

## 2D-GUIDED 3D GAUSSIAN SEGMENTATION

Kun Lan<sup>1</sup>, Haoran Li<sup>1</sup>, Haolin Shi<sup>1</sup>, Wenjun Wu<sup>1</sup>, Yong Liao<sup>1</sup>, Lin Wang<sup>2</sup>, Pengyuan Zhou<sup>\*1</sup>

<sup>1</sup>University of Science and Technology of China, <sup>2</sup>AI Thrust, HKUST(GZ)

{lankun, lhr123, mar, wu-wen-jun, yliaoy}@mail.ustc.edu.cn

linwang@ust.hk, zpymyyn@gmail.com

### ABSTRACT

Recently, 3D Gaussian, as an explicit 3D representation method, has demonstrated strong competitiveness over NeRF (Neural Radiance Fields) in terms of expressing complex scenes and training duration. These advantages signal a wide range of applications for 3D Gaussians in 3D understanding and editing. Meanwhile, the segmentation of 3D Gaussians is still in its infancy. The existing segmentation methods are not only cumbersome but also incapable of segmenting multiple objects simultaneously in a short amount of time. In response, this paper introduces a 3D Gaussian segmentation method implemented with 2D segmentation as supervision. This approach uses input 2D segmentation maps to guide the learning of the added 3D Gaussian semantic information, while nearest neighbor clustering and statistical filtering refine the segmentation results. Experiments show that our concise method can achieve comparable performances on mIOU and mAcc for multi-object segmentation as previous single-object segmentation methods.

**Index Terms**— 3D Gaussian, 3D Segmentation

### 1. INTRODUCTION

The recently emerged 3D Gaussian technique [1] marks a significant advancement over previous 3D representation methods such as point clouds [2], meshes [3], signed distance functions (SDF) [4], and neural radiance fields (NeRF) [5], especially in terms of training time and scene reconstruction quality. The mean of each 3D Gaussian represents the position of its center point, the covariance matrix indicates rotation and size, and spherical harmonics express color. Starting with point clouds obtained from SFM [6], 3D Gaussians inherently contain the scene's geometric information, thus saving time in locating areas with concentrated objects in space. Moreover, their explicit expression method further accelerates calculations of color and density for every 3D Gaussian in space, enabling real-time rendering. Additionally, adaptive density control endows them with the capability to express detailed features. These advantages make it widely applicable in 3D understanding and editing. Nonetheless, there is little research on 3D Gaussian segmentation, which is another

critical pillar of the realm.

A few Gaussian segmentation methods have been proposed recently, yet they require further improvement. For example, Gaussian Grouping [7] requires an extended training period of about 15 minutes. SAGA [8] is complex in its implementation and struggles with segmenting multiple objects simultaneously. Additionally, the explicit expression of 3D Gaussians leads to storage overhead, preventing it from directly transferring 2D semantic features into 3D, as in NeRF segmentation [9, 10]. Finally, the scarcity of datasets and the lack of annotations impede the application of supervised segmentation methods, commonly utilized in 2D and point cloud segmentation.

In light of the aforementioned challenges, we propose leveraging a pre-trained 2D segmentation model to guide 3D Gaussian segmentation. Inspired by the 2D segmentation approach, which assigns a probability distribution vector for each pixel across different categories, we first assign an object code to each 3D Gaussian to indicate the Gaussian's categorical probability distribution. Subsequently, we employ an algorithm that guides the classification of each 3D Gaussian by minimizing the error between the 2D segmentation map and the rendered segmentation map at a given pose. Finally, we employ KNN clustering to resolve semantic ambiguity in 3D Gaussians and statistical filtering to remove erroneously segmented 3D Gaussians. We validated the effectiveness of our approach through experiments in object-centric and 360° scenes. Our contributions can be summarized as follows.

- We propose an efficient 3D Gaussian segmentation method supervised by 2D segmentation, which can learn the semantic information of a 3D scene in less than two minutes and segment multiple objects in 1-2 seconds for a given viewpoint.
- Extensive experiments on LLFF, NeRF-360, and Mip-NeRF 360 have demonstrated the effectiveness of our method, obtaining an mIOU of 86%.

