FD-3DGS: Flexible Disentangled 3DGS for Scenes Understanding and Manipulation

Shiyao Xu
Cybever

Junlin Han†
University of Oxford

Jie Yang
Cybever

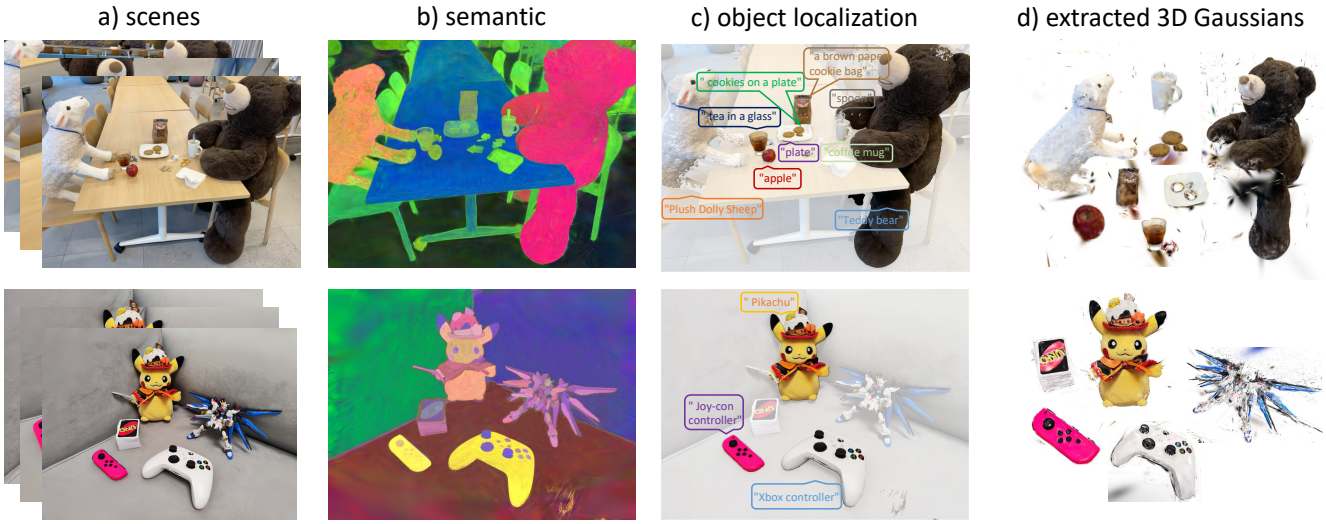

Figure 1. We present FD-3DGS, where we utilize the position attribute of 3D Gaussians to generate the corresponding semantic feature flexibly. Our model can perform queries based on the learned semantic information, not only at the rendered views (c) but more importantly, we can directly manipulate 3D Gaussian points (d) according to their semantic features (b).

Abstract

Understanding 3D scenes is crucial for various applications, but remains a challenging task. Existing methods, even those using explicit 3D representations like 3D Gaussian Splatting (3DGS), often fail to fully leverage the richness of these representations for comprehensive scene understanding and manipulation. We present a novel method, FD-3DGS, that builds a "plug-and-play" semantic field for 3DGS. This field leverages the inherent property of 3D Gaussians, where nearby Gaussians share similar semantic meanings. By employing a multi-resolution hash encoding of Gaussian coordinates, we extract language information from powerful models like CLIP and effectively integrate it into the 3D scene. Through a carefully designed training process involving improved feature rendering and enhanced contrastive learning, FD-3DGS disentangles and refines semantic information within the 3DGS representation. This results in high-quality semantic features that enable more accurate scene understanding and support direct querying of individual 3D Gaussian points. Compared to existing approaches, FD-3DGS achieves higher accuracy in tasks like language-based object extraction, demonstrating the effectiveness of our method for 3D scene understanding.

1. Introduction

3D scene understanding is a classic task in vision and graphics and has been widely used in autonomous driving, human-object interaction, etc. Given a set of 3D representations or multi-view images of a scene, the goal of the task is to learn the semantic information of the scene along with the reconstruction process. Previous work [17,21,23,35,39] has integrated semantic information from pretrained vision foundation models such as CLIP [30] and DINO [5] into Neural Radiance Fields (NeRF) [25] so that they can generate semantic maps for 3D scenes and support language-

†Work completed while at Cybever

based scene editing. However, these methods are limited by the continuous implicit 3D representations, there remains potential for further exploration in optimizing both generation speeds and the quality of the resulting semantic features. More recently, the emergence of 3D Gaussian Splatting [16] has brought 3D reconstruction into a new era. 3D Gaussian Splatting represents scenes with collections of 3D Gaussians and utilizes the tile-based splatting solution, which allows competitive training time and high-quality real-time rendering for large-resolution datasets.

Subsequently, works on 3D Gaussians-based scene understanding and semantic feature generation emerged. Most of these [14, 29, 32, 38, 42, 43, 45] introduce an extra semantic feature into the 3D Gaussians and then do splatting and rasterizing just the same as the color rendering procedure to obtain the attached semantic feature in a given pose. However, due to the differential raster rendering used in 3DGS, they utilize shared memory in cuda to update the parameters. When introducing a high dimensional semantic feature into 3D Gaussian, such as the 512-dimensional CLIP feature, it will significantly increase the memory and time cost for the total training procedure. Existing methods [29, 32] utilize some additional pertained decoders and reduce the introduced feature dimension to avoid this memory limitation. Although this can maintain a faster training and rendering speed for language field generation, it often requires a complex and time-consuming pre-training process for each scene. For an individual scene, it needs more time than building the semantic feature field to compress the feature space. At the same time, this additional encoder-decoder [29] or quantified vocabulary learning for language features [32] often introduces noise into the semantic field and hinders the object understanding and targeting tasks with high accuracy.

