International Journal of Computer Vision
https://doi.org/10.1007/s11263-024-02183-8

FMGS: Foundation Model Embedded 3D Gaussian Splatting for Holistic 3D Scene Understanding

Xingxing Zuo[1] · **Pouya Samangouei**[1] · **Yunwen Zhou**[1] · **Yan Di**[1] · **Mingyang Li**[1]

Received: 16 December 2023 / Accepted: 4 July 2024

Abstract

Precisely perceiving the geometric and semantic properties of real-world 3D objects is crucial for the continued evolution of augmented reality and robotic applications. To this end, we present Foundation Model Embedded Gaussian Splatting (FMGS), which incorporates vision-language embeddings of foundation models into 3D Gaussian Splatting (GS). The key contribution of this work is an efficient method to reconstruct and represent 3D vision-language models. This is achieved by distilling feature maps generated from image-based foundation models into those rendered from our 3D model. To ensure high-quality rendering and fast training, we introduce a novel scene representation by integrating strengths from both GS and multi-resolution hash encodings (MHE). Our effective training procedure also introduces a pixel alignment loss that makes the rendered feature distance of same semantic entities close, following the pixel-level semantic boundaries. Our results demonstrate remarkable multi-view semantic consistency, facilitating diverse downstream tasks, beating state-of-the-art methods by 10.2object detection, despite that we are 851× faster for inference. This research explores the intersection of vision, language, and 3D scene representation, paving the way for enhanced scene understanding in uncontrolled real-world environments. We plan to release the code on the [project page].

Keywords Gaussian splatting · Vision-language embeddings · Foundation models · Open-vocabulary semantics

1 Introduction

3D scene understanding is a critical task in various computer vision and robotics applications. Yet, most existing methods primarily concentrate on either 3D geometry and appearance estimation (Schonberger & Frahm, 2016; Mildenhall et al., 2020; Kerbl et al., 2023) or 3D object detection and scene segmentation trained on datasets with closed sets of classes (Dai et al., 2017; Grinvald et al., 2019; Narita et al., 2019). However, for an intelligent agent to interact smoothly with the physical world, merely understanding a subset of the space characterized by pre-identified labels is insufficient. Inspired by the latest advancements in foundation models (FMs) with impressive language and vision semantics (Radford et al., 2021; Alayrac et al., 2022), this paper endeavors to develop a more natural 3D scene representation supporting open-world visual recognition and understanding. It integrates both geometric and open-vocabulary semantic information, facilitating easy querying for downstream tasks such as object detection and semantic segmentation in open-world scenarios..

In this paper, we utilize Gaussian Splatting (Kerbl et al., 2023) as the backbone for reconstructing 3D geometry and appearance, which has demonstrated superior performance in terms of rendering quality for novel-view image synthesis and training efficiency. To assist open-vocabulary 3D scene understanding, we rely on pre-train 2D vision-language CLIP (Radford et al., 2021) and lift the corresponding information into 3D by a novel multi-view training procedure. We note that, in research communities, the system that is

Communicated by Hong Liu.

✉ Xingxing Zuo
zuox@caltech.edu

✉ Mingyang Li
mingyangli009@gmail.com

Pouya Samangouei
samangouei@google.com

Yunwen Zhou
verse@google.com

Yan Di
yanditum@google.com

[1] Google, Mountain View, USA

Published online: 12 August 2024

most similar to us is LEFR (Kerr et al., 2023), which integrates implicit NERF (Mildenhall et al., 2020) based scene representation and CLIP embeddings. Compared to LERF, our system develops a different architecture, and offers a range of technical advancements, including high efficiency and superior representation of open-vocabulary semantics, which results in significantly better performance (approximately 10.2key metrics).

A straightforward approach to enhance 3D Gaussian Splatting with vision-language FM embeddings is to attach each Gaussian with a learnable feature vector, which can be trained through image rasterization to formulate loss functions. However, maintaining high-quality rendering with GS typically requires millions of Gaussians in a nominal room-scale environment. Employing per-Gaussian feature vectors inevitably results in excessive memory consumption and significantly slows down training, limiting the practical applications of this system. Motivated by iNGP (Müller et al., 2022), we model our system by using 3D Gaussians together with multi-resolution hash encoding (MHE) to distill the foundation model embeddings. Specifically, to obtain the language embedding from the Gaussians, we utilize their mean values to query the MHE field at corresponding positions. Subsequently, this queried MHE is processed through a Multi-Layer Perceptron (MLP) to generate the output language embedding.

In the training phase, we employ a supervision mechanism on the MHE-based language FM CLIP feature field using a hybrid feature map. This map is derived from the average of multi-scale image crops obtained from various viewpoints. This approach enables the embedding to effectively capture language features corresponding to each scale ensuring a comprehensive representation. For instance, the embedding might represent a 'red book' when viewed up close, while depicting a 'library' from a more distant perspective. It is noteworthy that CLIP embeddings are designed to encapsulate the overall concept presented in a 2D image, exhibiting minimal variation across individual pixels. Additionally, CLIP embeddings are not perfectly multi-view consistent, meaning that when a 3D object is observed by a moving camera from different views, there are differences in the computed CLIP embeddings across frames. To solve the above-mentioned problems, we rely on a multi-view consistency training process to ensure that 3D models, when rendered from different image views, exhibit minimal variations. Additionally, to allow pixel-aligned query experience, DINO (Caron et al., 2021) embeddings are used together with CLIP embeddings similar to LERF (Kerr et al., 2023). By carefully analyzing the properties in both CLIP and DINO embeddings, we design an additional pixel alignment loss to further improve the object localization and scene understanding capabilities. This loss is grounded in the dot product similarity of CLIP/DINO features between the central pixel and its surroundings, guiding the rendered CLIP feature map to replicate the same similarity pattern observed in the DINO feature map.

This research paves the way for enhanced real-world applications, such as augmented reality experiences where users can interact with objects using natural language and robotic systems that can navigate and manipulate environments based on linguistic commands. By bridging the gap between language and 3D representation, FMGS opens up new possibilities for understanding and interacting with our surroundings.