### 2. RELATED WORK

**3D Gaussian**, a recently proposed explicit representation method, has attained remarkable achievements in three-

dimensional scene reconstruction [1]. Its biggest advantage is the capability of real-time rendering. Utilizing a series of scene images and corresponding camera data, it employs 3D Gaussians to depict scene objects. Each 3D Gaussian is defined by parameters including mean, covariance matrix, opacity, and spherical harmonics. The mean pinpoints the Gaussian’s central position in the 3D scene. Expressed by a scaling matrix  $S$  and a rotation matrix  $R$ , the covariance matrix describes the Gaussian’s size and shape, while the spherical harmonics encode its color information. Gaussian Splatting then utilizes point-based rendering for efficient 3D to 2D projection.

Recent developments have seen numerous advancements in Gaussian Splatting. Innovations like DreamGaussian [11] and GaussianDreamer [12] merge this technique with Diffusion model [13], facilitating text-to-3D generation. 4D Gaussian Splatting [14] extends these methods to dynamic scene representation and rendering. Focusing on segmentation, Gaussian Grouping [7] and SAGA [8] have made significant strides. They both employ the Segment Anything Model (SAM) [15] to derive 2D prior segmentation data, guiding the learning of added semantic information in 3D Gaussians. In Gaussian Grouping, this information is conveyed similarly to coefficients of spherical harmonic functions, whereas SAGA uses learnable low-dimensional features. However, SAM’s reliance on geometric structures limits its semantic inclusivity in each mask. Thus, both methods propose strategies to ensure consistency of SAM’s segmentation outcomes from various perspectives. Gaussian Grouping treats images from different angles as a sequence of video frames, utilizing a pre-trained model for mask propagation and matching. In contrast, SAGA consolidates consistent, multi-granularity segmentation information across viewpoints, employing a custom-designed SAM-guidance loss.

**3D Segmentation in Radiance Fields.** Prior to the advent of 3D Gaussians, NeRF [5] stood as a prominent method in 3D characterization, sparking a plethora of derivative works [10, 16–21], including several focusing on decomposing and segmenting NeRF. A notable example is Object NeRF [17], which introduced a dual-pathway neural radiance field adept at object decomposition. Its scene branch processes spatial coordinates and viewing directions, outputting density and color details of a point from the viewer’s perspective, primarily encoding the background of the 3D scene and offering geometric context for the object branch. Uniquely, the object branch, in addition to spatial and directional inputs, integrates a learnable object activation code, enabling the independent learning of neural radiance fields for each scene object. And the 3D guard mask helps mitigate occlusion issues between objects during the learning phase. Similarly, Switch-NeRF [21] demonstrates the decomposition of large-scale neural radiance fields through a trainable gating network.

DM-NeRF [16] introduces an object field for NeRF seg-

mentation, using it to generate a one-hot vector indicating the ownership of each spatial point by an object. SPI-NeRF [20] employs a semantic radiance field, assessing the likelihood of scene locations being associated with specific objects. ISRF [10] adds semantic features to specific points and incorporates DINO [22] features of rendered images into this framework through a teacher-student model, allowing for feature interpolation at any given point. Techniques such as K-means clustering, nearest neighbor matching, and bilateral search are integrated, enabling interactive NeRF segmentation. Additionally, OR-NeRF[18] chooses to back-project 2D segmentation results into a 3D space, propagating them across different viewpoints, and then re-rendering them onto a 2D plane.

These 3D Gaussian and NeRF segmentation methods either take a long time or struggle to preserve the detailed features of the scene in the segmentation result. For this reason, we propose a method that can segment multiple objects while preserving the detailed features in a short time.

### 3. METHOD

Given a well-trained scene using 3D Gaussian representation, scene rendering images, and corresponding camera parameters, we initially employed an interactive 2D segmentation model [23] to segment the rendered images. Then, the obtained 2D segmentation maps are used as guidance to facilitate the learning of semantic information (object code) added to the 3D Gaussians. Finally, we use KNN clustering to address issues of semantic ambiguity in certain 3D Gaussians, while optional statistical filtering can help eliminate those 3D Gaussians that have been erroneously segmented. The pipeline is depicted in Fig. 1.