Building upon the properties of 3DGS, we introduce a novel method, Flexible Disentangled 3DGS **(FD-3DGS)**, for scene understanding and manipulation. Instead of relying on cumbersome pre-training, FD-3DGS employs a direct, plug-and-play semantic feature field designed specifically for 3D Gaussians. Our approach stems from the simple intuition that points in close proximity should share similar semantic properties. We leverage pre-constructed 3D Gaussians and a plugged multi-resolution hash encoding to learn semantic features, then utilize an improved point-based rendering technique to generate a smoother feature map. To guide this learning process, we employ the pre-trained CLIP model [30] and a Segment Anything Model (SAM) [20] to obtain pixel-aligned features. This allows us to extract object-specific CLIP features and unproject them into image-aligned space for training. Recognizing that high-dimensional CLIP features can introduce noise, we generate lower-dimensional semantic features with more aggregation and then use a 1×1 convolution layer to align them to the 512-dimensional CLIP space. Furthermore, we introduce progressive contrastive learning to enhance feature quality, resulting in clearer object boundaries and more concentrated features within object types. This facilitates object understanding and manipulation tasks. With FD-3DGS, object extraction becomes a simple query based on semantic feature attributes, allowing direct access to the underlying 3D Gaussian representation for subsequent operations. This streamlined and efficient approach significantly improves scene understanding and manipulation capabilities.

In summary, the core contributions of our work are:

- We simplify the process of semantic feature generation and directly use the 3D Gaussians to learn the corresponding feature for each point, eliminating the need for cumbersome pre-processing, post-processing, and additional training processes.
- We generate the high-quality semantic features distilled from current pretrained language models, such as CLIP, with a competitive speed (around 40 minutes), and making the deployment and actual usage of 3DGS-based scene understanding easier and more friendly. The overall query speed of our models reached 30-60 seconds.
- Our FD-3DGS achieves semantic understanding and object extraction that can be directly performed on the 3D Gaussian points for the first time. The proposed plug-and-play semantic feature field can be inserted into any 3DGS models.

2. Related Work

2.1. 3D Scene Understanding

3D scene understanding intends to distill the language information into 3D scenes. Previous work introduced a feature field branch into the Neural Radiance Field [25] and rendered features together with images. DFF [21], N3F [35], and F3RM [31] extended the generation of feature fields to the unlabeled dataset. They distilled LSeg [22] and DINO features [5] from the multiview images to the feature field and therefore they can perform simple feature-based queries and understanding on the field. 3D-OVS [23] and LERF [17] later distilled the more powerful CLIP features into NeRF to achieve open-vocabulary scene understanding. While LERF considered the multiple physical scales of objects could achieve a fine result. With the emergence of 3D Gaussian Splatting, subsequent work [14, 29, 32, 38, 42, 43, 45] attempts to utilize this powerful, efficient model for 3D scene understanding. They inherited previous operations on NeRF and added semantic features corresponding to the color attribute in 3D Gaussians. When introducing high-dimensional features into the attributes of

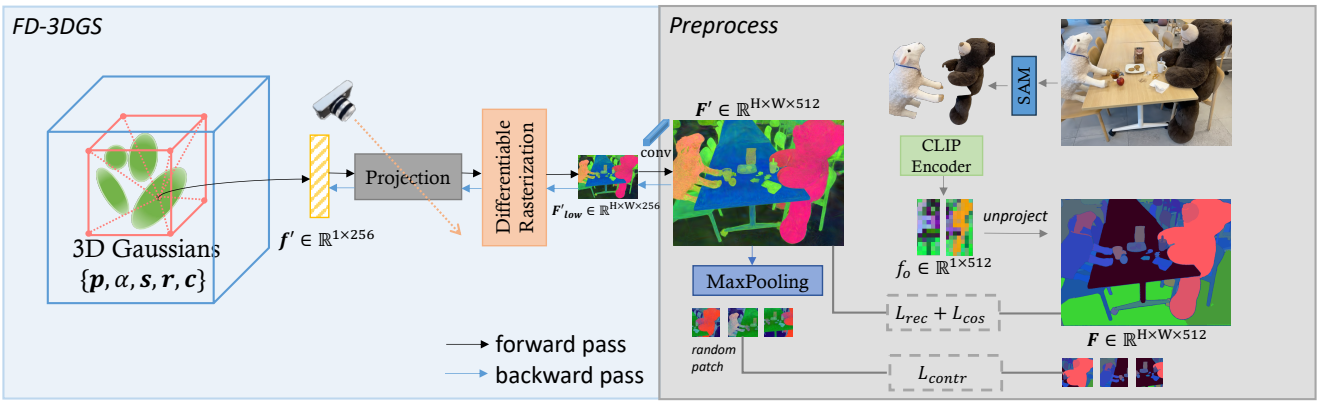

Figure 2. Overview of our FD-3DGS. Our model starts from an off-the-shell 3DGS and we fix all attributes in 3DGS, only using coordinates p to learn the semantic information. We utilize the CLIP encoder to extract the object-level semantic information and use SAM to unproject the features back into the pixel space to supervise the generation of semantic feature grids.

3D Gaussians, it faces the problem that splat-based rasterization rendering of high-dimensional will cause OOM error and unacceptable training time. Therefore, these methods tried first to compress the dimensions of CLIP and DINO features through additional encoder-decoder or discretized vector encoding, then they do the rendering to obtain the feature map. Although the reduced dimension does speed up the semantic learning process, they are still limited in 2D space operation, rendering the semantic feature for multi-views, performing retrieval operations, and stitching different views together to get the final result, which is considered as a pseudo-3D result. Different from these works, we learn the low dimensional semantic features from the pre-constructed 3D Gaussian points and optimize the plugged semantic field. During the optimization process, we utilize a lightweight convolution to map the low-dimensional features to the standard high-dimensional semantic space and later invert the query feature back to 3D Gaussians, to achieve direct 3D manipulation.