The overview of this work is shown in Fig. 1. Our key contributions can be summarized as follows:

- Novel semantic scene representation: We introduce a novel approach combining 3D Gaussians (parameterized by mean, covariance, opacity, and spherical harmonics) for geometry and appearance representation, with MHE for efficient semantic embedding. This approach addresses memory constraints in room-scale scenes including millions of 3D Gaussians.
- Multi-view consistent language embeddings: Our training process utilizes Gaussian-splatting based rendering from multiple views, ensuring consistency across 3D space in static scenarios. Language embeddings remain invariant to viewpoints, enforcing local proximity consistency within Gaussian volumes.
- Addressing pixel misalignment: We address pixel alignment challenges of CLIP features by extracting and aggregating them at multiple resolutions for a hybrid CLIP feature, which is used for supervising the training. Regularization with pixel-aligned DINO features and a novel dot-product similarity loss enhances spatial precision and object differentiation.
- State-of-the-art performance: Our methods demonstrate superior performance in open-vocabulary semantic object localization, outperforming existing state-of-the-art approaches with quantitative and qualitative results by a wide margin, despite being hundreds of times faster.

2 Related Works

We review three main areas of related articles: 3D scene representation, open-vocabulary object recognition and scene understanding, and combined 3D scene representation and semantic understanding.

3D Scene Representation Scene representation in 3D can be roughly categorized by mesh-based, voxel-based, point-based, and implicit ones. Voxel-based methods typically discretize 3D space into regular grid cell elements where each grid cell corresponds to a voxel. To estimate the dense 3d voxel cells, probabilistic fusion methods were firstly (Izadi

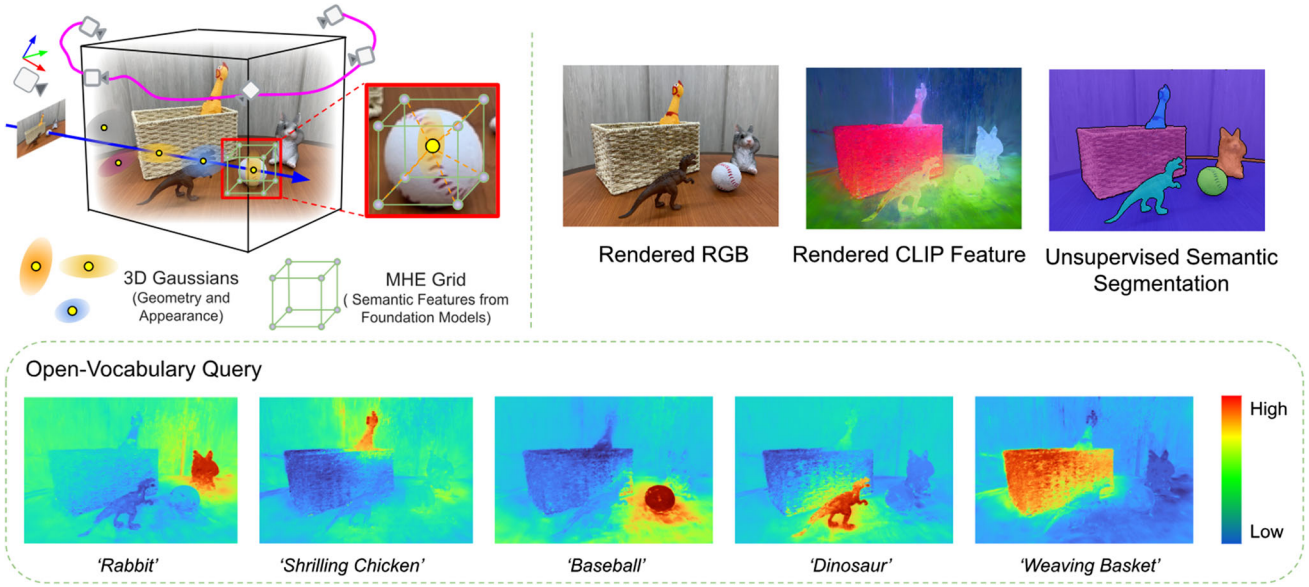

Fig. 1 Overview of our proposed FMGS (Foundation Model Embedded Gaussian Splatting). The semantic scene representation which consists of 3D Gaussians and MHE is firstly trained using the proposed method (top-left sub-figure), and subsequently used for rendering the RGB images and pixel-aligned semantic feature maps from foundational models (top-right sub-figure). These semantic features enable applications such as open-vocabulary queries for object detection and unsupervised semantic segmentation

et al., 2011) used and researchers also developed end-to-end learn-able methods (Sun et al., 2021), by using either depth sensors (Izadi et al., 2011) or monocular camera systems (Yang et al., 2020). To visualize estimated voxel fields, they are typically converted into a mesh-based representation. This enables efficient rendering on modern computer graphics systems. While alternative methods, such as those using 3D meshes (Schöps et al., 2019; Lin et al., 2021), have achieved notable success in various fields, their discrete scene representation, whether voxel-based or mesh-based, imposes limitations on the ability to achieve photo-realistic reconstruction and rendering performance.

Neural implicit representation, e.g., NeRF series (Mildenhall et al., 2020; Barron et al., 2021, 2022, 2023), represent 3D scenes by fully-connected neural networks, in which volume density and radiance can be queried by input position and view direction vectors. To improve the training and rendering efficiency of NeRFs, 3D space can be discretized by using MHE similar to the concept used in voxel-based methods (Müller et al., 2022). TensoRF (Chen et al., 2022) models radiance fields as 4D tensors, factorizing them into compact low-rank tensor components using CP decomposition and introducing novel vector–matrix (VM) decomposition for improved rendering quality, reduced memory footprint, and faster reconstruction.