#### 3.1. Point-Based rendering and Semantic Information Learning

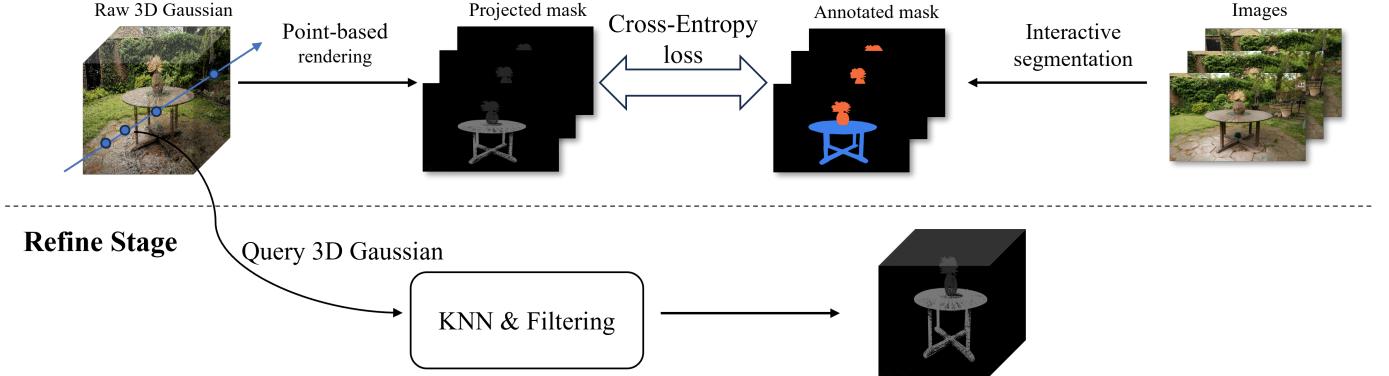
Gaussian Splatting [1] employs a point-based rendering technique ( $\alpha$ -blending) to render a 3D scene onto a plane, and the color of a pixel on the plane can be calculated as:

$$C = \sum_{i \in \mathcal{N}} c_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (1)$$

where  $\mathcal{N}$  denotes the ordered Gaussians overlapping the pixel,  $c_i$  represents the color of each 3D Gaussian projected onto the current pixel, and  $\alpha_i$  is given by evaluating a 2D Gaussian with covariance  $\Sigma$  multiplied with a learned per-Gaussian opacity. It is worth noting that  $\alpha$  expresses the opacity of any point in the projected 2D Gaussian, which decreases as its distance from the 2D Gaussian center increases.

To achieve segmentation of a 3D scene, semantic information needs to be incorporated into the representation of the scene. Inspired by 2D segmentation, we assign an object code

## Semantic Feature Learning



**Fig. 1.** The pipeline of our method. Given posed training images, we first utilize an interactive model to get the 2D prior segmentation information, then add semantic information to the 3D Gaussians, and project this information onto a 2D plane to make loss with the 2D prior knowledge, finally we use KNN and statistical filtering to refine the segmentation result.

$\mathbf{o} \in \mathcal{R}^K$  to each 3D Gaussian to represent the probability distribution of the current 3D Gaussian across various categories, where  $K$  is the number of categories. Note that we defined a background class and the first dimension of  $\mathbf{o}$  is used to represent it.

To use 2D segmentation maps as supervision for learning the added 3D semantic information, it is necessary to project the added semantic information from 3D onto a 2D plane. Inspired by  $\alpha$ -blending, we consider the pixel categories in the rendered 2D segmentation map as a weighted sum of the categories of multiple 3D Gaussians along the current ray during rendering. We assume that the first 3D Gaussian contributes the most, with each subsequent 3D Gaussian's contribution diminishing in accordance with its distance from the rendering plane, and this contribution is also proportional to the size of the 3D Gaussian itself. The category of each pixel on the rendered image can be represented by the object code of  $\mathbf{o}$  the 3D Gaussians as:

$$\hat{\mathbf{o}} = \sum_{i \in \mathcal{N}} o_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (2)$$

which simply replaces the color  $c$  of each 3D Gaussian in Eq. (1) with the object code of each 3D Gaussian.