2.2. 3D Scene Segmentation and Manipulation

3D scene segmentation aims to estimate the category or generate the segmented mask of each point in the scene. Current work [3, 7, 9, 12, 33, 34, 36, 40] usually introduce segmentation models into 3D scenes. SemanticNeRF [40] predict the semantic label together with images from the labeled dataset. OpenNeRF [9] distilled OpenSeg [13] into NeRF for 3D segmentation. SA3D [7] and SAGA [6] leverage the interactive prompt segmentation of SAM and introduce SAM into NeRF and 3DGS respectively. GaussianGrouping [37] introduced Identity Encodings into 3D Gaussians in order to maintain the consistency of segmentation results in different views. GARField [18] token physic scale into account when segmentation. After segmenting the target objects or regions from the 3D scene, works [8, 10] also tried to manipulate the target regions using InstructP2P [4, 15], SDS constrains [28], etc. These methods require the maintenance of additional classifiers to map the feature into K categories and re-render the semantic feature for retrieval in the mapping space. In essence, they are operating in the rendered 2D space, which requires tedious post-processing. Different from these works, we avoid the extra mapping process, no need for re-rendering or semantic tracing, and can directly manipulate in the 3D Gaussian space.

3. Method

We aim to avoid the cumbersome feature-pertaining methods and propose a simple yet efficient semantic feature learning model corresponding to 3D Gaussian to facilitate scene understanding and object manipulation. Therefore, we introduce a plug-in-play feature space into 3D Gaussians and propose our flexible disentangled 3DGS. In this section, we will first review the principles of 3DGS in Sec. 3.1, then we will detail how we construct our FD-3DGS, including the image preprocessing in Sec. 3.2, design for the insert feature learning from the 3D points based on a multiresolution hash encoding in Sec. 3.3, and the feature enhancement procedure for contrastive and aggregated learning in Sec. 3.4. Finally, we show how to unproject the semantic information into our feature space therefore to query directly on the 3D Gaussians in Sec. 3.5.

3.1. Preliminary: 3D Gaussian Splatting

3D Gaussian Splatting, first introduced by [16], represents a static scene using a set of colored 3D Gaussians. Each Gaussian is defined by the 3D position $\mathbf{p} = \{x, y, z\} \in \mathbb{R}^3$, a 3D scaling vector $\mathbf{s} \in \mathbb{R}^3$ in standard deviations, the rotational quaternion $\mathbf{r} \in \mathbb{R}^4$, and opacity α for the tile-based splitting.

In addition to the attributes $\{\mathbf{p}, \alpha, \mathbf{s}, \mathbf{r}\}$, 3DGS also op-

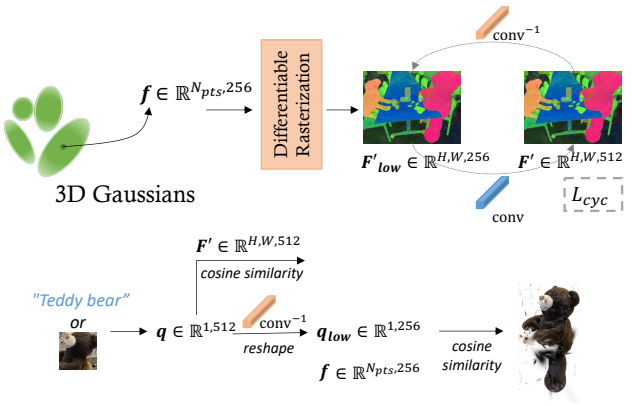

Figure 3. The learning process of the inverse convolution and 3D Gaussians extraction process. When querying, the input embedding is first mapped to the compressed CLIP space, then calculate the similarity with features in 3D Gaussians.

timizes spherical harmonics (SH) coefficients representing color $\mathbf{c} \in \mathbb{R}^k$ (where k indicates the degrees of freedom) of each Gaussians to capture the view-dependent appearance of the scene. It usually uses three degrees of SH coefficients to represent color $\mathbf{c} \in \mathbb{R}^3$. For each pixel in the views of a scene, it follows the tile-based rendering that allows $\alpha-$blending of anisotropic splats respecting visibility orders to calculate the color and the formulation is as follows:

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

where c_i and α_i denote the color and density of a given point respectively and the value is determined by the optimizable per-point attributes of each Gaussian.

3.2. Image Preprocessing for semantic feature preparation

Before we introduce the design of our FD-3DGS, we must describe how we prepare the input images to distill the semantic information from the CLIP feature extractor [30]. Current works such as LERF [17] and GARField [18] take the ambiguity of 3D points into account, which means the same point could have different meanings at different scales, and introduce physical scale into semantic generation as a condition. They extract CLIP and DINO [5] embeddings from multiple views over multiple scales and then render the semantic features together with images. In any case, what we believe essentially and efficiently for open-world scene understanding is the representative and useful local features so it can better represent instance-level objects. Therefore, we are dedicated to improving the quality of rendered semantic features.

Since CLIP is inherently a global image embedding and not conducive to pixel-aligned feature extraction, we propose to unproject the CLIP feature backing to the image-aligned feature space. For the multi-view input images $\{I_1, ..., I_N\}$, we first leverage SAM [20] to automatically obtain the total number of N_{obj} masked objects in the given view I_i and the segmentation map $S_i \in \mathbb{R}^{H \times W}$ (where H and W indices the image size) according to the N_{obj}. Then we calculate CLIP feature $\{f_o \in \mathbb{R}^{512} | o = 0, ..., N_{obj}\}$ for each object, and finally we could unproject the object-level CLIP features $F_i \in \mathbb{R}^{H \times W \times 512}$ back to the image-size, that is the pixel-align, according to the segmentation map S_i for each image.

It is worth noting that the CLIP feature extracted here for open-world understanding is actually very rough since the pre-trained 512-dimensional CLIP feature contains a lot of noise and redundancy information, which will hinder the quality of generated features and thus the accuracy of object queries and understanding. Therefore, in the following sections, we will introduce in detail how we reduce the interference of noise and make the boundaries between different objects of semantic features clearer.