Finally, point-based methods are originally widely used for directly processing data from depth sensors, for performing geometrical and semantic computer vision tasks (Qi et al., 2017; Karkus et al., 2021). Point-NeRF (Xu et al., 2022) efficiently combines point cloud and NeRF to achieve impressive fast view synthesis results. Recently, 3D Gaussian Splatting (GS) has been proposed to model points as 3D Gaussians for scene representation (Kerbl et al., 2023), and achieved state-of-the-art novel view synthesis rendering quality. However, in Kerbl et al. (2023), the number of Gaussians used for scene representation can easily surpass one million, which introduces strict memory and computational requirements for downstream use cases.

Open-Vocabulary Object Detection and Scene Understanding Advancements in open-vocabulary object detection in 2D images have been made by leveraging natural language prompts. LSeg (Li et al., 2022) employs a text encoder for semantic label embeddings and a transformer-based image encoder for dense pixel embeddings, using contrastive alignment to achieve zero-shot image segmentation and generalization to unseen categories. CRIS (Wang et al., 2022) leverages CLIP for image segmentation, employing a vision-language decoder to align text and pixel-level features, and text-to-pixel contrastive learning to enforce similarity between text and relevant pixel features. CLIP-Seg (Lüddecke & Ecker, 2022) leverages CLIP as a backbone, employs a transformer-based decoder for dense prediction, and generates image segmentation based on arbitrary text or image prompts. OV-Seg (Liang et al., 2022) improves open-vocabulary semantic segmentation by finetuning CLIP on masked image regions and text descriptions from noisy captions, achieving promising performance without dataset adaptations.

Current approaches often employ region proposal or mask prediction methods to guide open-vocabulary classification models. OpenSeg (Ghiasi et al., 2021) employs mask representations to facilitate visual grouping and align captions with predicted segmentation masks for open-vocabulary image segmentation. ViLD (Gu et al., 2021) advances open-vocabulary object detection by distilling knowledge from a pretrained image classification model (teacher) into a two-stage detector (student), aligning region embeddings of detected boxes with text and image embeddings inferred by the teacher. Detic (Zhou et al., 2022) expands object detectors' vocabulary by training their classifiers on image classification data, outperforming prior methods on open-vocabulary and long-tail detection benchmarks, achieving generalization to new datasets without finetuning and enabling detectors trained on all ImageNet classes. OVIR-3D (Lu et al., 2023) enables open-vocabulary 3D object instance retrieval by fusing text-aligned 2D region proposals into 3D space, leveraging 2D datasets.

Open-vocabulary scene understanding has also been explored by using point clouds as sensor inputs. PointCLIP (Zhang et al., 2021) aligns CLIP-encoded point cloud with 3D category texts, transferring knowledge from 2D to 3D recognition by projecting point cloud into multi-view depth maps, using an inter-view adapter for global feature extraction and few-shot knowledge fusion. ULIP series (Xue et al., 2022, 2023) learn a unified representation for images, texts, and 3D point cloud by leveraging pre-trained vision-language models and automatically synthesized triplets, improving the performance of various 3D backbones. Lu et al. (2023) leverage pre-trained image and vision-language models and cross-modal contrastive learning for open-vocabulary 3D point cloud detection without 3D annotations.

Combined 3D Scene Representation and Semantic Understanding Language has been incorporated into 3D scene understanding in various ways. For the task of visual question answering, systems like iQA (Gordon et al., 2018), ScanQA (Azuma et al., 2022), and SimVQA (Cascante-Bonilla et al., 2022) leverage 3D information to answer queries about the environment. For object recognition enhancement, language and shape information can be combined to improve object recognition, as seen in Corona et al. (2022) and Thomason et al. (2022).

Inspired by the success of implicit neural reconstruction (Mildenhall et al., 2020; Barron et al., 2021, 2022), researchers also start to explore incorporating language guidance into 3d neural scene representation. LERF (Kerr et al., 2023) enables open-ended language queries in 3D by incorporating language embeddings from models, e.g. CLIP, into NeRF. 3D-OVS (Liu et al., 2023) leverages pre-trained CLIP and DINO models in a weakly supervised manner, distilling multi-modal knowledge and object reasoning into a neural radiance field (NeRF) for segmentation tasks.

Tschernezki et al. (2022) leverage a pre-trained 2D image feature extractor to train a 3D student network, boosting performance in analyzing multiple images forming a 3D scene. FFD (Kobayashi et al., 2022) tackles scene editing by distilling knowledge from pre-trained 2D image feature extractors into a 3D feature field that guides local editing based on user queries. VL-Fields (Tsagkas et al., 2023), a neural implicit spatial representation fusing scene geometry and vision-language features, enables open-vocabulary semantic queries without requiring prior object class knowledge. FeatureNeRF (Ye et al., 2023) distills pre-trained vision models (DINO, Latent Diffusion) to learn generalizable NeRFs, leveraging neural rendering for 2D-to-3D mapping and extracting deep features from NeRF MLPs.

Additionally, ConceptFusion (Jatavallabhula et al., 2023) enables open-set and multimodal reasoning in 3D scene representations by fusing foundation model features with SLAM and multi-view fusion. ConceptGraphs (Gu et al., 2023) leverages 2D foundation models and multi-view association to capture semantic and spatial relationships for efficient task-driven planning. OpenMask3D (Takmaz et al., 2023) aggregates per-mask features using the multi-view fusion of CLIP-based image embeddings guided by predicted class-agnostic 3D instance masks. SA3D (Cen et al., 2023) enables 3D segmentation of target objects in neural radiance fields (NeRF) through one-shot manual prompting, leveraging density-guided inverse rendering, cross-view self-prompting, and an iterative process to project 2D segmentation masks onto 3D mask grids. PVLFF (Chen et al., 2023) generates a scene's feature field, combining vision-language and hierarchical instance features through contrastive loss from 2D instance segment proposals.

CLIP-Fields (Shafiullah et al., 2022) learns a spatial mapping to semantic embeddings via weak supervision from web-trained language and vision models, enabling tasks like object identification and robot navigation without direct human labeling. GNFactor (Ze et al., 2023), a multi-task robotic manipulation agent, leverages a shared 3D voxel representation and language-augmented neural fields for generalizable visual behavior cloning.