Assuming we have  $L$  images of 2D ground truth labels  $\{I_1, \dots, I_l, \dots, I_L\}, I_l \in \mathcal{R}^{H \times W}$ ,  $L$  is number of different camera poses in the dataset,  $H$  and  $W$  are the height and width of the label respectively. Each element in the ground truth label represents the category label of the corresponding pixel. Then we generate  $L$  corresponding projected segmentation maps  $\{\bar{I}_1, \dots, \bar{I}_l, \dots, \bar{I}_L\}, \bar{I}_l \in \mathcal{R}^{K \times H \times W}$  in the same camera viewpoint as the ground truth. In these projected segmentation maps, each element represents the probability of the pixel belonging to  $i^{th}$  category,  $i = 1, 2, \dots, K$ .

Next, the original 2D segmentation maps are transformed

into one-hot vector and then reshaped to be  $M \in \mathcal{R}^{K \times N}$ , where  $N = H \times W$ . As the projected segmentation maps, we perform a similar operation and obtain  $\bar{M} \in \mathcal{R}^{K \times N}$ . Then the ground truth object mask  $M$  and corresponding projected object mask  $\bar{M}$  are used to calculate the Cross-Entropy Loss(CES):

$$L_i = -\frac{1}{N} \sum_{n=1}^N M_i^n \log \bar{M}_i^n, \quad (0 \leq i < K). \quad (3)$$

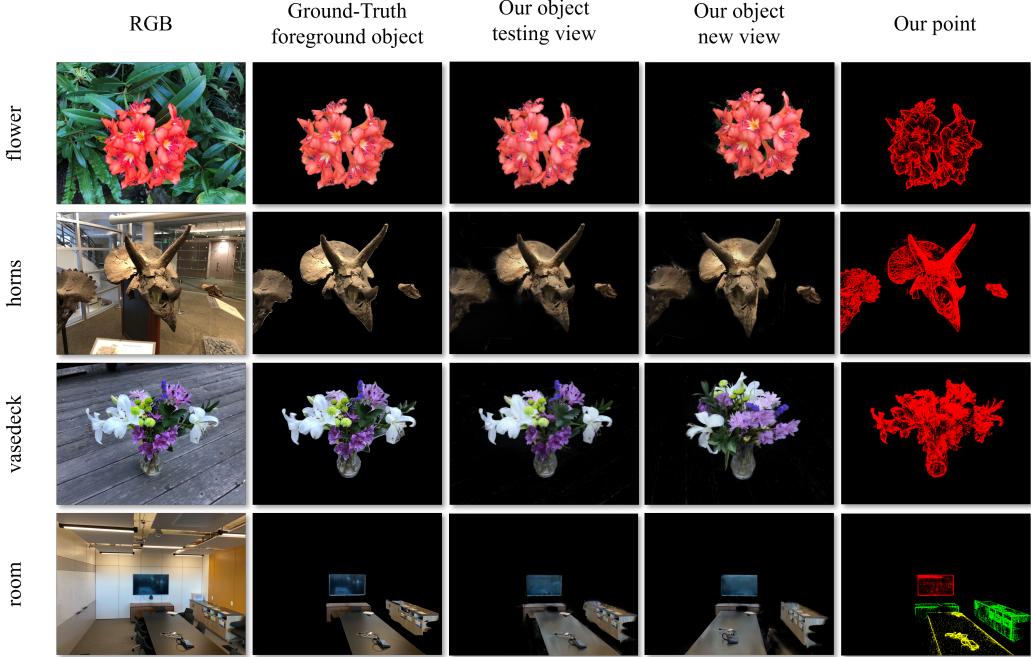
The final loss is the average of all the losses for the  $L$  pairs of images:

$$\mathcal{L} = \frac{1}{L} \sum_{l=1}^L CES_l, \text{ where } CES_l = \frac{1}{K} \sum_{i=1}^K L_i. \quad (4)$$

