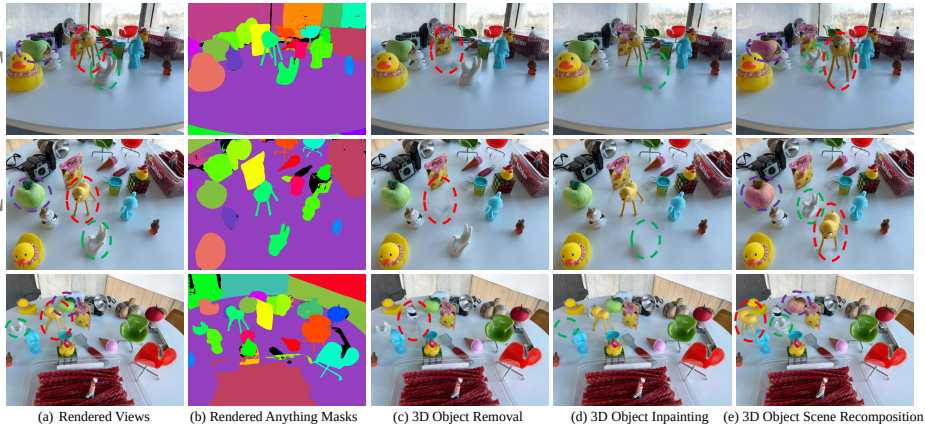
Gaussian Grouping: Segment and Edit Anything in 3D Scenes

Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke ♠

Computer Vision Lab, ETH Zurich



(a) Rendered Views (original scene) (b) Rendered Anything Masks (c) 3D Object Removal (red circle) (d) 3D Object Inpainting (green circle) (e) 3D Object Scene Recomposition (exchange red and green circle, colorize purple circle)

Fig. 1: Our Gaussian Grouping jointly reconstructs (column a) and segments (column b) anything in full open-world 3D scenes, with fine-grained instance and stuff level modeling. This enables versatile scene editing applications, such as 3D object removal (column c), 3D object inpainting (column d, which first removes the 3D object and then inpaints the holes) and scene re-composition and object colorization (column e). Since the segmentation information is encapsulated in the 3D Gaussians, editing tasks such as 3D object removal, colorization and object location exchange can be performed directly **w/o** training, while inpainting only requires minutes of fine-tuning.

Abstract. The recent Gaussian Splatting achieves high-quality and real-time novel-view synthesis of the 3D scenes. However, it is solely concentrated on the appearance and geometry modeling, while lacking in fine-grained object-level scene understanding. To address this issue, we propose Gaussian Grouping, which extends Gaussian Splatting to jointly reconstruct and segment anything in open-world 3D scenes. We augment each Gaussian with a compact Identity Encoding, allowing the Gaussians to be grouped according to their object instance or stuff membership in the 3D scene. Instead of resorting to expensive 3D labels, we supervise the Identity Encodings during the differentiable rendering by leveraging the 2D mask predictions by Segment Anything Model (SAM), along with introduced 3D spatial consistency regularization. Compared to the implicit NeRF representation, we show that the discrete and grouped 3D Gaussians can reconstruct, segment and edit anything in 3D with high visual quality, fine granularity and efficiency. Based on Gaussian Grouping, we further propose a local Gaussian Editing scheme, which shows

♠ Project Lead

efficacy in versatile scene editing applications, including 3D object removal, inpainting, colorization, style transfer and scene recomposition. Our code and models are at github.com/lkeab/gaussian-grouping.

Keywords: 3D Segment Anything · Scene Editing · Gaussian Splatting

1 Introduction

Open-world 3D scene understanding is an essential challenge, with far-reaching implications for robotics, AR / VR, and autonomous driving. Given a set of posed RGB images, our goal is to learn an effective 3D representation that jointly reconstructs and segments anything in the 3D scene. The representation should easily support a wide range of downstream scene editing applications. For example, in Figure 1, the 3D object of the scene can be easily removed or inpainted, and the scene can be recomposed by exchanging object locations.

While there has been remarkable progress in 2D scene understanding brought by SAM and its variants [14, 17, 64], their extension to 3D has been constrained. This is mostly due to the availability and the labor-intensive process of creating 3D scene datasets. Existing methods [8, 42] rely on manually-labeled datasets, which are both costly and limited in scope, or require accurately scanned point clouds [37, 47] as input. This hinders the development of the 3D scene understanding methods that can quickly generalize across various real-world scenes.

By taking multi-view captures, existing NeRF-based methods [16, 19, 28, 44] lift 2D masks or distill CLIP [39] / DINO [3] features via neural fields rendering. However, due to the implicit and continuous representation of NeRF, these methods require expensive random sampling and are computationally intensive to optimize. Further, it is hard to directly adjust NeRF-based approaches for the downstream local editing tasks [19], because the learned neural networks, such as MLPs, cannot decompose each part or module in the 3D scene easily. Several methods [4, 43] combine NeRF or stable-diffusion [41] with the masks of SAM, but they only focus on a single object.

Alternative to NeRFs, the recently emerged 3D Gaussian Splatting [15] has shown impressive reconstruction quality with high training and rendering efficiency. It represents the 3D scene with an array of colored and explicit 3D Gaussians, which are rendered into camera views for novel view synthesis. Nevertheless, Gaussian Splatting does not model object instances or semantic understanding. To achieve fine-grained scene understanding, we extend this approach beyond merely capturing the scene's appearance and geometry to include the individual objects and elements that constitute the 3D environments.

We propose *Gaussian Grouping*, which represents the whole 3D scene with a set of grouped 3D Gaussians. By inputting multi-view captures and the corresponding automatically generated masks by SAM, our method learns a discrete and grouped 3D representation for reconstructing and segmenting anything in the 3D scene. Gaussian Grouping inherits SAM's strong zero-shot 2D scene understanding capability and extends it to the 3D space by producing consistent novel view synthesis and segmentation.

To this end, instead of solely modeling the scene appearance and geometry, our approach also captures the identity of each Gaussian of the 3D scene by grouping. A new property, our *Identity Encoding*, is augmented to each Gaussian. The Identity Encoding is a compact and low-dimensional learnable embedding. To leverage the segmentation supervision in 2D, Identity Encoding is trained via differentiable Gaussian rendering, where the encoding vectors of various Gaussians are splatted onto the 2D rendering view. Then, we take the 2D rendered identity features and employ an extra linear layer to classify these splatted embeddings on each 2D location for identity classification.

To further boost the grouping accuracy, besides the standard cross-entropy loss for identity classification, we also introduce an **un**-supervised 3D Regularization Loss by leveraging 3D spatial consistency. The loss enforces the Identity Encodings of the top K-nearest 3D Gaussians to be close in their feature distance. We find it helps Gaussians inside the 3D object or heavily occluded to be supervised during training more sufficiently.

We highlight the advantages of Gaussian Grouping not only by providing high visual quality and fast training, but also through its downstream scene editing applications brought by our dense 3D segmentation. Since our model captures the 3D scenes as compositional structures, each group of 3D Gaussians operates independently, allowing for parts to be fully decoupled or separated. This decomposition is crucial in scenarios where individual components need to be identified, manipulated, or replaced **w/o** disrupting the entire scene structure.

Our contribution can be summarized as follows:

1. We introduce Gaussian Grouping, the first 3D Gaussian Splatting-based segmentation framework that lifts knowledge of SAM to 3D scene anything zero-shot segmentation without the need for 3D mask labels.
2. Gaussian Grouping supports various downstream tasks with our proposed Local Gaussian Editing scheme. Individual components are identified, manipulated, or replaced without disrupting the entire scene structure. By the grouped 3D Gaussians, we show extensive 3D scene editing cases, such as 3D scene re-composition, object inpainting, object removal and object style transfer, with both impressive visual effect and fine granularity.
3. Our training and rendering processes are swift, ensuring compliance with real-time operational requirements.

2 Related Works

3D Gaussian Models 3D Gaussian Splatting [15] has recently emerged as a powerful approach to reconstruct 3D scenes via real-time radiance field rendering. Several follow-up methods [29, 58, 59] extend it to dynamic 3D scenes by tracking of all dense scene elements [29] or deformation field modeling [54, 59]. Another stream of works [6, 48, 60] focuses on the 3D content creation, which combines Gaussian Splatting with diffusion-based models and shows high-quality generation results. However, none of the existing Gaussian Splatting works enables object / stuff-level or semantic understanding of the 3D scene. Our Gaussian Grouping extends Gaussian Splatting from pure scene appearance and ge-

ometry modeling to also support open-world and fine-grained scene understanding. We show that the grouped 3D Gaussians is an effective and flexible 3D representation in supporting a series of downstream scene editing applications.