Our work is close and directly comparable to LERF (Kerr et al., 2023) in terms of assumptions about information available at the training phase and query time. For example, it does not assume a priori knowledge of query categories at training time which is assumed 3D-OVS (Liu et al., 2023).

Recently, several concurrent works have emerged, addressing similar problems, and the survey papers (Chen & Wang, 2024; Fei et al., 2024) also offer comprehensive overviews of related fields. LEGaussians (Shi et al., 2024) introduces a method for quantizing high-dimensional concatenated CLIP and DINO features into compact ones to conserve memory, and attach to each individual Gaussian. Langsplat (Qin et al., 2024) segments images by SAM (Kirillov et al.,

2023) and then inputs hierarchical semantic segments into CLIP model to extract semantic features for the segments. To address memory constraints, Langsplat incorporates a scene-specific language autoencoder to encode CLIP features into lower dimensions, which are also attached to each individual Gaussian. Feature 3DGS (Zhou et al., 2024) distills pixel-aligned features from 2D foundation models, including SAM (Kirillov et al., 2023) and LSeg (Li et al., 2022), into GS by associating each Gaussian with a learnable vector. However, pixel-aligned LSeg suffers from a loss of semantic understanding capability, particularly for long-tail semantics (Kerr et al., 2023). Different from all the aforementioned methods, which attach semantic features to each Gaussian, we propose to seamlessly integrate 3D Gaussian scene representation together with MHE for efficient semantic encodings. Notably, our method does not necessitate additional scene-specific quantization or auto-decoder steps, thereby preserving the semantic features' representation capability from foundation models.

3 Background Methods

3.1 3D Gaussian Splatting

GS (Kerbl et al., 2023) represents an environment using a set of 3D Gaussians, each defined by a mean $\mu \in \mathbb{R}^3$, an anisotropic covariance matrix $\Sigma \in \mathbb{R}^{3\times3}$, an alpha value $\alpha \in [0, 1]$ representing opacity, and spherical harmonics coefficients (SH). Given a 3D position $\mathbf{x} \in \mathbb{R}^3$, the probability density function of 3D Gaussian is defined as:

$$G(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1}(\mathbf{x}-\mu)} \tag{1}$$

where $(\cdot)^T$ represents a transpose operation and $(\cdot)^{-1}$ denotes matrix inversion. To render 3D Gaussians in 2D, we project their mean positions by point projection, and project their covariance using the following equation:

$$\Sigma' = \mathbf{JW}\,\Sigma\,\mathbf{W}^T\mathbf{J}^T \tag{2}$$

where $\mathbf{W} \in \mathbb{R}^{3\times3}$ is the viewing transformation and $\mathbf{J} \in \mathbb{R}^{3\times3}$ is the Jacobian of the affine approximation of the projective transformation (Zwicker et al., 2001). To optimize covariance matrices, we use an equivalent representation:

$$\Sigma = \mathbf{RSS}^T\mathbf{R}^T \tag{3}$$

where $\mathbf{R} \in \mathbb{R}^{3\times3}$ and $\mathbf{S} \in \mathbb{R}^{3\times3}$ are rotation and scaling matrices, respectively. GS also includes spherical harmonics coefficients to model the appearance of the scene. Gradients for all parameters are derived explicitly to avoid overhead during training.

Each Gaussian encodes the color c using spherical harmonics, which gives a value depending on the viewing directions. The $\alpha-$blending point-based rendering for a pixel color $\mathbf{c}$ is done by blending $\mathcal{N}$ points in the depth order from front to back:

$$\mathbf{c} = \sum_{i\in\mathcal{N}} \mathbf{c}_i\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{4}$$

where α_i is given by a 2D Gaussian multiplied by a learned per Gaussian opacity (Yifan et al., 2019).

Note that although the image rendering model is similar across NeRFs and GS, the rendering algorithm is much more efficient in GS. NeRFs need to march along the ray to integrate volume, however, GS rendering uses a point-based $\alpha-$blending approach. This allows GS to include a real-time rendering solution that leverages GPU sorting algorithms and draws inspiration from tile-based rasterization. By using a 3D Gaussian representation, anisotropic splatting can be performed while respecting visibility order. This is achieved through sorting and alpha-blending. Additionally, a fast and accurate backward pass is enabled by tracking the traversal of sorted splats.

3.2 Multi-resolution Hash Encoding

Representing a 3D feature field can have many forms. A naive method is to attach a feature vector (or multiple) to each Gaussian, which can be optimized along with other Gaussian parameters (position, covariance, and so on). However, this is extremely costly in terms of computational cost and memory consumption especially when a large number of Gaussians are generated for scene representation. In fact, adding a 512×1 feature vector per Gaussian will increase the number of optimized parameters to be $9.83\times$ under authentic GS parameterization (Kerbl et al., 2023) (10 geometric parameters and 48 spherical harmonic appearance parameters per Gaussian) and $65.0\times$ under simplified GS parameterization (Keetha et al., 2023) (4 geometric parameters and 4 appearance parameters per Gaussian).