## 3.2. Gaussian Clustering

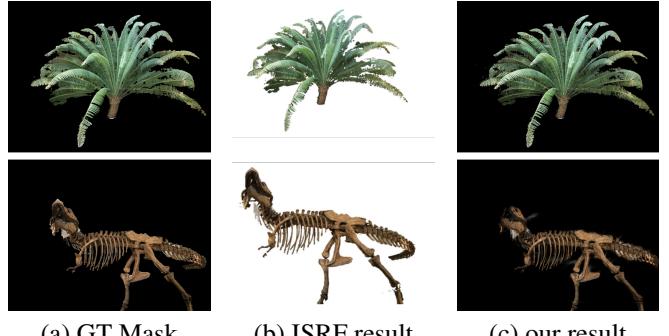
During experiments, we observed that employing 2D segmentation maps as the sole guide for learning 3D semantic information may lead to inaccuracies in the semantic information of some 3D Gaussians. These inaccuracies manifest either as 3D Gaussians approximating an initial state of uniform distribution across all categories or as exhibiting similar probabilities in a limited number of categories. To address this issue, and considering that objects are continuously distributed in space, we posit that each 3D Gaussian should typically be classified within the same category as other 3D Gaussians located within a certain proximity.

To remedy the inaccuracies in semantic information, we refer to the KNN clustering algorithm. For a 3D scene with pre-learned semantic information, we initially retrieve the object code, denoted as  $\mathbf{o}$ , of each 3D Gaussian used to represent the scene. These codes then undergo softmax process-



**Fig. 2.** The qualitative results of our method. The first and second columns are the original image and the foreground object obtained from the interactive segmentation model, respectively. The third and fourth columns are renderings of the 3D segmentation effect obtained by our method from the test viewpoint and a new viewpoint. The last column is the result of converting the different categories belonging to the 3D Gaussian into RGB values.

ing to deduce the probability distribution of each 3D Gaussian across various categories. 3D Gaussians with maximum probability values  $\max(\text{softmax}(\mathbf{o})) < \beta$  are selected. Finally, we fed the object codes of these selected 3D Gaussians along with their center coordinates into KNN for clustering. For a query 3D Gaussian, we calculate its distance from the surrounding 3D Gaussians, and the  $k$  3D Gaussians closest in distance are selected, the object code of the query Gaussian is set to the mean of these 3D Gaussians’ object code.



### 3.3. Gaussian Filtering

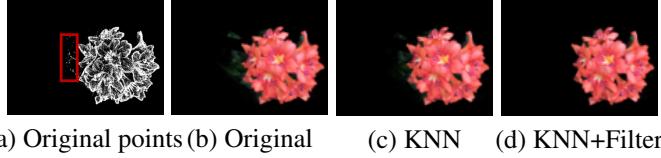
During experiments, We also found that after 3D semantic information learning and Gaussian clustering, some 3D Gaussians not belonging to the object intended for segmentation were incorrectly segmented out. We observed that these erroneously segmented 3D Gaussians are spatially distant from the rest of the segmented 3D Gaussians, as shown in Fig. 4(a). Therefore, we employ a statistical filtering algorithm similar to that used in point cloud segmentation to solve this problem. For each segmented Gaussian, we calculate its average distance  $D$  from the neighboring 3D Gaussians. Then, we compute the mean  $\mu$  and variance  $\sigma$  of these average distances. Finally, we remove those 3D Gaussians whose average distance  $D > \mu + \sigma$  from the current segmentation results.

**Fig. 3.** Comparison results. (a) is the GT Mask we used to guide the segmentation of 3D Gaussians, (b) is the result of ISRF (from the original paper), and (c) is our result.

## 4. EXPERIMENT

### 4.1. Setups

Due to the scarcity of 3D Gaussian segmentation methods and the lack of open source code for Gaussian Grouping [7] and SAGA [8], we chose to compare our method with previous NeRF segmentation methods [10]. For this purpose, we selected well-known NeRF datasets for our experiments, including LLFF [24], NeRF-360 [5], and Mip-NeRF 360 [25].