3.3. FD-3DGS

The goal of our method is to distill the preprocessed feature into 3D Gaussians, simplify the process of semantic feature generation, improve the quality, and facilitate scene understanding, therefore we introduce a simple yet efficient feature-embedded 3D Gaussians. Different from previous work, we did not directly add a feature attribute to the 3D Gaussian point, but we utilized the priors of pre-constructed 3D Gaussians, that points at similar locations should have similar semantic properties, to insert a plug-and-play voxel-based semantic feature grid into 3D Gaussians. The overview of our model can be seen in Fig. 2.

Given a 3D Gaussian, we use a **M**ultiresolution **H**ash **E**ncoding proposed in InstantNGP [26] to generate the semantic feature f' from the coordinate $\mathbf{p}$ of Gaussian points. Compared with the direct learning features method, which will cause the OOM issue and the obvious increased training time, our method could make the feature more aggregated, and the time for semantic feature f' query in hash table is only $O(1)$. The process is as follows:

$$f' = MHE(p) \tag{2}$$

In MHE, we first perform trilinear interpolation in the voxel grids of semantic feature according to the input position $\mathbf{p}$ to obtain the nearest voxels. For each voxel, it is mapped to a feature vector of D-dimension in the hash table. Then we concatenate the queried vectors together, and obtain our final feature $f' \in \mathbb{R}^{d_e}$ through a one-layer MLP. Each level's hash table contains E vectors of dimension D, and the resolution of each level is calculated by $N_l = \lfloor N_{min} \cdot b^l \rfloor$ where N_{min} is the coarsest resolution, b is the growth factor and l notes the level. Details of the MHE setting can be found in Sec. 4.

To generate the pixel-align 2D feature map, we follow the point-based $\alpha-$blending same as the color calculation in Eq. 1 using the queried feature f' and the formulation is as follows:

$$F' = \sum_{i \in N} f'_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j) \tag{3}$$

where f'_i here is what we calculate according to the MHE. Note that the CLIP features we used actually have 512 dimensions. To avoid the OOM errors during rasterization, we modified the backpropagation process of cuda rasterization, that is, returning the gradient of the feature through a manually allocated buffer instead of the shared memory. This allocated buffer will also slow down the rasterization. However, considering that high-dimensional features contain too much noise in the scene understanding task as we discussed before, finally, we set the feature f in MHE to 256-dimension and get the semantic feature $F' \in \mathbb{R}^{H \times W \times 256}$. Then we introduce the 1×1 convolution to align the rendered $F' \in \mathbb{R}^{H \times W \times 256 \to 512}$ to the GT CLIP feature F.

3.4. Feature Enhancement Procedure

As we mentioned before, the key to high-accuracy scene understanding is a high-quality semantic feature. For the features generated according to Eq. 3, the boundaries of objects are not clear, and it hinders object extraction and manipulation. In this section, we present our training process and how to improve the quality of semantic features, especially a clear object boundary.

As is shown in Fig. 2, our model is based on a set of learned 3D Gaussians following the same setting as the original paper [16] and then we introduce the feature field into 3D Gaussians. We fix all attributes of the constructed 3D Gaussians, but give the position $\mathbf{p} = (x, y, z)$ with gradient to learn the feature voxel-grid and the 1×1 convolution layer as we mentioned in Sec. 3.3. To optimize the total network, we designed the feature reconstruction loss (similar to the reconstruction of color in the original 3DGS):

$$L_{rec} = \sum_{i \in N_{img}} ||F'_i - F_i||_2^2 \tag{4}$$

We also calculate the cosine similarity of the generated feature and the GT feature inspired by OpenScene [27] to ensure the network output F' more consistent with the GT feature:

$$L_{cos} = 1 - cos(F', F) \tag{5}$$

Remember what we mentioned in Sec. 3.2, different objects in the generated features are prone to blurred boundaries. Similar to the contrastive supervision proposed in GARField [18], our goal is also to make the features of the same object closer and the features of different objects farther apart. Also inspired by PanopticLifting [33] and ContrastiveLift [2], we intend to enhance the contrastive of generated features through the following loss function. Unlike the NeRF-based method, our FD-3DGS cannot render rays and features by batches and calculate the feature contrast within a batch of rays in the rendering process, we therefore designed additional scaling processing for the rendered features. Specifically, to avoid the OOM problem when loss calculation, we first compress the generated feature $F'_{low} \in \mathbb{R}^{H \times W \times 256}$ with Max Pooling, and then we randomly pick a patch on the compressed feature, and perform the following contrastive loss calculation on this patch:

$$L_{contr} = -\frac{1}{N_p} log \frac{\mathbb{1}_{F_A = F_B} \cdot \mathrm{sim}(F'_{A'}, F'_{B'}; \gamma)}{\mathrm{sim}(F'_{A'}, F'_{B'}; \gamma)} \tag{6}$$

where N_p denotes the selected patch size, $\mathbb{1}$ is the indicator function to select features of the same category from F to supervise the generated F', and $\mathrm{sim}(\mathrm{A}, \mathrm{B}; \gamma) = exp(-\gamma||A-B||^2)$ is the Gaussian RBF kernel used to compute the similarity of generated feature F'.

Through this additional scaling processing, compared with the direct patch-picking method, we enlarge the window of contrastive calculation and avoid the poor results led by the localized areas. Also, L_{contr} makes the feature clusters with sharp boundaries.

In addition, we also introduce total variation loss L_{tv} as a regularization term to encourage spatial smoothness for the generated semantic features F' while preserving the semantic boundaries.

$$L_{tv} = \frac{1}{H * W * C} ||\nabla_u(F') + \nabla_v(F')|| \tag{7}$$

where u and v denote horizontal and vertical directions respectively.

Overall, the total training loss is defined as:

$$L = \lambda_{rec} L_{rec} + \lambda_{cos} L_{cos} + \lambda_{contr} L_{contr} + \lambda_{tv} L_{tv} \tag{8}$$

We utilize a progressive training strategy for semantic features, beginning with reconstruction and gradually incorporating contrastive constraints to prevent scattered results.