To mitigate this problem, we are motivated by multi-resolution hash embedding (MHE) (Müller et al., 2022), which provides efficient scene representation that consists of two trainable components. The first component first hashes a given position $\mathbf{x} \in \mathbb{R}^3$, and then looks up into a trainable hash table for the corresponding embedding. The second component is an MLP that takes the corresponding embeddings and makes predictions such as color and density. The representation contains multiple hash tables, one per each scale. Specifically, MHE first encodes a given position $\mathbf{q} = MHE_\theta(\mathbf{x})$. To do so, it contains a hash table with L levels. Each level contains up to E feature vectors with

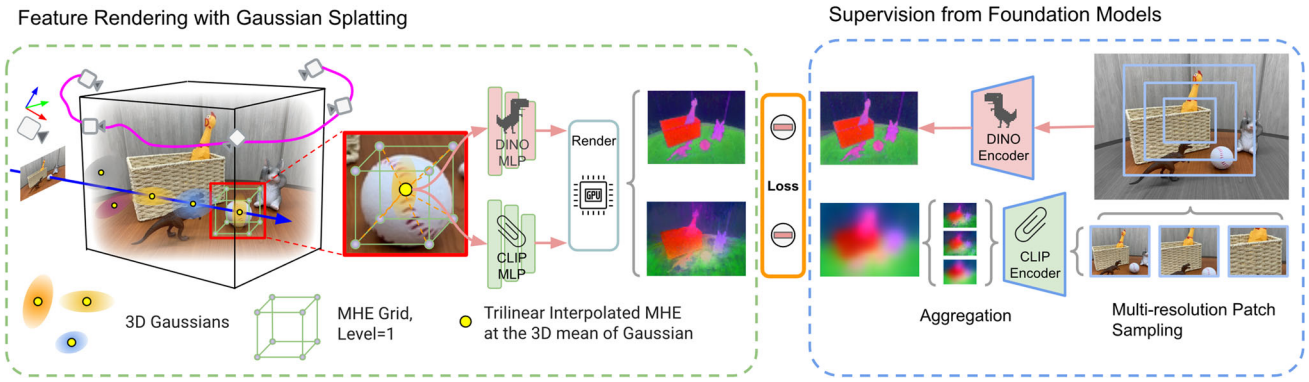

Fig. 2 FMGS Training pipeline: Left: Shows how FMGS' feature field renders CLIP and DINO feature maps for loss calculation. The feature field is a multi-resolution hash encoder (MHE) (Müller et al., 2022) that embeds semantic information into 3D Gaussians acquired from 3D Gaussian Splatting (Kerbl et al., 2023). Right: Shows the target DINO feature map and hybrid CLIP feature map from the foundation models. Note, for visualization simplicity, we only show a single-level MHE here but in implementation we have used multiple levels and concatenate their encodings

dimensionality D. Resolution of each level is determined by $N_l = \lfloor N_{\min} \cdot b^l \rfloor$ where $N_{\min}$ is the coarsest resolution, $N_{\max}$ is the finest resolution, and b is a growth factor.

To get $\mathbf{q}$ for a given position $\mathbf{x}$, we query MHE at all scales and concatenate the resulting features. For each scale, we find the enclosing voxel for $\mathbf{x}$. Then, each corner entry of the voxel is mapped into a feature vector with dimensionality D according to the trainable hash table. MHE trilinearly interpolates the queried corner entries according to their relative position of $\mathbf{x}$ in its hypercube for each level. This ensures the continuity of the encoded input and its composition with the neural network, avoiding grid-aligned discontinuities and blocky appearance. After this mapping is done, the features from all scales are concatenated to each other, and the auxiliary inputs $\psi \in \mathbb{R}^K$ which results in a feature vector $\mathbf{q}$ of size $L \times D + K$. The resulting encoding then goes to the second component which is an MLP network, $MLP_\Phi(\mathbf{q})$, produces the final output. This architecture significantly reduces the number of weights that are trained for each view while having an $O(1)$ GPU look up for hashing. Overall this results in significant improvements in quality and speed of training.

4 Method

Our method, i.e. Foundation Model Embedded Gaussian Splatting (FMGS), leverages strengths of both GS and MHE. We rely on GS for efficient and accurate scene geometry representation and on MHE for representing the scene's language content in a light-weighted manner. Given a set of input images, we compute the corresponding camera poses and 3D sparse visual points using an off-the-shelf structure from motion system, e.g., COLMAP (Schonberger & Frahm, 2016). After that we train GS and acquire 3D Gaussians.

Subsequently, we train the feature embedding field (MHE) in 3D by grounding 2D CLIP embeddings. This requires us to generate pixel-aligned features on a set of calibrated input images. However, CLIP embeddings are global in nature and not suitable for pixel-aligned feature extraction. To overcome this challenge, we introduce a framework to learn a volumetric language embedding field that embeds over the 3D Gaussians. The field effectively generate features that is the average CLIP features across all views that include that 3D Gaussian. To supervise our dense feature field, we create a hybrid feature map based on CLIP embeddings across multi-scale crops of training views. Figure 2 provides an overview of our training pipeline.

4.1 Feature Field Architecture

3D Gaussian Splatting produces millions of Gaussians to enable high quality rendering of a room-scale scene. This makes it very inefficient to have one CLIP feature per Gaussian since these features are high dimensional and keeping all of these features in GPU memory is not feasible.

To this end, we parameterize our feature field efficiently using MHE. For a given 3D Gaussian $G(\mathbf{x})$ with mean position $\mathbf{x}$, we first encode $\mathbf{x}$ to a feature vector $\mathbf{q} = MHE_\theta(\mathbf{x})$ where θ is our multi-resolution hash table parameters. We subsequently feed this output into an MLP, which generates our language embedding $\hat{\mathbf{f}} = MLP_\phi^{CLIP}(\mathbf{q})$, with $\hat{\mathbf{f}}$ belonging to $\mathbb{R}^D$. We also normalize $\hat{\mathbf{f}}$ to make it a unit vector.