Radiance-based Open World Scene Understanding Existing semantic scene modeling approaches combined with radiance fields are NeRF-based [31, 33, 66]. Semantic-NeRF [67] initiated the incorporation of semantic information into NeRFs and facilitated the generation of semantic masks from novel views. Building upon it, subsequent research focusing on in-domain scene modeling, has expanded the scope by introducing instance modeling [11, 22, 44, 52] or encoding visual features [19, 40, 49–51] that support semantic delineation. Unlike our approach, most of these methods are designed for in-domain scene modeling and cannot generalize to open-world scenarios.

By distilling 2D features such as CLIP [39] or DINO [3], Distilled Feature Fields [19] explores embedding pixel-aligned feature vectors, and LERF [16] learns language field to help open-world 3D semantics. In contrast to our work, these open-world approaches are only limited in semantic segmentation (hard to separate similar objects of the same category) and cannot produce very accurate segmentation masks as shown in our experiment. To our knowledge, we propose the first Gaussian-based method to tackle open-world 3D scene understanding, where we show the advantages compared to existing NeRF-based approaches [16, 19, 44] in segmentation quality, efficiency and good extension to downstream scene editing applications.

SAM in 3D Segment Anything Model (SAM) [17] was released as a foundational vision model for zero-shot 2D segmentation. Several works lift SAM's 2D masks to 3D segmentation via NeRF [4,5] or 3D point cloud [57]. However, these NeRF-based approaches only focus on a single / few objects of the 3D scene, while our Gaussian Grouping operates in automatic *everything* mode to attain the holistic understanding of each instance / stuff of the full scene.

Radiance-based Scene Editing Manipulating 3D scenes via a radiance field is challenging. For existing NeRF-based scene editing / manipulation works [13, 25, 27, 55, 63], Clip-NeRF [53], Object-NeRF [56], and LaTeRF [35] design approaches to change and complete objects represented by NeRFs; however, their application scenario is limited to simple objects, rather than scenes with significant clutter and texture. Some other works specify bounding boxes [36, 62], to allow flexible compositing of various objects [65] or combining with physical simulation [24]. Most recently, 3D object inpainting is studied in SPIn-NeRF [34] and language-based scene editing is proposed in [12]. Compared to them, we apply our Gaussian Grouping to versatile scene editing tasks, and show benefits brought by the fine-grained scene modeling, where multiple scene editings can be easily superimposed on the same image. The design of Local Gaussian Editing makes the whole training / editing process both simple and efficient.

3 Method

Our work aims to build an expressive 3D scene representation, which not only models appearance and geometry, but also captures every instance and stuff identity of the scene. We design our method based on the recent 3D Gaussian Splatting [15], and extend it from pure 3D reconstruction to fine-grained scene understanding. Our approach, called *Gaussian Grouping*, is capable of: 1) modeling each 3D part of the scene with appearance, geometry together with their mask identities; 2) fully decomposing the 3D scene into discrete *groups*, *e.g.* representing different object instances to enable editing; 3) allowing for fast training and rendering, while not diluting the original 3D reconstruction quality.

Gaussian Grouping effectively leverages the dense 2D mask proposals of SAM, and lifts them to segment anything in the 3D scene via radiance fields rendering. In Sec 3.1, we first briefly review the 3D Gaussian Splatting method on radiance field rendering. We then detail the input data pre-processing steps and further describe the proposed Gaussian Grouping in Section 3.2. With the constructed 3D representation, finally, we show its advantages in downstream scene editing tasks by the efficient Local Gaussian Editing in Section 3.3.

3.1 Preliminaries: 3D Gaussian Splatting

Gaussian Splatting, as introduced by [15], encapsulates 3D scene information using a suite of 3D colored Gaussians. This technique has established its effectiveness in the reconstruction tasks, exhibiting high inference speeds and remarkable quality of reconstruction within timeframes on par with those of NeRF. Yet, its potential in scene understanding has not been thoroughly investigated. Our research reveals that 3D Gaussians hold considerable promise for the open-world and complex 3D scene understanding.

To represent the scene, each Gaussian's property is characterized by a centroid $\mathbf{p} = \{x, y, z\} \in \mathbb{R}^3$, a 3D size $\mathbf{s} \in \mathbb{R}^3$ in standard deviations, and a rotational quaternion $\mathbf{q} \in \mathbb{R}^4$. To allow fast α-blending for rendering, an opacity value $\alpha \in \mathbb{R}$ and a color vector $\mathbf{c}$ are represented in the three degrees of spherical harmonics (SH) coefficients. These adjustable parameters are collectively symbolized by S_{Θ_i}, where $S_{\Theta_i} = \{\mathbf{p}_i, \mathbf{s}_i, \mathbf{q}_i, \alpha_i, \mathbf{c}_i\}$ represents the set of parameters for the i-th Gaussian. Gaussian Splatting projects these 3D Gaussians onto the 2D image plane and implement differentiable rendering for each pixel.

3.2 3D Gaussian Grouping

In this section, we describe the design of our Gaussian Grouping. To enable 3D Gaussians for fine-grained scene understanding, our key insight is that we preserve all attributes of the Gaussians (such as their position, color, opacity, and size) in their original setting, but add new Identity Encoding parameters (similar to the format of color modeling). This allows each Gaussian to be assigned to its represented instances or stuff in the 3D scene.

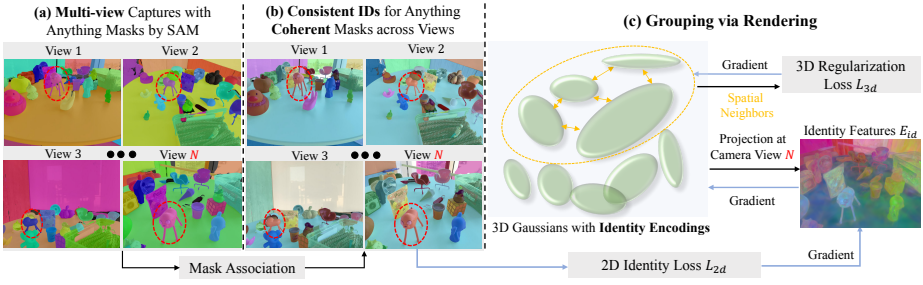


Fig. 2: The method pipeline of our Gaussian Grouping contains three main steps: **(a)** We first prepare the input by deploying SAM to automatically generate masks in *everything* mode for each view independently. **(b)** Then, to obtain the consistent mask IDs across training views, we take a universal temporal propagation model [7] to associate the mask labels and generate a coherent multi-view segmentation. **(c)** With the prepared training input, we jointly learn all properties of the 3D Gaussians, including their group Identity Encoding, by differentiable rendering. Our encoding is supervised by the 2D Identity Loss, leveraging the coherent segmentation views, and a 3D Regularization loss. We use color the denote object IDs across frames for input views. We omit the rendering process for other Gaussian parameters and the density control part for simplicity, as it is inherited from [15].

