEEE Conf. Comput. Vis. Pattern Recog. (2016)
33. Siegel, J.W., Xu, J.: Sharp bounds on the approximation rates, metric entropy, and n-widths of shallow neural networks. Foundations of Computational Mathematics **24**(2), 481–537 (2024)
34. Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J.J., Mur-Artal, R., Ren, C., Verma, S., et al.: The replica dataset: A digital replica of indoor spaces. arXiv preprint arXiv:1906.05797 (2019)
35. Taher, M., Alzugaray, I., Davison, A.J.: Fit-ngp: Fitting object models to neural graphics primitives. In: IEEE Int. Conf. Robot. Autom. pp. 18186–18192. IEEE (2024)
36. Wu, Q., Wang, K., Li, K., Zheng, J., Cai, J.: Objectsdf++: Improved object-compositional neural implicit surfaces. In: Int. Conf. Comput. Vis. pp. 21764–21774 (October 2023)

37. Yang, B., Zhang, Y., Xu, Y., Li, Y., Zhou, H., Bao, H., Zhang, G., Cui, Z.: Learning object-compositional neural radiance field for editable scene rendering. In: Int. Conf. Comput. Vis. pp. 13779–13788 (October 2021)
38. Yarotsky, D.: Error bounds for approximations with deep relu networks. Neural networks **94**, 103–114 (2017)
39. Ye, M., Danelljan, M., Yu, F., Ke, L.: Gaussian grouping: Segment and edit anything in 3d scenes. In: Eur. Conf. Comput. Vis. pp. 162–179. Springer (2024)
40. Zhang, Z.: Improved adam optimizer for deep neural networks. In: 2018 IEEE/ACM 26th international symposium on quality of service (IWQoS). pp. 1–2. Ieee (2018)
41. Zhi, S., Laidlow, T., Leutenegger, S., Davison, A.J.: In-place scene labelling and understanding with implicit scene representation. In: Int. Conf. Comput. Vis. pp. 15838–15847 (2021)
42. Zhi, S., Sucar, E., Mouton, A., Haughton, I., Laidlow, T., Davison, A.J.: ilabel: Revealing objects in neural fields. IEEE Robot. and Auto. Letters (RAL) **8**(2), 832–839 (2022)
43. Zhou, S., Chang, H., Jiang, S., Fan, Z., Zhu, Z., Xu, D., Chari, P., You, S., Wang, Z., Kadambi, A.: Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 21676–21685 (2024)
44. Zhou, X., Nguyen, B.H., Magne, L., Golyanik, V., Leimkühler, T., Theobalt, C.: Splat the net: Radiance fields with splattable neural primitives. In: Int. Conf. Learn. Represent. (2026)