4.2 Embed the Foundation Models

We embed the semantic embeddings from foundation models to our scene representation. Training the semantic embedding has three aspects. First, we use our scene representation to

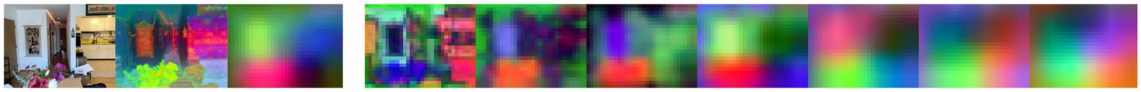

Fig. 3 The features extracted from foundation models. The left three subfigures include the RGB image, extracted DINO features from the foundation model, and the hybrid CLIP feature, which is an average of multi-scale CLIP feature maps shown on the right. On the right, the shown seven CLIP feature maps are the extracted from an image pyramid at multiple scales using the foundation model. The resolution of CLIP features decreases from left to right

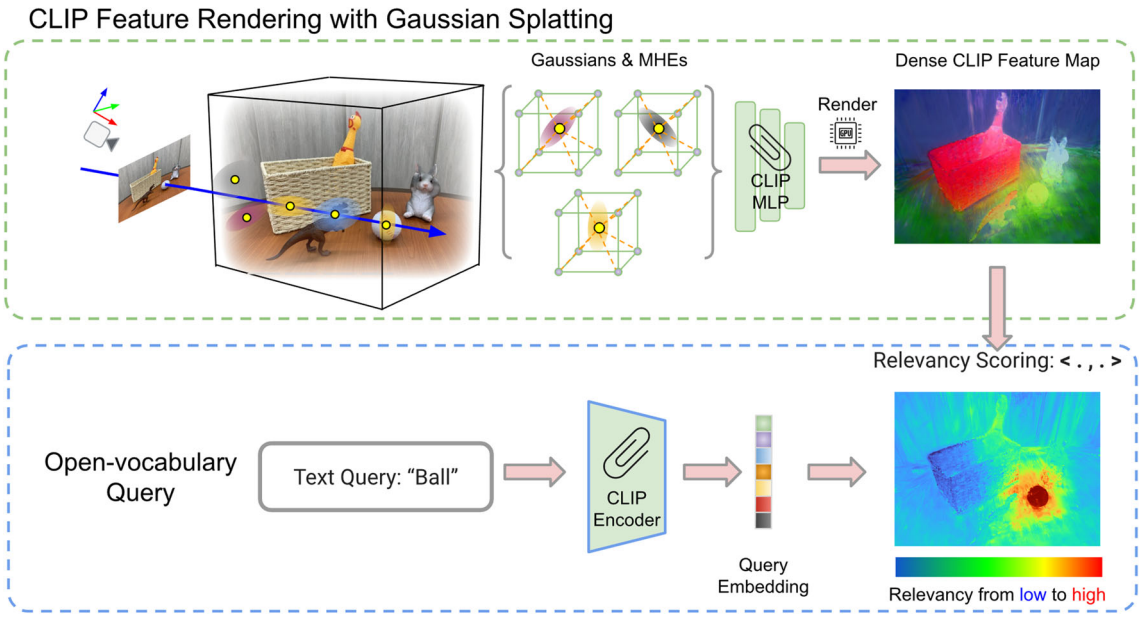

Fig. 4 FMGS Query pipeline: Top: Given a query view to localize a query, FMGS first renders the dense CLIP feature map. Bottom: given an open-vocabulary query, FMGS generates a relevancy map highlighting the relevant part of the rendered CLIP feature map to the query embedding. The highest relevant is colored as red while the lowest relevant part is colored as blue. Note, for visualization simplicity, we show a single-level MHE in this figure while using multiple-level MHEs in our implementations (Color figure online)

render a predicted feature map $\hat{\mathbf{F}} \in \mathbb{R}^{W\times H\times D}$ where W is the width, H is the height, and D is the dimension of the feature map. Second, we generate target feature maps $\mathbf{F}$ by feeding the image view to foundation models. Finally, we need to ensure that the predicted feature map is aligned with the corresponding target features and follows the same object boundaries in terms of feature similarity.

Hybrid CLIP Feature for Supervision To supervise our feature field outputs, given a calibrated input image, we first rasterize the features into a 2D feature map $\hat{\mathbf{F}}$ where the (i, j)th feature is acquired by point-based $\alpha-$blending:

$$\hat{\mathbf{f}}_{i,j} = \sum_{k\in\mathcal{N}} \hat{\mathbf{f}}_k \alpha_k \prod_{l=1}^{i-1}(1-\alpha_l) \tag{5}$$

To generate our target CLIP feature map, denoted as $\mathbf{F}$, we initially pre-compute a multi-scale feature pyramid of CLIP embeddings, similar to the approach used in LERF (Kerr et al., 2023). This involves feeding image patches at various sizes into the CLIP foundation model. However, in contrast to LERF, which trains its scene representation by interpolating embeddings from the pre-computed CLIP feature pyramid at random scales, we rely on a single hybrid CLIP feature map for training our scene representation. We scale up the embeddings of the smaller scales in the pre-computed CLIP feature pyramid bilinearly to the largest scale feature map, and generate the hybrid feature map by averaging them. We define our CLIP loss by the following Huber loss:

$$\mathcal{L}_{CLIP} = \begin{cases} 0.5|\hat{\mathbf{F}} - \mathbf{F}|^2, & \text{if } |\hat{\mathbf{F}} - \mathbf{F}| < \delta \\ \delta \cdot (|\hat{\mathbf{F}} - \mathbf{F}| - 0.5 \cdot \delta), & \text{otherwise} \end{cases} \tag{6}$$

where δ is a hyperparameter, which is set to be 1.25 empirically. As seen in Fig. 3 where we use PCA to visualize feature maps following FFD (Kobayashi et al., 2022), we notice that the target CLIP feature map is not fine-grained enough when embedding similarities of neighboring pixels are considered. This results in poor pixel-alignment gradient signals on Gaus-

sians that are not relevant semantically. On the other hand, DINO (Caron et al., 2021) features give sharp boundaries between objects (Amir et al., 2021) in terms of embedding similarity, which can be used for additional regularization.