**Fig. 4.** Ablation results. (a) is the segmented Gaussian center point, (b) is the original segmentation result without KNN or Filtering, (c) is the result after KNN, and (d) is the final result with KNN and Filtering

Both LLFF and NeRF-360 are centered on objects in the scene, with the difference that the camera viewpoint of the former varies in a small range, while the latter contains a 360° image around the object. Mip-NeRF 360 features an unbounded scene, and its camera viewpoint also varies in a large range. In the Gaussian Clustering stage, the probability threshold  $\beta$  of each 3D Gaussian is set at 0.65, while the 50 3D Gaussians closest to its distance are filtered for subsequent computation. The 3D Gaussians are built and trained on a single Nvidia GeForce RTX 3090 GPU.

#### 4.2. Result

Fig. 2 illustrates the segmentation effects of this method in various scenes. The first two rows demonstrate the segmentation performance when the camera position varies within a small range. The third row depicts the segmentation effect in a 360° scene. The final row highlights the results of multi-object segmentation, where distinct objects such as the TV, desk, and table are segmented separately. Our method’s efficiency is enhanced by the addition of object code, a simple yet effective tool for handling complex scenes. In the first row, this code enables the successful removal of complex background elements like the leaves behind the flower. In the third row, it ensures accurate segmentation even when there is a significant change in the viewing angle. Moreover, the object code, which encapsulates the probability distribution of the 3D Gaussian across all classes, facilitates the simultaneous segmentation of multiple objects in a scene.

Fig. 3 illustrates the comparative results between our method and ISRF [10]. Owing to the explicit representation of 3D Gaussians, our segmentation results are more precise in detail compared to those of ISRF, which is particularly evident in the leaf section of Fig. 3.

#### 4.3. Ablations

Fig. 4 presents the results of the ablation experiments, clearly demonstrating the effectiveness of KNN clustering and statistical filtering. Fig. 4(b) shows the initial segmented foreground object obtained without KNN clustering or statistical filtering, where it is noticeable that some leaves behind the

flower are erroneously segmented. Fig. 4(c) displays the segmented foreground image after KNN clustering. Since KNN primarily addresses Gaussians with ambiguous semantic information, its impact on the visualization result is minimal. However, it can be observed that some incorrectly segmented Gaussians have been removed. Finally, Fig. 4(d) shows the result obtained after applying both KNN clustering and statistical filtering, which successfully filters out those Gaussians that were incorrectly segmented.

## 5. CONCLUSION

We propose a 3D Gaussian segmentation method guided by 2D segmentation maps, attaching a probability distribution vector for each 3D Gaussian on various categories to enable the segmentation of the majority of 3D Gaussians in the scene. Meanwhile, we employ KNN clustering to utilize the spatial continuity of objects, ensuring that nearby 3D Gaussians belong to the same category. Additionally, optional statistical filtering is used to help remove those 3D Gaussians that are incorrectly segmented. As an initial step in 3D understanding and editing, this method has a wide range of potential applications in downstream tasks. We demonstrate the effectiveness of our method on common NeRF datasets.

## References

- [1] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis, “3D Gaussian Splatting for Real-Time Radiance Field Rendering,” *ACM Transactions on Graphics*, vol. 42, no. 4, pp. 1–14, July 2023.
- [2] Weiping Liu, Jia Sun, Wanyi Li, Ting Hu, and Peng Wang, “Deep learning on point clouds and its application: A survey,” *Sensors*, vol. 19, no. 19, 2019.
- [3] Dawar Khan, Alexander Plopski, Yuichiro Fujimoto, Masayuki Kanbara, Gul Jabeen, Yongjie Jessica Zhang, Xiaopeng Zhang, and Hirokazu Kato, “Surface remeshing: A systematic literature review of methods and research directions,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 3, pp. 1680–1713, 2022.
- [4] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove, “Deepsdf: Learning continuous signed distance functions for shape representation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [5] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng, “Nerf: Representing scenes as neural radiance fields for view synthesis,” *Commun. ACM*, vol. 65, no. 1, pp. 99–106, dec 2021.