Algorithm 1 *Gaussian Grouping*

$p \leftarrow$ SfM Points ▷ 3D Positions
$m = (m_1, m_2, \ldots, m_K) \leftarrow$ SAM ▷ **SAM's Masks** at Various K Views
$(\hat{M}_1, \hat{M}_2, \ldots, \hat{M}_K) \leftarrow$ Zero-shot Tracking(m) ▷ Multi-view **Associated Masks**
$s, \alpha, c, \mathbf{e} \leftarrow$ InitAttributes() ▷ Covariances, Opacities, Colors, **Identity Encodings**
$i \leftarrow 0$ ▷ Iteration Count
while not converged **do**
  $V, \hat{I}, \hat{M} \leftarrow$ SampleTrainingView() ▷ Camera View V, Image and **Mask**
  $I, \mathbf{E}_{\text{id}} \leftarrow$ Rasterize(p, s, a, c, $\mathbf{e}$,V) ▷ Rendered Image and **Identity Encoding**
  $\mathcal{L}_{\text{image}} \leftarrow \mathcal{L}(I, \hat{I})$ ▷ Original Image Rendering Loss
  $L_{\text{id}} \leftarrow \lambda_{2d}\mathcal{L}_{2\text{d}}(\mathbf{E}_{\text{id}}, \hat{M}) + \lambda_{3d}\mathcal{L}_{3\text{d}}(\mathbf{e})$ ▷ **Identity Grouping Loss**, Eq. 3
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{image}} + \mathcal{L}_{\text{id}}$ ▷ Total Loss
  p, s, a, c, $\mathbf{e} \leftarrow$ Adam($\nabla\mathcal{L}$) ▷ Backprop & Step
  if IsRefinementIteration(i) **then**
    for all J Gaussians $(p_j, s_j, \alpha_j, c_j, \mathbf{e}_j)$ in $(p, s, a, c, \mathbf{e})$ **do**
      if $\alpha < \epsilon$ or IsTooLarge(p_j, s_j) **then** ▷ Pruning
        RemoveGaussian()
      end if
      if $\nabla_p L > \tau_p$ **then** ▷ Densification
        if $\|S\| > \tau_S$ **then** ▷ Over-reconstruction
          SplitGaussian($p_j, s_j, \alpha_j, c_j, e_j$)
        else ▷ Under-reconstruction
          CloneGaussian($p_j, s_j, \alpha_j, c_j, e_j$)
        end if
      end if
    end for
  end if
  $i \leftarrow i + 1$
end while

The process of Gaussian Grouping is illustrated in Figure 2. We outline the pseudocode for our Gaussian Grouping in Algorithm 1, where we highlight the introduced core components in both red and bold texts.

(a) 2D Image and Mask Input To prepare the input for Gaussian Grouping, in Figure 2(a), we first deploy SAM to automatically generate masks for each image of the multi-view collection. The 2D masks are individually produced per image. Then, to assign each 2D mask a unique ID in the 3D scene, we need to associate the masks of the same identity across different views and obtain the total number K of instances / stuff in the 3D scene.

(b) Identity Consistency across Views Instead of resorting to the cost-based linear assignment [44] during training, we treat the multi-view images of a 3D scene as a video sequence with gradually changing views. To achieve the 2D mask consistency across views, we employ a well-trained zero-shot tracker [7] to propagate and associate masks. This also provides the total number of mask identities in the 3D scene. We visualize associated 2D mask labels in Figure 2(b). Compared to the cost-based linear assignment proposed in [44], we find it simplifies the training difficulty while avoiding repeatedly computing the matching relation in each rendering iteration, resulting in over 60× speeding up. It also achieves better performance than the cost-based linear assignment, especially under the dense and overlapping masks by SAM. Besides, we show the robustness of our 3D masks association in Figure 5, where the 2D associated masks [7] from video contain obvious errors.

(c) 3D Gaussian Rendering and Grouping To generate 3D-consistent mask identities across views of the scene, we propose to group 3D Gaussians belonging to the same instance / stuff. In addition to the existing Gaussian properties, we introduce a new parameter, i.e., Identity Encoding, to each Gaussian. The Identity Encoding is a learnable and compact vector of length 16, which we find is sufficient to distinguish the different objects / parts in the scene with computation efficiency. During training, similar to Spherical Harmonic (SH) coefficients representing the color of each Gaussian, we optimize the introduced Identity Encoding vector to represent its instance ID of the scene. Note that different from the view-dependent appearance modeling of the scene, the instance ID is consistent across various rendering views. Thus, we set the SH degree of the Identity Encoding to 0, to only model its direct-current component. Unlike NeRF-based methods [16,19,44] designing extra semantic MLP layers, the Identity Encoding serves as the learnable property for each Gaussian to group the 3D scene.

To optimize the introduced Identity Encoding of each Gaussian, in Figure 2(c), we render these encoded identity vectors into 2D images in a differentiable manner. We take the differentiable 3D Gaussian renderer from [15] and treat the rendering process similar to the color (SH coefficients) optimization in [15].

3D Gaussian splatting adopts neural point-based α'-rendering [20,21], where the influence weight α' of each Gaussian can be evaluated in 2D for each pixel. Following [15], the influence of all Gaussians on a single pixel location is computed by sorting the Gaussians in depth order and blending $\mathcal{N}$ ordered points overlapping the pixels [30,33]:

$$E_{\text{id}} = \sum_{i \in \mathcal{N}} e_i \alpha'_i \prod_{j=1}^{i-1} (1 - \alpha'_j) \tag{1}$$

where the final rendered 2D mask identity feature E_{id} for each pixel is a weighted sum over the Identity Encoding e_i of length 16 for each Gaussian, weighted by the Gaussian's influence factor α_i' on that pixel. Refer to [61], we compute α_i' by measuring a 2D Gaussian with covariance Σ^{2D} multiplied with a learned per-point opacity α_i, and

$$\Sigma^{\text{2D}} = JW\Sigma^{\text{3D}}W^T J^T \tag{2}$$

where Σ^{3D} is the 3D covariance matrix, Σ^{2D} is the splatted 2D version [68]. J is the Jacobian of the affine approximation of the 3D-2D projection, and W is the world-to-camera transformation matrix.

(d) Grouping Loss After associating 2D instance labels across each training view, suppose there are K masks in total in the 3D scene. To group each 3D Gaussian by the instance /stuff mask identities, we design the grouping loss $\mathcal{L}_{\text{id}}$ for updating the Identity Encoding of Gaussians with two components:

1. ***2D Identity Loss***: Since the mask identity labels are in 2D, instead of directly supervising the Identity Encoding e_i of the 3D Gaussians. Given the rendered 2D features E_{id} in Eq. 1 as input, we first add a linear layer f to recover its feature dimension back to K and then take $softmax(f(E_{\text{id}}))$ for identity classification, where K is the total number of masks in the 3D scene. We adopt a standard cross-entropy loss $\mathcal{L}_{\text{2d}}$ for K categories classification.

2. ***3D Regularization Loss***: To further boost the grouping accuracy of Gaussians, besides the standard cross-entropy loss for indirect 2D supervision, we also introduce an unsupervised 3D Regularization Loss to directly regularize the learning of Identity Encoding e_i. 3D Regularization Loss leverages the 3D spatial consistency, which enforces the Identity Encodings of the top k-nearest 3D Gaussians to be close in their feature distance. This allows the 3D Gaussians inside the 3D object, or heavily occluded (not visible in nearly all training views) during the point-based rendering (Eq. 1) to be supervised more sufficiently. In Eq. 3, we denote F as softmax operation combined after linear layer f (shared in computing the 2D Identity loss). We formalize the KL divergence loss with m sampling points as,

$$\mathcal{L}_{\text{3d}} = \frac{1}{m}\sum_{j=1}^{m} D_{\text{kl}}(P||Q) = \frac{1}{mk}\sum_{j=1}^{m}\sum_{i=1}^{k} F(e_j)\log\left(\frac{F(e_j)}{F(e_i')}\right) \tag{3}$$

where P contains the sampled Identity Encoding e of a 3D Gaussian, while the set $Q = \{e_1', e_2', ..., e_k'\}$ consists of its k nearest neighbors in 3D Euclidean space. We omit the softmax operation combined after linear layer f for simplicity.

Combined with the conventional 3D Gaussian Reconstruction Loss $\mathcal{L}_{\text{rec}}$ on image rendering [15], the total loss $\mathcal{L}_{\text{render}}$ for fully end-to-end training is

$$\mathcal{L}_{\text{render}} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{id}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{2d}}\mathcal{L}_{\text{2d}} + \lambda_{\text{3d}}\mathcal{L}_{\text{3d}} \tag{4}$$

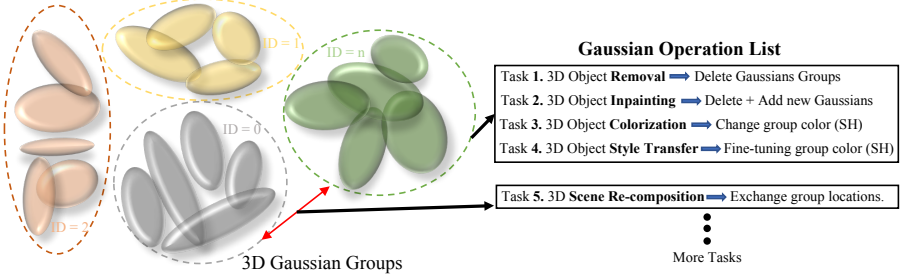


Fig. 3: The grouped 3D Gaussians after training, where each group represents a specific instance / stuff of the 3D scene and can be fully decoupled. Our representation is efficient to support versatile downstream scene editing applications, where we design a Gaussian Operation List consisting of simple operations like group deletion, group addition, finetuning Spherical Harmonic (SH) and exchanging 3D center locations.