3.5. Query-based Decomposition and 3D Gaussians Manipulation

Since only the CLIP feature is used to learn the feature field, a trained FD-3DGS can be used to perform 3D zero-shot segmentation by image or text queries. When querying, existing methods can only calculate the relevancy score in the generated feature map and return the target object masks, which we think they didn't really query on 3D space. Different from them, we propose to not only obtain the selected object mask according to the rendered features but also support to query directly on the 3D Gaussian points.

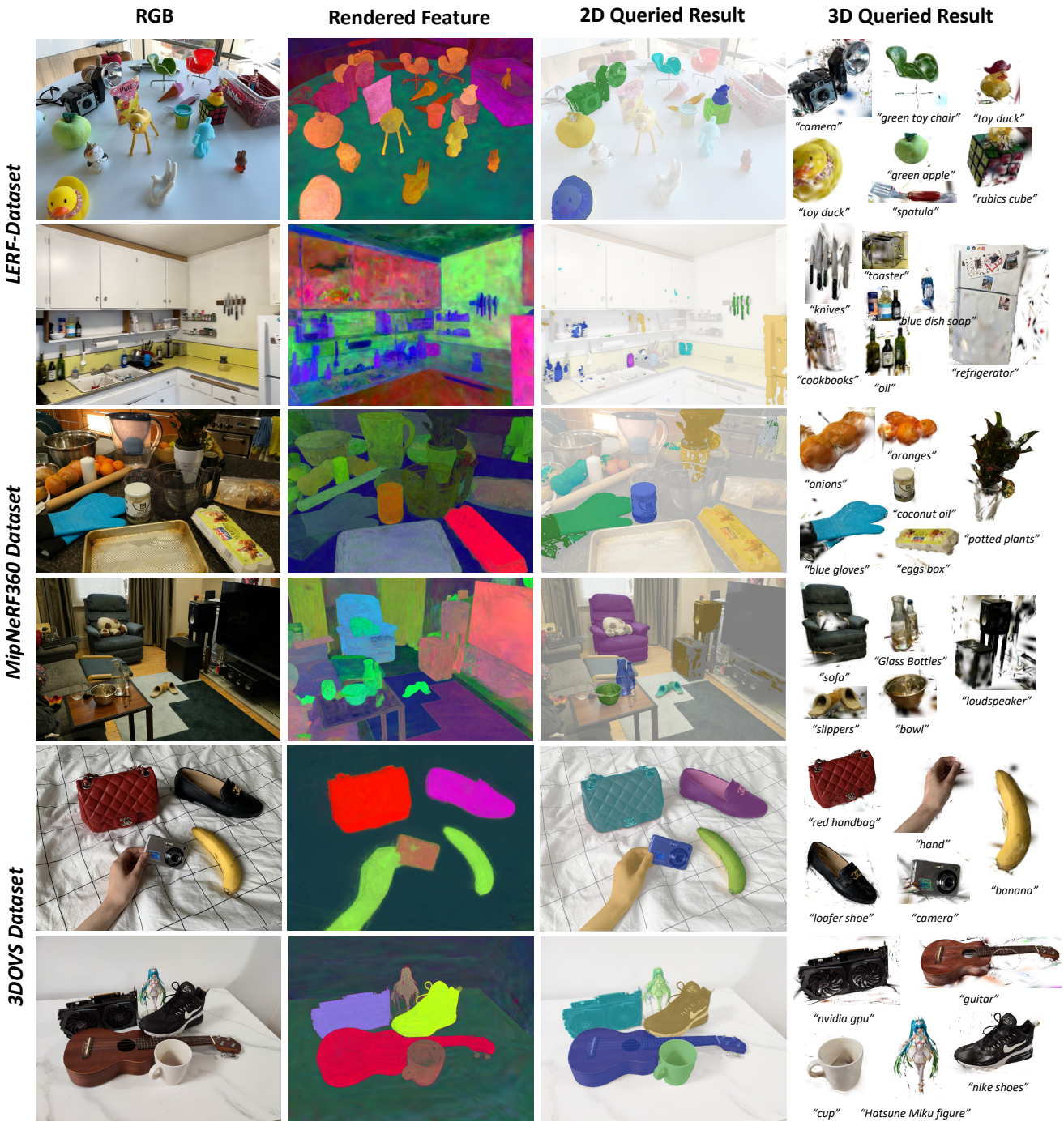

Figure 4. Results of our FD-3DGS on LERF [17], MipNeRF360 [1], and 3DOVS [23] datasets. We show the PCA visualization of our semantic features, queried results on the 2D rendered semantic features, and the results directly on 3D Gaussian points. Our model can not only query on the rendered semantic features but also directly manipulate and extract the 3D Gaussian points corresponding to the object.

As is shown in Fig. 3, when querying, we first get the CLIP feature $q \in \mathbb{R}^{512}$ of the input prompt, image or language, then we could easily get the relevancy score of each pixel in the feature map by computing the cosine similarity $score = \frac{F' \cdot q}{||F' \cdot q||}$ between the rendered features and the query vector. We then filter out the target object by a user-specified threshold.

However, this is still performed on the rendered image plane. Due to the characteristics of the explicit structure of 3D Gaussian Splatting, we want to directly get the 3D