Regularization with DINO Feature To transfer the characteristics of DINO features while maintaining the CLIP embedding semantics, we (a) add a DINO feature field loss and (b) define a pixel-alignment loss between the DINO and CLIP feature fields. The DINO feature field shares the same hash grid parameters as CLIP and gives the same encoding $\mathbf{q}$ for a given $\mathbf{x}$. Then the DINO feature field outputs $\hat{\mathbf{d}} = MLP_{\psi}^{DINO}(\mathbf{q})$ where ψ denotes the parameters of the MLP that are not shared with MLP_{ϕ}^{CLIP}. This feature field is supervised by passing the $sampled\ image$ once to the pre-trained DINO model without scaling, yielding $\mathbf{D} \in \mathbb{R}^{W \times H \times L}$ where L is the DINO feature dimension. We then render $\hat{\mathbf{D}}$ using the same approach as rendering $\hat{\mathbf{F}}$. The DINO regularization loss is as follows:

$$\mathcal{L}_{DINO} = |\hat{\mathbf{D}} - \mathbf{D}|^2 \tag{7}$$

Pixel-alignment with Dot Product Similarity We define a pixel-alignment loss by defining a kernel around every pixel and enforce the dot product similarity in normalized embedding spaces (between DINO and CLIP) are consistent across the center pixel and surrounding ones. We normalize both rendered features to the unit norm, and then compute the loss:

$$\mathcal{L}_{pixel} = \frac{1}{K^2 - 1} \sum_{i \in \mathcal{P}} \sum_{\substack{j \in \mathcal{N}(i), \\ j \neq i}} |\hat{\mathbf{d}}_i^T \hat{\mathbf{d}}_j - \hat{\mathbf{f}}_i^T \hat{\mathbf{f}}_j| \tag{8}$$

where $\mathcal{P}$ denotes the set of all the pixels in the image, and $\mathcal{N}(i)$ is the $K \times K$ patch kernel around the rendered feature at pixel i. This makes the rendered CLIP feature follow the same similarity pattern as the DINO feature. Note that we stop the gradient back-propagation through the rendered DINO features in this training loss, which means MLP_{ψ}^{DINO} would not be affected by this loss. $\mathcal{L}_{pixel}$ is also termed as "dotsim" loss for the rest of the paper, since it is formulated by dot product similarity.

Training Loss Overall our total loss is

$$\mathcal{L}_{total} = \lambda \mathcal{L}_{CLIP} + (1 - \lambda)\mathcal{L}_{DINO} + \gamma \mathcal{L}_{pixel} \tag{9}$$

where λ and γ are weights to balance different loss terms. We take the mean reduction over all the pixels in the image plane when computing the different loss terms. We also empirically find out that adding the pixel-alignment loss $\mathcal{L}_{pixel}$ with an appropriate weight γ has significant benefits, resulting in crisp and high-quality rendered CLIP feature maps and producing the best performance. However, an excessively large weight γ can exaggerate the differences in CLIP features and overly enrich them, which can be detrimental to object-level semantic understanding.

In Fig. 5, we provide examples of features extracted from foundation models for training and the rendered features generated by our trained hybrid semantic scene representation. It is evident that the rendered feature maps exhibit higher quality when compared to the raw feature maps obtained directly from the foundation models, owing to our training process enforcing multiple-view consistency.

4.3 Relevancy Score

At query time, when provided with a query prompt and a viewing direction, FMGS generates a relevancy map that assigns high scores to semantically relevant locations (see Fig. 4). To obtain this relevancy map, we first render the feature map $\hat{\mathbf{F}}$ using our learned semantic feature field via GS rasterization. Then, we calculate the CLIP embedding $\mathbf{f}_{query}$ corresponding to the query prompt.

To obtain the dense relevancy map, we define a set of canonical phrases with CLIP embeddings $\mathcal{F}_{can}$ following the methodology similar to Kerr et al. (2023). Then, we compute pairwise softmax scores based on the cosine similarity between the prompt embedding and $\hat{\mathbf{f}}_{i,j}$, representing the $\hat{\mathbf{F}}$ at location (i, j), as well as the canonical embeddings for canonical phrases. We take the minimum value of the softmax over all canonical prompts and deem it the relevancy score r:

$$r_{i,j} = \min_n \frac{\exp(\hat{\mathbf{f}}_{i,j}^T \mathbf{f}_{query})}{\exp(\hat{\mathbf{f}}_{i,j}^T \mathbf{f}_{query}) + \exp(\hat{\mathbf{f}}_{i,j}^T \mathbf{f}_{can}^n)}, \mathbf{f}_{can}^n \in \mathcal{F}_{can} \tag{10}$$

With the above definition, the relevancy score is higher when a query embedding is closer to the rendered feature than the canonical features. We follow Kerr et al. (2023) and choose the following canonical prompts: "object", "stuff", "things", and "texture". We also find that these work well for a wide range of queries removing the need for tuning these canonical terms. In Fig. 5, we present representative relevancy maps generated by matching the query embedding with our rendered CLIP feature map and the target CLIP feature map from the foundation model used in our training. It's evident that the relevancy map derived from our rendered CLIP feature map overall exhibits finer granularity and higher quality.

4.4 Implementation Details

Our approach employs a hash grid for representing language features, which is notably larger than a typical RGB hash grid. This hash grid comprises 24 layers, spanning resolutions from 16 to 512, and possesses a hash table size of 2^{20}

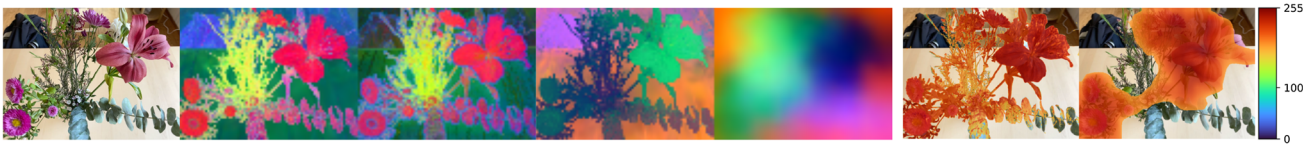

Fig. 5 Features for Training and Rendered Views. Left: From left to right, the figures show the RGB image, the rendered DINO feature map, the raw DINO feature map extracted for training, the rendered CLIP feature map, and the raw CLIP feature map used for training. Right: We display the relevancy scores for the rendered and raw CLIP feature maps with the text query 'flower', where the color bar indicates relevancy scores normalized within the 0–255 range. Notably, querying the raw CLIP feature map is much inferior to querying the rendered CLIP feature map (Color figure online)