3.3 Gaussian Grouping for Scene Editing

After the 3D Gaussian field training and grouping (Sec 3.2), as in Figure 3, we represent the whole 3D scene with a set of grouped 3D Gaussians. To perform various downstream local scene editing tasks, we propose efficient Local Gaussian Editing. Thanks to the decoupled scene representation, instead of fine-tuning all 3D Gaussians, we freeze the properties for most of the well-trained Gaussians and only adjust a small part of existing or newly added 3D Gaussians relevant to the editing targets. For 3D object removal, we simply delete the 3D Gaussians of the editing target. For 3D scene re-composition, we exchange the 3D location between two Gaussian groups. These two editing applications are direct with no parameter tuning. For 3D object inpainting, we first delete the relevant 3D Gaussians and then add a small number of new Gaussians to be supervised by the 2D inpainting results by LaMa [46] during rendering. For 3D object colorization, we only tune the color (SH) parameters of the corresponding Gaussian group to preserve the learned 3D scene geometry. For 3D object style transfer, we further unfreeze the 3D positions and sizes to achieve more realistic results.

Our local Gaussian editing scheme is time-efficient, as shown in our experiment. Because of our fine-grained mask modeling, it also supports multiple concurrent local editings without interfering with each other or re-training the whole global 3D scene representation with new editing operations. Compared to NeRF-based approaches [12,19,34], we also provide extensive visual comparisons for scene editing cases and discussion on mask granularity in the Supp. file.

4 Experiments

4.1 Dataset and Experiment Setup

Datasets To measure segmentation or fine-grained localization accuracy in open-world scene, we evolve the existing LERF-Localization [16] evaluation dataset and propose the *LERF-Mask* dataset, where we manually annotate three scenes from LERF-Localization with accurate masks instead of using coarse bounding boxes. For each 3D scene, we provide 7.7 text queries with corresponding GT

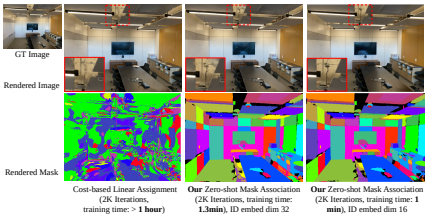


Fig. 4: Ablation on the Identity Consistency across views, where we treat multi-view images as a video and associate the mask labels to generate coherent segmentation labels [7] for training. We founding using cost-based linear assignment [44] leads to slower training and inferior testing performance in both reconstruction and segmentation.

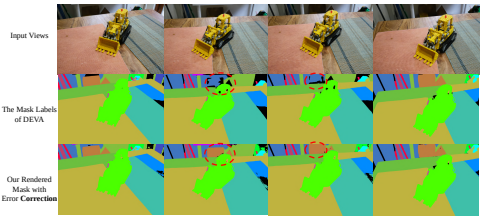


Fig. 5: Robustness to input masks errors on Mip-NeRF 360 [1]. In the 2nd and 3rd columns (middle two views), SAM + DEVA fails to segment and associate the chair across frames. However, owing to the shared 3D Gaussian representation during reconstruction, Gaussian Grouping successfully corrects the error in mask labels and segments the black chair during the multi-view rendering.

mask labels on average. We also provide 3D panoptic segmentation results on Replica [45] and ScanNet [8] dataset. More details are in the Supp. file

To evaluate the reconstruction quality, we tested our Gaussian Grouping on 7 of full 9 sets of scenes presented in Mip-NeRF 360 [1], where the flowers and treehill are skipped due to the non-public access right. We also take diverse 3D scene cases from LLFF [32], Tanks & Temples [18] and Instruct-NeRF2NeRF [12] for visual comparison on scene editing.

Implementation Details We implement Gaussian Grouping based on Gaussian Splatting [15]. We add a 16-dimension identity encoding to each Gaussian and implement forward and backward cuda rasterization similar to the RGB feature. The output classification linear layer has 16 input and 256 output channels. In training, $\lambda_{2d} = 1.0$ and $\lambda_{2d} = 2.0$. We use the Adam optimizer for both Gaussians and the linear layer, with a learning rate of 2.5e-3 for identity encoding and 5e-4 for the linear layer. For 3D regularization loss, we choose $k = 5$ and $m = 1000$. All datasets are trained for 30K iterations on one A100 GPU. Refer to our Supp. file for more details on scene editing and SAM's mask granularity.

4.2 Ablation Experiments

Ablation on Mask Cross-view Association To study the effect of cross-view masks association [7] for input preparation, we replace the associated masks input to the individual masks predicted by SAM per image in Gaussian Grouping. We take the cost-based linear assignment strategy proposed in [44], and perform the visual comparison on rendering results in Figure 4. The linear assignment not only heavily slows down the whole training due to the assignment computation in each iteration, but also produces noisy mask predictions. This is owing to the large gradients brought by the unstable cost-based liner assignment, especially at the initial training stage of the network, where masks prediction is nearly random. For 2K training iteration, linear assignment requires 1 hour but

Table 1: Influence of Identity Encoding on Mip-NeRF 360 [1] dataset with its 7 public scenes. The joint training of the introduced Identity Encodings does not hurt the original Gaussian reconstruction quality.

Model	Scene Seg	Scene Edit	PSNR↑	SSIM↑	LPIPS↓	FPS
Baseline: Gaussian Splatting [15]	-	-	28.69	0.870	0.182	∼200
Gaussian Grouping	✓	✓	28.43	0.863	0.189	∼170

Table 2: Ablation of K of 3D Regularization Loss on the 3D object removal. RAcc: Object Removal Accuracy.

Model	Gaussian Splatting	Gaussian Grouping				
		K=0	K=1	k=2	K=5	K=10
PSNR	30.32	30.51	30.62	30.61	**30.72**	30.62
RAcc	N/A	41.2%	40.5%	67.5%	76.6%	**77.8%**

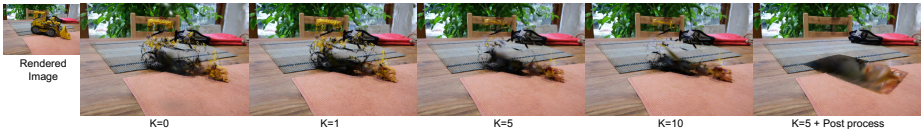


Fig. 6: Visual ablation of K in the 3D Regularization Loss on object removal editing of MipNeRF360. We remove Gaussians classified as lego with various K. We compute the convex hull of the removed 3D Gaussian points as the post process.

our associated mask input only requires 1 minute. Also, we compare the scene render reconstruction quality in Figure 4, where the appearance details of the rope attached to the ceiling are much better preserved by our method.

Masks Association Errors Correction and Robustness Gaussian Grouping can also correct the 2D segmentation errors produced by DEVA, as shown in Figure 5. Even taking the training input with partial 2D mask labels lost by tracking, Gaussian Grouping is robust in 3D association and can retrieve them based on the shared 3D Gaussian representation across different views.

Influence of the Identity Encoding In Table 1, we study the influence of our introduced Identity Encoding on original Gaussian Splatting's 3D reconstruction performance and speed. The performance of Gaussian Grouping is on par with the original Gaussian Splatting method with negligible decrease, but it allows anything in the whole 3D scene to be segmented. By Identity Encoding, we lift the 2D zero-shot segmentation of SAM to the 3D open world. Further, the grouped 3D Gaussians support a wide range of downstream editing tasks. We simply train all model components in an end-to-end manner jointly. This is different from Panoptic Lifting [22] which requires to block gradients from the segmentation branch back to the reconstruction branch.

Ablation on Identity Encoding Dimension We study the impact of dimension for our Identity Encoding in Figure 4. To keep the training efficiency, we set the dimension of Identity Encoding to only 16 because it not only shows a good segmentation separation between objects but also keeps the training efficiency. Doubling the dimension to 32 does not bring a better reconstruction quality compared to 16 but make training 1.3 times slower.

Ablation of 3D Regularization Loss We perform ablation of K in our 3D Regularization Loss on the Kitchen dataset of Mip-NeRF 360 [1] to select K. As in Fig. 6 and Table 2, $K = 5$ achieves both the best balance between the scene reconstruction and 3D object removal accuracy. Figure 6 further visualizes the influence of K during the 3D object removal process.