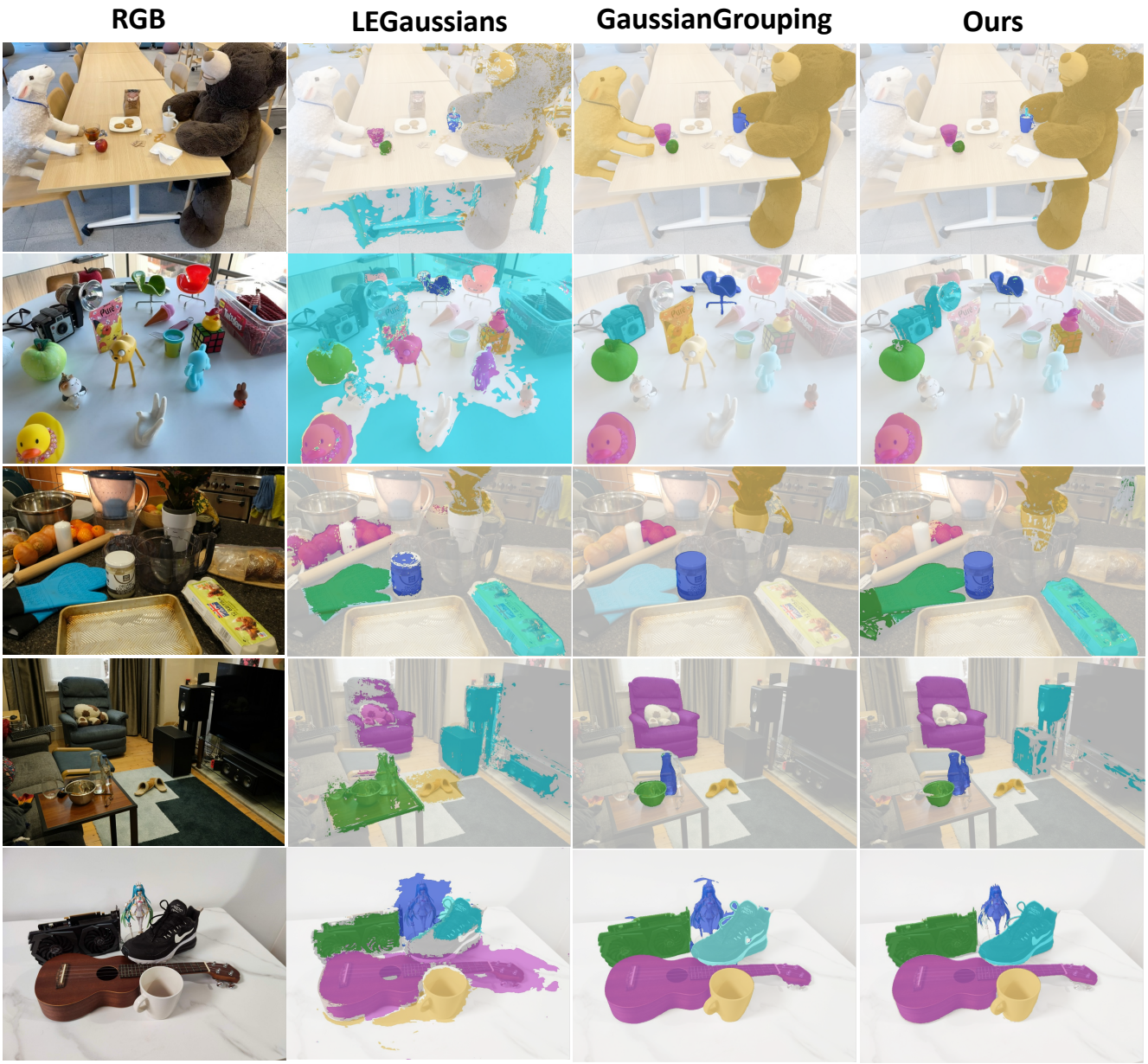

Figure 5. Comparison Results with the latest methods LEGaussians and GaussiansGrouping. We use the same text prompt as in Fig. 4 and query on the rendered images. It is clear that our semantic fields inserted on 3D Gaussians can extract more complete objects, even those parts that are occluded from the rendering view can be extracted together in 3D space.

Gaussian points, which can later be used for subsequent applications. Different from the semantic tracing proposed in GaussianEditor [10], what we need is just to unproject the dense feature F'_{low} back to the original CLIP feature space. Inspired by Glow [19], we also want to learn the inverse of 1×1 convolution for the feature transformation after rasterization. In this way, we can map the features associated with the position of 3D Gaussian points before rasterization into the original CLIP feature space. We add an optional pseudo inverse 1×1 convolution layer during training as is shown in Fig. 3. During training, we can learn this inverse convolution using Cycle Loss [44]:

$$L_{cyc} = ||\text{conv}^{-1}(F') - F'_{low}||_1 + ||\text{conv}(F'_{low}) - F'||_1 \quad (9)$$

To speed up the invert layer learning, we usually fix the $conv$ layer and update $conv^{-1}$ separately. Then we unproject the query vector $q \in \mathbb{R}^{1\times 512}$ to the compressed feature space q_{low} to directly perform the query on 3D Gaussian points using the cosine similarity same as what we do on the rendered features.

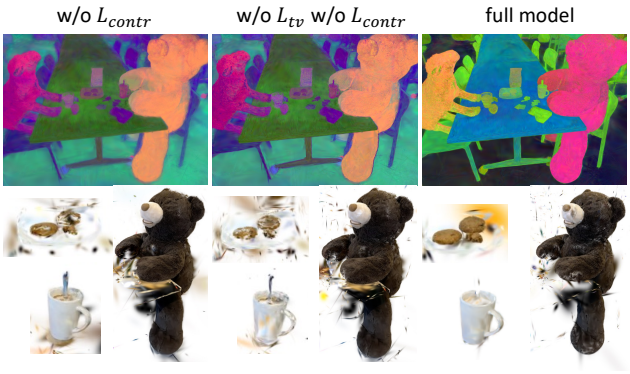

Figure 6. Ablation Results: We drop contrastive learning and TV regularization mentioned in Sec. 3.4 for comparison. The top row shows the PCA visualization of semantic features and the bottom shows the results directly on 3D Gaussians using "cookies on a plate", "coffee mug", and "Teddy Bear" in *teatime* scene.