- [6] Johannes L. Schonberger and Jan-Michael Frahm, “Structure-from-motion revisited,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [7] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke, “Gaussian Grouping: Segment and Edit Anything in 3D Scenes,” *arXiv e-prints*, p. arXiv:2312.00732, Dec. 2023.
- [8] Jiazhong Cen, Jiemin Fang, Chen Yang, Lingxi Xie, Xiaopeng Zhang, Wei Shen, and Qi Tian, “Segment Any 3D Gaussians,” *arXiv e-prints*, p. arXiv:2312.00860, Dec. 2023.
- [9] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann, “Decomposing nerf for editing via feature field distillation,” in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds. 2022, vol. 35, pp. 23311–23330, Curran Associates, Inc.
- [10] Rahul Goel, Dhawal Sirikonda, Saurabh Saini, and P. J. Narayanan, “Interactive segmentation of radiance fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 4201–4211.
- [11] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng, “DreamGaussian: Generative Gaussian Splatting for Efficient 3D Content Creation,” *arXiv e-prints*, p. arXiv:2309.16653, Sept. 2023.
- [12] Taoran Yi, Jiemin Fang, Junjie Wang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang, “GaussianDreamer: Fast Generation from Text to 3D Gaussians by Bridging 2D and 3D Diffusion Models,” *arXiv e-prints*, p. arXiv:2310.08529, Oct. 2023.
- [13] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 10684–10695.
- [14] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang, “4D Gaussian Splatting for Real-Time Dynamic Scene Rendering,” *arXiv e-prints*, p. arXiv:2310.08528, Oct. 2023.
- [15] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick, “Segment Anything,” *arXiv e-prints*, p. arXiv:2304.02643, Apr. 2023.
- [16] Bing Wang, Lu Chen, and Bo Yang, “DM-NeRF: 3D Scene Geometry Decomposition and Manipulation from 2D Images,” *arXiv e-prints*, p. arXiv:2208.07227, Aug. 2022.
- [17] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui, “Learning object-compositional neural radiance field for editable scene rendering,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 13779–13788.
- [18] Youtan Yin, Zhoujie Fu, Fan Yang, and Guosheng Lin, “OR-NeRF: Object Removing from 3D Scenes Guided by Multiview Segmentation with Neural Radiance Fields,” *arXiv e-prints*, p. arXiv:2305.10503, May 2023.
- [19] Jesus Zarzar, Sara Rojas, Silvio Giancola, and Bernard Ghanem, “SegNeRF: 3D Part Segmentation with Neural Radiance Fields,” *arXiv e-prints*, p. arXiv:2211.11215, Nov. 2022.
- [20] Ashkan Mirzaei, Tristan Aumentado-Armstrong, Konstantinos G. Derpanis, Jonathan Kelly, Marcus A. Brubaker, Igor Gilitschenski, and Alex Levinshtein, “Spin-nerf: Multiview segmentation and perceptual inpainting with neural radiance fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 20669–20679.
- [21] Zhenxing MI and Dan Xu, “Switch-neRF: Learning scene decomposition with mixture of experts for large-scale neural radiance fields,” in *The Eleventh International Conference on Learning Representations*, 2023.
- [22] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin, “Emerging properties in self-supervised vision transformers,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 9650–9660.
- [23] Yuying Hao, Yi Liu, Yizhou Chen, Lin Han, Juncai Peng, Shiyu Tang, Guowei Chen, Zewu Wu, Zeyu Chen, and Baohua Lai, “Eiseg: An efficient interactive segmentation tool based on paddlepaddle,” *arXiv e-prints*, pp. arXiv–2210, 2022.
- [24] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortíz-Cayón, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar, “Local light field fusion: Practical view synthesis with prescriptive sampling guidelines,” *ACM Trans. Graph.*, vol. 38, no. 4, jul 2019.
- [25] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman, “Mip-nerf 360: Unbounded anti-aliased neural radiance fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 5470–5479.