Fig. 7: Ablation on the Grouping Loss on the Bear inpainting case. The joint supervision of 2D and 3D losses addresses the "transparent bear issue", which shows better Gaussian Grouping accuracy. After deleting the Gaussians belonging to the bear, we detect the image hole region with no Gaussians projection covering and then use LaMa [46] to produce a single view inpainting result to guide the learning for newly added inpainting Gaussians.

Visual Ablation on the Grouping Losses In Figure 7, we study the effect of our grouping loss components, where solely using 2D Identity Loss will result in the 'transparent bear issue'. This is due to Gaussians inside the bear being occluded during training and cannot be supervised sufficiently. We address it by proposing 3D Identity/Regularization Loss for joint training with the 2D loss.

4.3 3D Multi-view Segmentation

Open-vocabulary Segmentation Comparison We compare the segmentation quality of Gaussian Grouping in 3D scenes with the state-of-the-art open-vocabulary 3D segmentation methods, such as LERF [16] and SA3D [4]. To compare fine-grained mask localization quality, we annotate the test views of three 3D scenes from the LERF-Localization [16] dataset with accurate masks to replace its original coarse bounding boxes. In Table 3, the advantage of our Gaussian Grouping is significant, doubling the performance of LERF and SA3D on both the "figurines" and "ramen" scenes. We also show the visual segmentation comparison in Figure 8, where the segmentation prediction by our method is much more accurate with a clear boundary, while LERF-based similarity computing only provides a rough localization area. Since SAM does not support language prompts, both SA3D and our method adopt Grounding DINO [26] to identify the mask ID in a 2D image, and then pick the corresponding mask from our anything masks prediction.

3D Panoptic Segmentation Comparison We compare the panoptic segmentation quality with Panoptic Lifting [44] in Table 4. We use the same semantic mask labels from Mask-DINO [23] with semantic information for both Panoptic Lifting [44] and our method. Gaussian Grouping outperforms Panoptic Lifting in both performance and speed.

4.4 3D Scene Editing

3D Object Removal 3D object removal is to completely delete an object from the 3D scene, where the background behind the removed instance / stuff can be noisy or have a hole because of no observation. In Figure 9, we compare the removal effect of our Gaussian Grouping with the Distilled Feature Fields

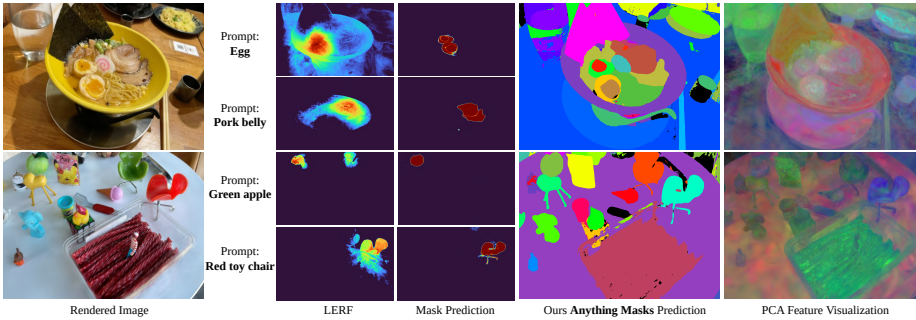


Fig. 8: Segmentation comparison between LERF [16] and our Gaussian Grouping on the rendering view. The masks predicted by Gaussian Grouping contain much sharper and more accurate boundaries than LERF. Also, our approach is better at distinguishing objects with similar colors, such as the "Green apple" prompt case.

Table 3: Comparison of Open Vocabulary Segmentation on LERF-Mask dataset. We adopt the detections from Grounding DINO [26] to select mask IDs in a 3D scene.

Model	figurines		ramen		teatime	
	mIoU	mBIoU	mIoU	mBIoU	mIoU	mBIoU
DEVA [7]	46.2	45.1	56.8	51.1	54.3	52.2
LERF [16]	33.5	30.6	28.3	14.7	49.7	42.6
SA3D [4]	24.9	23.8	7.4	7.0	42.5	39.2
LangSplat [38]	52.8	50.5	50.4	44.7	69.5	65.6
Gaussian Grouping	**69.7**	**67.9**	**77.0**	**68.7**	**71.7**	**66.1**

Table 4: Panoptic segmentation comparison on novel views. For semantic information of the prediction, we adopt Mask-DINO [23] as the semantic mask label generator for each view. We refer to the reported setting from [9].

Model	Replica			ScanNet		
	mIoU (%)	PQ^{scene}(%)	FPS	mIoU(%)	PQ^{scene}(%)	FPS
Panoptic Lifting [44]	66.22	64.34	∼10	67.01	60.74	∼10
Gaussian Grouping	**71.15**	**66.52**	**∼140**	**68.70**	**61.83**	**∼150**

Fig. 9: 3D Object removal on the Tanks & Temples dataset [18]. Compared to DFFs [19], our Gaussian Grouping can remove the large-scale objects, such as truck, from the 3D scene with greatly reduced artifacts **w/o** leaving a blurry background.

(DFFs) [19]. For challenging scene cases with large objects, our method can clearly separate the 3D object from the background. While the performance of DFFs is limited by the quality of its CLIP-distilled features, which results in the complete foreground removal (Train case) or inaccurate region removal with obvious artifacts (Truck case).

3D Object Inpainting Based on the 3D object removal, in the 3D object inpainting task, we aim to further fill the "hole regions" due to missing observation across all views and make it a photo-realistic and view-consistent natural 3D scene. We first detect the regions which are invisible in all views after deletion, and then inpaint these "invisible regions" instead of the whole "2D object

Fig. 10: Comparison on 3D object inpainting cases, where SPIn-NeRF [34] requires 5h training while our method with better inpainting quality only needs 1 hour training and 20 minutes finetuning.

Fig. 11: Multi-object editing within the same 3D scene, where we concurrently perform 3D object removal for objects in the red and green circles, and colorization for the glove in the purple circle.

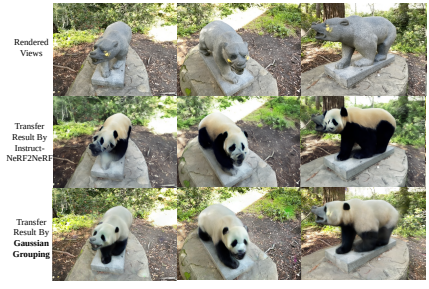


Fig. 12: Visual comparison of 3D object style transfer between Instruct-NeRF2NeRF [12] and Gaussian Grouping under the same instruction prompt "Turn the bear into a panda".

regions". In each rendering view, we adopt the 2D inpainted image to guide the learning of the newly introduced 3D Gaussian. In Figure 10, compared to SPIn-NeRF [34], the inpainting result of our Gaussian Grouping better preserves spatial detail and multi-view coherence.

3D Object Style Transfer Gaussian Grouping supports 3D object style transfer efficiently. We compare with the recent Instruct-NeRF2NeRF [12] on 3D object style transfer in Figure 12, using the same instruction prompt "turn the bear into a panda" and the same image guidance by InstructPix2Pix [2]. Gaussian Grouping produces more coherent and natural transferred "bear" across views. Since our Gaussian Grouping models "anything masks" of the open-world 3D scene, including the bear, the spatial details of background regions (outside the bear) are also faithfully preserved by our method. While for Instruct-NeRF2NeRF, a large portion of background regions are unnecessarily getting blurry with degraded quality. We describe the detailed pipeline of 3D object style transfer in the Supp. file.

3D Multi-Object Editing In Figure 11, we demonstrate multiple editing actions (like removing objects and colorization) on different Gaussian groups of the 3D scene. This grouped 3D Gaussian representation enables concurrent editing of several objects while maintaining non-interference among them.

Limitation Due to the lack of dynamic modeling and time-dependent updating, the proposed 3D Gaussian Grouping method is currently limited to the static 3D scene. Also, it would also be interesting to further explore fully unsupervised 3D Gaussian grouping in the future.

5 Conclusion

We propose Gaussian Grouping, the first 3D Gaussian-based approach to jointly reconstruct and segment anything in the open-world 3D scene. We introduce an Identity Encoding for 3D Gaussians that is supervised both by 2D mask predictions from SAM and 3D spatial consistency. Based on this grouped and discrete 3D scene representation, we further show it can support versatile scene editing applications, such as 3D object removal, 3D object inpainting, 3D object style transfer and scene recomposition, with both high-quality visual effects and good time efficiency.