4. Implementation Details

In image preprocessing (Sec. 3.2), we utilize SAM with ViT-H settings to extract objects from the multi-view images, followed by CLIP to obtain the image-aligned embeddings. Notably, CLIP here can be replaced with improved models like MaskCLIP [41], which we use with a FeatUp [11] layer under ViT-B settings for sharper boundaries. Our FD-3DGS could directly insert the semantic feature field into any pre-constructed 3DGS. In this paper, we utilize pretrained original 3D Gaussian Splatting in default settings for 30,000 iterations. For our FD-3DGS, we fix all 3DGS attributes and only train the feature field and the convolution layer for 1500 iterations. The feature field employs multiresolution hash encoding with 16 levels (resolutions from 16 to 128), a hash table size of 2^{20} and 8 feature dimensions per entry. After encoding, a FullyFused MLP with a 128-dimensional hidden layer and ReLU.

We introduce contrastive loss by first applying MaxPooling to the generated features, then randomly selecting features with 32 patch sizes. When calculating the similarity of generated feature F' in Euclidean space, we set the coefficient $\gamma = 1.0$ for the RBF kernel. Contrastive loss is introduced progressively, starting at 800 iterations. To balance the weight of CLIP feature, we set λ_{rec} to 1000 and the rest to 1. Experiments are conducted on an NVIDIA A100 GPU (40GB RAM). The training takes **45 minutes** on a dataset of 180 images at 986 $\times$ 728 resolution, faster than previous methods.

5. Experiments

5.1. Setups

Dataset: We select several scenes from Mip-NeRF360 dataset [1] ("room", "counter"), LERF [17] ("teatime", "figurines", "ramen", "waldo_kitchen") and 3D-OVS [23] ("sofa", "table"). Each scene contains 180 to 320 images captured in a wide range of views in MipNeRF360 and LERF, and about 30 images in 3D-OVS.

Test Scene	LEGaussian [32]		GaussianGrouping [37]		Ours	
	mIoU(%)	mBIoU(%)	mIoU(%)	mBIoU(%)	mIoU(%)	mBIoU(%)
figurines	51.2	44.6	56.5	52.9	**91.8**	**83.4**
teatime	43.3	42.8	66.1	63.2	**80.1**	**76.3**
counter	59.4	41.3	58.7	52.9	**82.1**	**65.9**
room	21.2	16.3	**73.7**	**71.1**	70.5	69.0
table	59.2	31.2	79.7	69.6	**95.1**	**87.5**

Table 1. Comparison of open vocabulary segmentation on LERF [17], MipNeRF360 [1], and 3D-OVS [23] dataset. We use the segmentation results of GroundingDINO [24] as GT to calculate mIoU and mBIoU. Our model performs better in most cases.

Result: We compared our model directly with LEGaussians [32] and GaussianGrouping [37] for object extraction and scene segmentation. To obtain quantitative results, we use the segmentation results of GroundingDINO in test views as GT and calculate the mIoU, mBIoU in Table 1. The results in Fig. 5, Fig. 4 and Table 1 demonstrate the performance of our method. In most cases, our model performs better in both quality and quantity. In *room* scene, both the results of GroundingDINO and other models cannot identify the occluded bottles (blue part in *room* in Fig. 5), but ours can extract the occluded part. Therefore, the quantitative result in this scenario is slightly lower, but we think our result is more reasonable. Fig. 4 shows that our method can direct query and manipulate on the 3D Gaussians.

5.2. Ablation Study

To demonstrate the effectiveness of our feature enhancement procedure in Sec. 3.4, we mainly conduct the ablation study on our contrastive learning and the total variation regularization. We set up two experiments, no contrastive learning, and no contrastive learning + TV regularization in Fig. 6. It can be clearly seen that the boundary of objects without contrastive learning and TV regularization is fuzzy, and extracted 3D Gaussians even have missing parts. For queries on small objects, such as *"spoon handle"* and *"coffee mug"*, it is obvious that the complete model is more sensitive and can separate the *"spoon handle"* from the *"coffee mug"*, which demonstrate the effectiveness of the mentioned procedure and regularization term.

6. Conclusion

We present FD-3DGS, a flexible disentangled 3DGS for scene understanding and object manipulation. We propose a simplified semantic field that can be inserted into any 3DGS. The most important is that for the first time, we allow the semantic-based 3D editing to be performed directly on the explicit 3D Gaussians, unlocking a new level of control and precision for interacting with 3D Gaussians, bringing more possibilities to subsequent applications.

References

[1] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 6, 8

[2] Yash Bhalgat, Iro Laina, João F Henriques, Andrew Zisserman, and Andrea Vedaldi. Contrastive lift: 3d object instance segmentation by slow-fast contrastive fusion. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 5

[3] Wang Bing, Lu Chen, and Bo Yang. Dm-nerf: 3d scene geometry decomposition and manipulation from 2d images. *arXiv preprint arXiv:2208.07227*, 2022. 3

[4] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023. 3

[5] 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 International Conference on Computer Vision (ICCV)*, 2021. 1, 2, 4

[6] Jiazhong Cen, Jiemin Fang, Chen Yang, Lingxi Xie, Xiaopeng Zhang, Wei Shen, and Qi Tian. Segment any 3d gaussians. *arXiv preprint arXiv:2312.00860*, 2023. 3

[7] Jiazhong Cen, Zanwei Zhou, Jiemin Fang, Chen Yang, Wei Shen, Lingxi Xie, Xiaopeng Zhang, and Qi Tian. Segment anything in 3d with nerfs. In *NeurIPS*, 2023. 3

[8] Yiwen Chen, Zilong Chen, Chi Zhang, Feng Wang, Xiaofeng Yang, Yikai Wang, Zhongang Cai, Lei Yang, Huaping Liu, and Guosheng Lin. Gaussianeditor: Swift and controllable 3d editing with gaussian splatting, 2023. 3

[9] Francis Engelmann, Fabian Manhardt, Michael Niemeyer, Keisuke Tateno, Marc Pollefeys, and Federico Tombari. OpenNeRF: Open Set 3D Neural Scene Segmentation with Pixel-Wise Features and Rendered Novel Views. In *International Conference on Learning Representations*, 2024. 3

[10] Jiemin Fang, Junjie Wang, Xiaopeng Zhang, Lingxi Xie, and Qi Tian. Gaussianeditor: Editing 3d gaussians delicately with text instructions. In *CVPR*, 2024. 3, 7