6 Appendix

In this supplementary material, we first conduct additional experiment analysis of our Gaussian Grouping in Section 6.1, including multi-granularity, segmentation efficiency, quantitative editing evaluation and robustness. Then, in Section 6.2, we describe the detailed process of annotating our proposed LERF-Mask datasets with the visualization of annotation examples. We further provide a more detailed 3D object inpainting and style transfer pipeline description in Section 6.3. Finally, we illustrate the algorithm pseudocode of our Gaussian Grouping and more implementation details in Section 6.4, including the method limitation analysis. Please refer to our project page for extensive 3D results comparison.

6.1 Supplementary Experiments

Segmentation Efficiency In Fig. 13, we compare the segmentation results with SA3D [4] on the proposed LERF-Mask dataset for open-world 3D segmentation. Our Gaussian Grouping shows great advantages in segmentation efficiency. Using the same running environment, *our Gaussian Grouping jointly segments **all objects** of the 3D scene in 9 minutes, while SA3D requires 35 minutes for **each object*** due to its inverse rendering design in 3D voxel grids. To segment each object of the 3D scene, SA3D repeatedly needs separate new training, which makes SA3D time-consuming and not user-friendly in multi-object segmentation or editing scenarios.

Multi-Granularity of Masks As in Fig. 14, our method can process SAM's anything masks at different granularity levels in the multiple views as follows:

*Step **1**).* Firstly, SAM predicts dense anything mask proposals (including both large-granularity and small-granularity ones) for each frame.

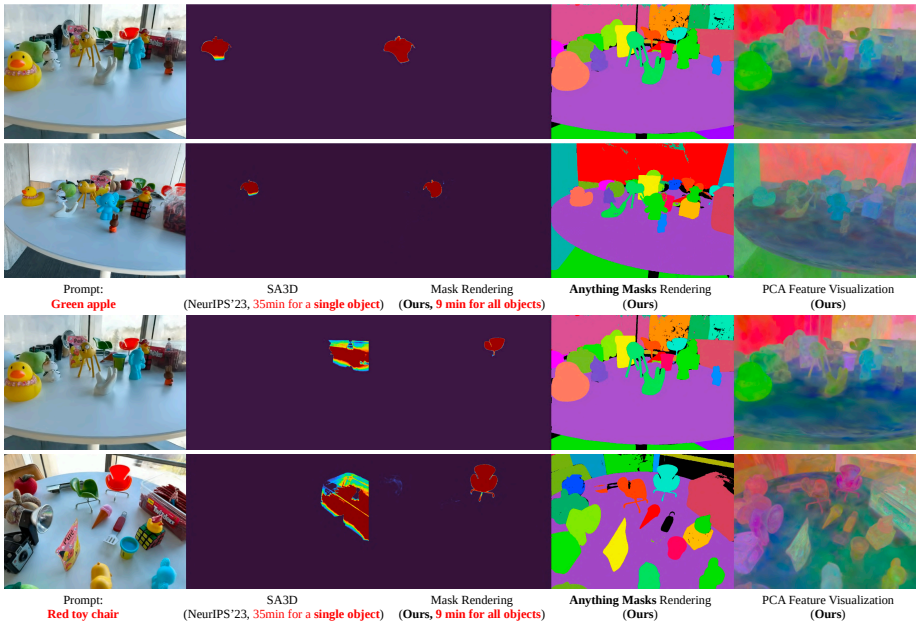


Fig. 13: Segmentation comparison between SA3D [4] and our Gaussian Grouping on the rendering view. We adopt PCA to visualize the rendered Identity Encoding features in the rightmost column. *Note that SA3D does not support concurrent multi-object segmentation* due to its design limitation in the inverse rendering, which requires training and rendering for each segmentation target for around 35 minutes (∼20min for training and ∼15min for rendering). In contrast, our Gaussian Grouping shows great efficiency by segmenting all objects in the scene only in 9 minutes.

Step 2). Then, these masks are scored and sorted based on their mask areas. Masks are ranked in descending (from large to small) or ascending (from small to large) order to pick varying levels of granularity for the label of each pixel.

Step 3). Furthermore, we filter large-area or small-area masks through their overlapping IoU with a threshold.

Step 4). For mask association, masks are temporally propagated in a bi-direction to obtain in-clip consensus. Refer to Sec 3.2 of Deva [7] for details.

Quantitative Evaluation of Editing Following Instruct-NeRF2NeRF [12], we provide the CLIP Text-Image Direction Similarity evaluation for three of our editing tasks in Table 5. Gaussian grouping supports versatile scene editing tasks, including 3D Object Inpainting, 3D Object Style Transfer and 3D Object Removal. For each of these editing tasks, Gaussian Grouping outperforms the corresponding SOTA method specifically designed for it. For inpainting, the descriptions of original and edited scenes are "Lego toy on the table" and "A flat table". For style transfer, the descriptions are "A bear statue" and "Turn the bear into a panda". For object removal, the descriptions are "A truck on urban street" and "Urban street".

Fig. 14: Consistent tracking results at multi-granularity masks. The 1st row is in coarse granularity, and the 2nd row is fine-grained.

Table 5: Quantitative comparison on three Scene Editing Tasks using the CLIP Text-Image Direction Similarity [12].

Task	Dataset	Model	CLIP Text-Image Direction Similarity
3D Object Inpainting	MipNeRF360/Kitchen	SPIN-NeRF [34]	0.126
		Ours	**0.153**
3D Object Style Transfer	Instruct-NeRF2NeRF/Bear	Instruct-NeRF2NeRF [12]	0.171
		Ours	**0.178**
3D Object Removal	Tandt/Truck	DFFs [19]	0.166
		Ours	**0.183**

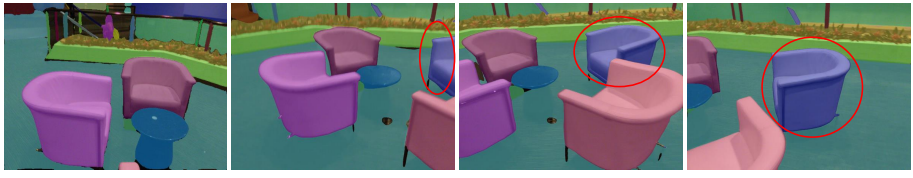


Fig. 15: Tracking results for a new object in subsequent frames. The red circle highlights the new chair appearing in the 2nd frame.

Robustness of New Objects SAM+Tracking can process new objects on subsequent frames. When new high-confident segmentation doesn't match the previous objects and has high confidence, a new instance ID is assigned to it (see the new chair in the red circle in Fig. 15).

Sparse View Input For few-shot or sparse-view input, video tracking still obtains a good result for pre-processing, since it is visual feature-based and camera-motion robust. In Figure 16, we use a 3-view input for DEVA pre-processing and still get a decent tracking result. 3D reconstruction of the original Gaussian Splatting is not good with sparse-view input, but it is beyond the scope of this paper. It is retained as a component for sparse-view Gaussian Splatting reconstruction, intended for further research in the future.

6.2 Details on the LERF-Mask Annotation

Annotation Pipeline To measure the segmentation or fine-grained localization accuracy in the open-wold 3D scene, we construct the LERF-Mask dataset based on the existing LERF-Localization [16] evaluation dataset, where we manually

Fig. 16: Sparse 3-view tracking and reconstruction comparison.

annotate three scenes from LERF-Localization with accurate masks instead of using coarse bounding boxes. For each 3D scene, we provide 7.7 text queries with corresponding GT mask labels on average. We use Roboflow [10] platform for label annotation, and it uses SAM [17] as an auxiliary tool for interactive segmentation. Similar to the annotation used in LERF, for each of the 3 scenes, we choose 2-4 novel views for testing and annotating the rendering of novel views.

Annotation Examples All language prompts used for our LERF-Mask dataset evaluation are listed in Table 6, which contains 23 prompts in total. Also, we provide visualization on the mask annotations in Figure 17.

Scene	Text queries		
Figurines	green apple porcelain hand rubber duck with red hat	green toy chair red apple	old camera red toy chair
Ramen	chopsticks pork belly	egg wavy noodles in bowl	glass of water yellow bowl
Teatime	apple cookies on a plate sheep tea in a glass	bag of cookies paper napkin spoon handle	coffee mug plate stuffed bear

Table 6: Prompt labels used during segmentation experiments in our proposed LERF-Mask dataset (23 total).

6.3 Local Gaussian Editing: Steps of Object Inpainting & Style Transfer

3D Object Inpainting Pipeline For inpainting, we remove the 3D Gaussians of the selected target by using a Gaussian Grouping model well-trained for 3D reconstruction and segmentation and add new Gaussians for finetuning. The steps are as follows:

Step 1). Train the Gaussian Grouping model with our proposed 2D and 3D Identity Grouping loss.

Step 2). Select the target object for inpainting. For each Identity Encoding associated with a 3D Gaussian, we acquire its linear layer classification result. Subsequently, we remove those 3D Gaussians that are classified as the label of

Fig. 17: Annotation visualization of our proposed LERF-Mask dataset. We manually annotate three scenes from LERF-Localization [16] with accurate masks instead to replace the coarse bounding boxes in [12]. The text queries are detailed in Table 6.

the selected object. Also, we remove the 3D Gaussians with position inside the convex hull of the object Gaussians.

Step 3). On the rendering views after the deletion of the object, we detect the "blurry hole" with Grounding-DINO [26] as the mask for 2D inpainting and use DEVA [7] for association. We use LAMA [46] inpainting on each view as the target for finetuning.

Step 4). After the 3D Gaussians of the target object are deleted, we clone 200K new Gaussians near the deletion region. We freeze the other Gaussians, and only finetune the newly introduced 3D Gaussians.

Step 5). During the finetuning, we employ L1 loss only in the outside regions of the object mask, and adopt LPIPS loss inside the bounding box of the object

mask.

3D Object Style Transfer Pipeline For style transfer, we finetune the 3D Gaussians belonging to the corresponding target by using a Gaussian Grouping model well-trained for 3D reconstruction and segmentation. The steps are as follows:

Step 1). Train our Gaussian Grouping model with our proposed 2D and 3D Identity Grouping loss.

Step 2). Select the target object for style transfer. For each Identity Encoding associated with a 3D Gaussian, we acquire its linear layer classification result. Subsequently, we only finetune those 3D Gaussians that are classified as the label of the selected object. Also, we finetune the 3D Gaussians with position inside the convex hull of the selected object. The Gaussians irrelevant to the editing target are frozen. During finetuning, we freeze the 3D position of Gaussains and make other Gaussian parameters (color, variance, opacity, etc.) trainable.

Step 3). During the finetuning process, we dynamically update the target images using an image-level style transfer model that has been pre-trained. Specifically, we employ InstructPix2Pix [2], introducing a noise input composed of the rendered view combined with random noise. This approach involves conditioning the diffusion model on a ground truth image to enhance accuracy and consistency.

Step 4). To preserve the spatial details of the background regions, rendering losses are exclusively performed within the mask of the style transfer target. On the 2D rendered view, we employ L1 loss inside the object mask and LPIPS loss within the bounding box that encloses the object mask.

6.4 More Implementation Details

More implementation details We implement Gaussian Grouping based on Gaussian Splatting [15]. We add a 16-dimension identity encoding as a feature of each Gaussian, and implement forward and backward cuda rasterization similar to the direct current of Spherical Harmonics. The 3D Identity Encoding has a shape of $N * 1 * 16$, where N is the number of Gaussians. We set the degree of Spherical Harmonics to zero since instance identity does not change across views. The rendered 2D Identity Encoding has a shape of $16 * H * W$. 3D Identity Encoding and 2D Identity Encoding share the same identity classification linear layer with $(16, 256)$ input and output channels.

During training, we set $\lambda_{\text{2d}} = 1.0$ and $\lambda_{\text{3d}} = 2.0$. We use the Adam optimizer for both Gaussians and the linear layer, with a learning rate of 0.0025 for identity encoding and 0.0005 for the linear layer. For 3D regularization loss, we set the nearest neighboring number to $k = 5$, and the sampling points number to $m = 1000$. To improve efficiency and avoid calculating loss at boundary points, we downsample the point cloud to 300K to calculate the loss. 3D regularization loss only affects the segmentation of identity encoding and does not affect the

density of the Gaussians. We use the same adaptive density control as Gaussian Splatting. All datasets are trained for 30K iterations on one A100 GPU.

6.5 Limitation

Our Gaussian Grouping segments "anything masks" with the assistance of SAM. But the "anything mask labels" by original SAM [17] have no direct semantic language information. We adopt the Grounding-DINO [26] for open vocabulary segmentation to pick the 2D object, and match our anything masks rendering. When some language prompts are very complicated, the Grounding-DINO can not acquire the correct mask from the input text prompt and will give a wrong mask prediction. In this case, even if we provide the correct mask in anything mask rendering, we do not obtain explicit category information. Also, the zero-shot 2D association accuracy of DEVA [7] will also limit the open-world 3D segmentation performance of Gaussian Grouping. This can be solved by further improvement of the vision language detection model and better association schemes in the future.

References

1. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: CVPR (2022)
2. Brooks, T., Holynski, A., Efros, A.A.: Instructpix2pix: Learning to follow image editing instructions. In: CVPR (2023)
3. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: ICCV (2021)
4. Cen, J., Zhou, Z., Fang, J., Yang, C., Shen, W., Xie, L., Zhang, X., Tian, Q.: Segment anything in 3d with nerfs. In: NeurIPS (2023)
5. Chen, X., Tang, J., Wan, D., Wang, J., Zeng, G.: Interactive segment anything nerf with feature imitation. arXiv preprint arXiv:2305.16233 (2023)
6. Chen, Z., Wang, F., Liu, H.: Text-to-3d using gaussian splatting. arXiv preprint arXiv:2309.16585 (2023)
7. Cheng, H.K., Oh, S.W., Price, B., Schwing, A., Lee, J.Y.: Tracking anything with decoupled video segmentation. In: ICCV (2023)
8. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: CVPR (2017)
9. Dou, B., Zhang, T., Ma, Y., Wang, Z., Yuan, Z.: Cosseggaussians: Compact and swift scene segmenting 3d gaussians with dual feature fusion. arXiv preprint arXiv:2401.05925 (2024)
10. Dwyer, B., Nelson, J., Solawetz, J.: Roboflow (version 1.0) [software]. https://roboflow.com. (2022)
11. Fu, X., Zhang, S., Chen, T., Lu, Y., Zhu, L., Zhou, X., Geiger, A., Liao, Y.: Panoptic nerf: 3d-to-2d label transfer for panoptic urban scene segmentation. In: International Conference on 3D Vision (3DV) (2022)
12. Haque, A., Tancik, M., Efros, A., Holynski, A., Kanazawa, A.: Instruct-nerf2nerf: Editing 3d scenes with instructions. In: ICCV (2023)
13. Kania, K., Yi, K.M., Kowalski, M., Trzciński, T., Tagliasacchi, A.: CoNeRF: Controllable Neural Radiance Fields. In: CVPR (2022)
14. Ke, L., Ye, M., Danelljan, M., Liu, Y., Tai, Y.W., Tang, C.K., Yu, F.: Segment anything in high quality. In: NeurIPS (2023)
15. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM TOG **42**(4), 1–14 (2023)
16. Kerr, J., Kim, C.M., Goldberg, K., Kanazawa, A., Tancik, M.: Lerf: Language embedded radiance fields. In: ICCV (2023)
17. 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: ICCV (2023)
18. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics (ToG) **36**(4), 1–13 (2017)
19. Kobayashi, S., Matsumoto, E., Sitzmann, V.: Decomposing nerf for editing via feature field distillation. In: NeurIPS (2022)
20. Kopanas, G., Leimkühler, T., Rainer, G., Jambon, C., Drettakis, G.: Neural point catacaustics for novel-view synthesis of reflections. ACM TOG **41**(6), 1–15 (2022)
21. Kopanas, G., Philip, J., Leimkühler, T., Drettakis, G.: Point-based neural rendering with per-view optimization. In: Computer Graphics Forum. vol. 40, pp. 29–43 (2021)