[11] Stephanie Fu, Mark Hamilton, Laura E. Brandt, Axel Feldmann, Zhoutong Zhang, and William T. Freeman. Featup: A model-agnostic framework for features at any resolution. In *The Twelfth International Conference on Learning Representations*, 2024. 8

[12] Xiao Fu, Shangzhan Zhang, Tianrun Chen, Yichong Lu, Lanyun Zhu, Xiaowei Zhou, Andreas Geiger, and Yiyi Liao. Panoptic nerf: 3d-to-2d label transfer for panoptic urban scene segmentation. In *International Conference on 3D Vision (3DV)*, 2022. 3

[13] Golnaz Ghiasi, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. Scaling open-vocabulary image segmentation with image-level labels. In *Computer Vision – ECCV 2022*, pages 540–557, Cham, 2022. Springer Nature Switzerland. 3

[14] Jun Guo, Xiaojian Ma, Yue Fan, Huaping Liu, and Qing Li. Semantic gaussians: Open-vocabulary scene understanding with 3d gaussian splatting. *arXiv preprint arXiv:2403.15624*, 2024. 2

[15] Ayaan Haque, Matthew Tancik, Alexei Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 3

[16] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. 2, 3, 5

[17] Justin* Kerr, Chung Min* Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *International Conference on Computer Vision (ICCV)*, 2023. 1, 2, 4, 6, 8

[18] Chung Min* Kim, Mingxuan* Wu, Justin* Kerr, Matthew Tancik, Ken Goldberg, and Angjoo Kanazawa. Garfield: Group anything with radiance fields. In *arXiv*, 2024. 3, 4, 5

[19] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018. 7

[20] 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:2304.02643*, 2023. 2, 4

[21] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing nerf for editing via feature field distillation. In *Advances in Neural Information Processing Systems*, volume 35, 2022. 1, 2

[22] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and Rene Ranftl. Language-driven semantic segmentation. In *International Conference on Learning Representations*, 2022. 2

[23] Kunhao Liu, Fangneng Zhan, Jiahui Zhang, Muyu Xu, Yingchen Yu, Abdulmotaleb El Saddik, Christian Theobalt, Eric Xing, and Shijian Lu. Weakly supervised 3d open-vocabulary segmentation. *arXiv preprint arXiv:2305.14093*, 2023. 1, 2, 6, 8

[24] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023. 8

[25] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2

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

[27] Songyou Peng, Kyle Genova, Chiyu "Max" Jiang, Andrea Tagliasacchi, Marc Pollefeys, and Thomas Funkhouser. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 5

[28] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *The*

Eleventh International Conference on Learning Representations, 2022. 3

[29] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. *arXiv preprint arXiv:2312.16084*, 2023. 2

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1, 2, 4

[31] William Shen, Ge Yang, Alan Yu, Jansen Wong, Leslie Pack Kaelbling, and Phillip Isola. Distilled feature fields enable few-shot language-guided manipulation. In *7th Annual Conference on Robot Learning*, 2023. 2

[32] Jin-Chuan Shi, Miao Wang, Hao-Bin Duan, and Shao-Hua Guan. Language embedded 3d gaussians for open-vocabulary scene understanding. *arXiv preprint arXiv:2311.18482*, 2023. 2, 8

[33] Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulò, Norman Müller, Matthias Nießner, Angela Dai, and Peter Kontschieder. Panoptic lifting for 3d scene understanding with neural fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9043–9052, June 2023. 3, 5

[34] Ayça Takmaz, Elisabetta Fedele, Robert W. Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann. OpenMask3D: Open-Vocabulary 3D Instance Segmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 3

[35] Vadim Tschernezki, Iro Laina, Diane Larlus, and Andrea Vedaldi. Neural Feature Fusion Fields: 3D distillation of self-supervised 2D image representations. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2022. 1, 2

[36] Suhani Vora, Noha Radwan, Klaus Greff, Henning Meyer, Kyle Genova, Mehdi S. M. Sajjadi, Etienne Pot, Andrea Tagliasacchi, and Daniel Duckworth. Nesf: Neural semantic fields for generalizable semantic segmentation of 3d scenes, 2021. 3

[37] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. *arXiv preprint arXiv:2312.00732*, 2023. 3, 8

[38] Daiwei Zhang, Gengyan Li, Jiajie Li, Mickaël Bressieux, Otmar Hilliges, Marc Pollefeys, Luc Van Gool, and Xi Wang. Egogaussian: Dynamic scene understanding from egocentric video with 3d gaussian splatting. *arXiv preprint arXiv:2406.19811*, 2024. 2

[39] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J. Davison. In-place scene labelling and understanding with implicit scene representation. In *ICCV*, 2021. 1

[40] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J. Davison. In-place scene labelling and understanding with implicit scene representation. In *ICCV*, 2021. 3

[41] Chong Zhou, Chen Change Loy, and Bo Dai. Extract free dense labels from clip. In *European Conference on Computer Vision (ECCV)*, 2022. 8

[42] Hongyu Zhou, Jiahao Shao, Lu Xu, Dongfeng Bai, Weichao Qiu, Bingbing Liu, Yue Wang, Andreas Geiger, and Yiyi Liao. Hugs: Holistic urban 3d scene understanding via gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21336–21345, 2024. 2

[43] Shijie Zhou, Haoran Chang, Sicheng Jiang, Zhiwen Fan, Zehao Zhu, Dejia Xu, Pradyumna Chari, Suya You, Zhangyang Wang, and Achuta Kadambi. Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. *arXiv preprint arXiv:2312.03203*, 2023. 2

[44] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017. 7

[45] Xingxing Zuo, Pouya Samangouei, Yunwen Zhou, Yan Di, and Mingyang Li. Fmgs: Foundation model embedded 3d gaussian splatting for holistic 3d scene understanding. *International Journal of Computer Vision*, pages 1–17, 2024. 2