22. 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: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12871–12881 (2022)
23. Li, F., Zhang, H., xu, H., Liu, S., Zhang, L., Ni, L.M., Shum, H.Y.: Mask dino: Towards a unified transformer-based framework for object detection and segmentation. In: CVPR (2023)
24. Li, Y., Lin, Z.H., Forsyth, D., Huang, J.B., Wang, S.: Climatenerf: Extreme weather synthesis in neural radiance field. In: ICCV (2023)
25. Liu, H.K., Shen, I., Chen, B.Y., et al.: Nerf-in: Free-form nerf inpainting with rgb-d priors. arXiv preprint arXiv:2206.04901 (2022)
26. Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Li, C., Yang, J., Su, H., Zhu, J., et al.: Grounding dino: Marrying dino with grounded pre-training for open-set object detection. arXiv preprint arXiv:2303.05499 (2023)
27. Liu, S., Zhang, X., Zhang, Z., Zhang, R., Zhu, J.Y., Russell, B.: Editing conditional radiance fields. In: ICCV (2021)
28. Liu, Y., Hu, B., Huang, J., Tai, Y.W., Tang, C.K.: Instance neural radiance field. In: ICCV (2023)
29. Luiten, J., Kopanas, G., Leibe, B., Ramanan, D.: Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. arXiv preprint arXiv:2308.09713 (2023)
30. Max, N.: Optical models for direct volume rendering. IEEE TVCG **1**(2), 99–108 (1995)
31. Mazur, K., Sucar, E., Davison, A.J.: Feature-realistic neural fusion for real-time, open set scene understanding. In: ICRA (2023)
32. Mildenhall, B., Srinivasan, P.P., Ortiz-Cayon, R., Kalantari, N.K., Ramamoorthi, R., Ng, R., Kar, A.: Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. ACM Transactions on Graphics (TOG) (2019)
33. 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: ECCV (2020)
34. Mirzaei, A., Aumentado-Armstrong, T., Derpanis, K.G., Kelly, J., Brubaker, M.A., Gilitschenski, I., Levinshtein, A.: Spin-nerf: Multiview segmentation and perceptual inpainting with neural radiance fields. In: CVPR (2023)
35. Mirzaei, A., Kant, Y., Kelly, J., Gilitschenski, I.: Laterf: Label and text driven object radiance fields. In: ECCV (2022)
36. Ost, J., Mannan, F., Thuerey, N., Knodt, J., Heide, F.: Neural scene graphs for dynamic scenes. In: CVPR (2021)
37. Peng, S., Genova, K., Jiang, C., Tagliasacchi, A., Pollefeys, M., Funkhouser, T., et al.: Openscene: 3d scene understanding with open vocabularies. In: CVPR (2023)
38. Qin, M., Li, W., Zhou, J., Wang, H., Pfister, H.: Langsplat: 3d language gaussian splatting. arXiv preprint arXiv:2312.16084 (2023)
39. 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: ICML (2021)
40. Rebain, D., Jiang, W., Yazdani, S., Li, K., Yi, K.M., Tagliasacchi, A.: Derf: Decomposed radiance fields. In: CVPR (2021)
41. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR (2022)
42. Schult, J., Engelmann, F., Hermans, A., Litany, O., Tang, S., Leibe, B.: Mask3D: Mask Transformer for 3D Semantic Instance Segmentation. In: ICRA (2023)

43. Shen, Q., Yang, X., Wang, X.: Anything-3d: Towards single-view anything reconstruction in the wild. arXiv preprint arXiv:2304.10261 (2023)
44. Siddiqui, Y., Porzi, L., Bulò, S.R., Müller, N., Nießner, M., Dai, A., Kontschieder, P.: Panoptic lifting for 3d scene understanding with neural fields. In: CVPR (2023)
45. Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J.J., Mur-Artal, R., Ren, C., Verma, S., Clarkson, A., Yan, M., Budge, B., Yan, Y., Pan, X., Yon, J., Zou, Y., Leon, K., Carter, N., Briales, J., Gillingham, T., Mueggler, E., Pesqueira, L., Savva, M., Batra, D., Strasdat, H.M., Nardi, R.D., Goesele, M., Lovegrove, S., Newcombe, R.: The Replica dataset: A digital replica of indoor spaces. arXiv preprint arXiv:1906.05797 (2019)
46. Suvorov, R., Logacheva, E., Mashikhin, A., Remizova, A., Ashukha, A., Silvestrov, A., Kong, N., Goka, H., Park, K., Lempitsky, V.: Resolution-robust large mask inpainting with fourier convolutions. In: WACV (2022)
47. Takmaz, A., Fedele, E., Sumner, R.W., Pollefeys, M., Tombari, F., Engelmann, F.: OpenMask3D: Open-Vocabulary 3D Instance Segmentation. In: NeurIPS (2023)
48. Tang, J., Ren, J., Zhou, H., Liu, Z., Zeng, G.: Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. arXiv preprint arXiv:2309.16653 (2023)
49. Tschernezki, V., Laina, I., Larlus, D., Vedaldi, A.: Neural feature fusion fields: 3d distillation of self-supervised 2d image representations. arXiv preprint arXiv:2209.03494 (2022)
50. Tschernezki, V., Laina, I., Larlus, D., Vedaldi, A.: Neural feature fusion fields: 3D distillation of self-supervised 2D image representations. In: International Conference on 3D Vision (3DV) (2022)
51. Vora, S., Radwan, N., Greff, K., Meyer, H., Genova, K., Sajjadi, M.S.M., Pot, E., Tagliasacchi, A., Duckworth, D.: Nesf: Neural semantic fields for generalizable semantic segmentation of 3d scenes (2021)
52. Wang, B., Chen, L., Yang, B.: Dm-nerf: 3d scene geometry decomposition and manipulation from 2d images. arXiv preprint arXiv:2208.07227 (2022)
53. Wang, C., Chai, M., He, M., Chen, D., Liao, J.: Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In: CVPR (2022)
54. Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., Wang, X.: 4d gaussian splatting for real-time dynamic scene rendering. arXiv preprint arXiv:2310.08528 (2023)
55. Wu, Q., Liu, X., Chen, Y., Li, K., Zheng, C., Cai, J., Zheng, J.: Object-compositional neural implicit surfaces. In: ECCV (2022)
56. 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: ICCV (2021)
57. Yang, Y., Wu, X., He, T., Zhao, H., Liu, X.: Sam3d: Segment anything in 3d scenes. arXiv preprint arXiv:2306.03908 (2023)
58. Yang, Z., Yang, H., Pan, Z., Zhu, X., Zhang, L.: Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. arXiv preprint arXiv:2310.10642 (2023)
59. Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., Jin, X.: Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. arXiv preprint arXiv:2309.13101 (2023)
60. Yi, T., Fang, J., Wu, G., Xie, L., Zhang, X., Liu, W., Tian, Q., Wang, X.: Gaussiandreamer: Fast generation from text to 3d gaussian splatting with point cloud priors. arXiv preprint arXiv:2310.08529 (2023)

61. Yifan, W., Serena, F., Wu, S., Öztireli, C., Sorkine-Hornung, O.: Differentiable surface splatting for point-based geometry processing. ACM Transactions on Graphics (TOG) **38**(6), 1–14 (2019)
62. Yu, H.X., Guibas, L.J., Wu, J.: Unsupervised discovery of object radiance fields. arXiv preprint arXiv:2107.07905 (2021)
63. Yuan, Y.J., Sun, Y.T., Lai, Y.K., Ma, Y., Jia, R., Gao, L.: Nerf-editing: geometry editing of neural radiance fields. In: CVPR (2022)
64. Zhang, C., Han, D., Qiao, Y., Kim, J.U., Bae, S.H., Lee, S., Hong, C.S.: Faster segment anything: Towards lightweight sam for mobile applications. arXiv preprint arXiv:2306.14289 (2023)
65. Zhang, J., Liu, X., Ye, X., Zhao, F., Zhang, Y., Wu, M., Zhang, Y., Xu, L., Yu, J.: Editable free-viewpoint video using a layered neural representation. ACM Transactions on Graphics (TOG) **40**(4), 1–18 (2021)
66. Zhang, X., Kundu, A., Funkhouser, T., Guibas, L., Su, H., Genova, K.: Nerflets: Local radiance fields for efficient structure-aware 3d scene representation from 2d supervision. In: CVPR (2023)
67. Zhi, S., Laidlow, T., Leutenegger, S., Davison, A.J.: In-place scene labelling and understanding with implicit scene representation. In: ICCV (2021)
68. Zwicker, M., Pfister, H., Van Baar, J., Gross, M.: Surface splatting. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques. pp. 371–378 (2001)