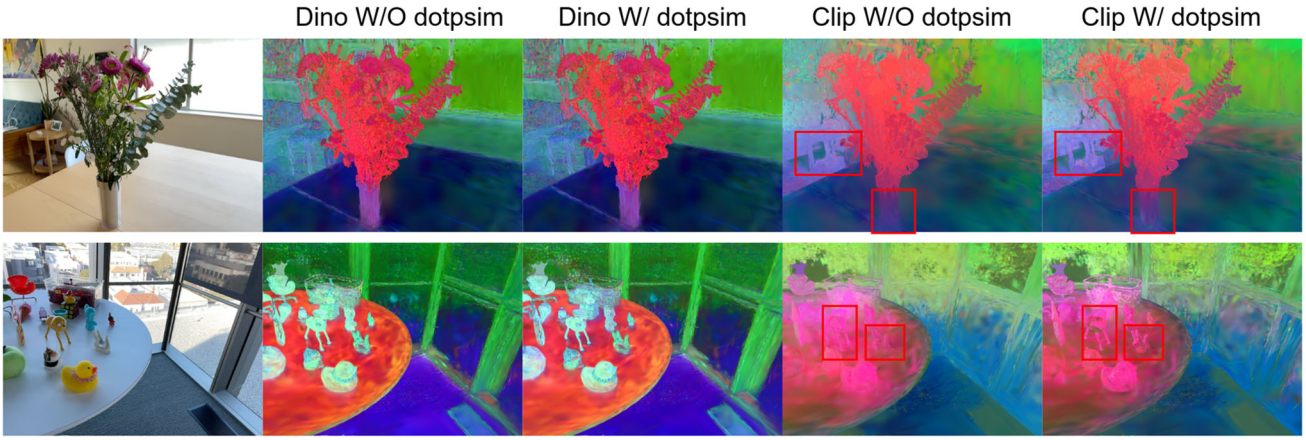

Fig. 6 Effect of dot product similarity (dotpsim) loss. From left to right: RGB image, rendered DINO feature without dotpsim, rendered DINO feature with dotpsim, rendered CLIP without dotpsim, and rendered CLIP feature map with dotpsim. The DINO feature maps do not have significant differences with or without dotpsim. From the CLIP feature maps, we can see that objects can be further distinguished from each other and the background. Differences are highlighted in the red boxes (Color figure online)

with an associated feature dimension of 8. The architecture of the CLIP and DINO MLP models used for MLP_{ϕ}^{CLIP} and MLP_{ψ}^{DINO} aligns with that of LERF (Kerbl et al., 2023). Furthermore, we leverage the OpenCLIP (Cherti et al., 2022) ViT-B/16 model, which has undergone training on the LAION-2B dataset. Notably, this model operates with an image pyramid that varies in scale from 0.05 to 0.5 of image size, encompassing a total of seven scales for pre-computing a CLIP feature pyramid. The pre-computed feature pyramid is subsequently processed by average pooling to generate the final hybrid CLIP feature for training our semantics embedded field.

Initially, we train the Vanilla Gaussian Splatting scene representation (Kerbl et al., 2023) through a total number of 30K iterations, with approximately 10 min total time for a room-scale scene. It's worth noting that representing such a scene requires the utilization of millions of Gaussians. Subsequently, we maintain the frozen states of the geometric attributes and spherical harmonics associated with these Gaussians throughout the subsequent training process for semantic embedding fields.

To mitigate GPU memory constraints, we strategically select approximately 40% of the Gaussians based on criteria such as high opacity values and a 2D radius of projected Gaussian exceeding 2 pixels in at least one training view. Only these selected Gaussians are involved in the rendering process when we train the semantic embeddings. For optimization, we employ the RAdam optimizer with a weight decay of 10^{-9}. We incorporate an exponential learning rate scheduler, which spans from an initial value of 5×10^{-3} and gradually decreases to 4×10^{-3} over the course of 4.2K training steps (after the initial 30K original GS training steps). In our training regimen, all models initially undergo 2.5K steps without the pixel alignment loss being enabled. These training and testing procedures are executed on an NVIDIA RTX A5000 GPU with 24GB of GPU RAM. The semantic feature field training time with a total of 4.2K steps takes about 1.4 hours. During training, we use weighting factors to balance the CLIP loss ($\lambda = 0.2$) and the pixel-alignment loss ($\gamma = 0.01$).

5 Experiments

Our hybrid semantic scene representation, FMGS, seamlessly integrates the 3D Gaussians and multi-resolution hashing encoding and supports both photo-realistic rendering and open-vocabulary object detection. In this section, we carefully evaluate the performance of open-vocabulary object detection (or localization) of our proposed method in uncontrolled real-world scenarios. To showcase the embedding quality of our method, we also evaluate it out-of-the-box on the open-vocabulary semantic segmentation task. We compare our method to other SOTA approaches for each experiment and show significant improvement over their results.

5.1 Object Detection in the Wild

By distilling the language embeddings extracted from off-the-shelf vision-language model, CLIP, our FMGS is applicable for associating a wide range of textual prompts with the relevant vision clues. We test the open-vocabulary object understanding capability of our method by object detection experiments.

Dataset: We use the same dataset as used in the LERF (Kerr et al., 2023) for object detection evaluation, for the purpose of fair comparison. It consists of five labelled scenes with 2D bounding boxes of objects associated with text prompts. There are objects including both common and long-tail ones with different sizes, and the queries for objects are quite diverse, like 'vase', 'eucalyptus', 'big white crinkly flower', 'pikachu', 'twizzlers', 'spoon handle', 'power outlet', 'waldo', 'stuffed bear', 'cookies on a plate', etc. The location of queried images are labelled by bounding boxes in the test images, which are rendered at novel views from trained NeRF models of individual scenes. The scenes in LERF dataset are collected by an iPhone, and each scene comprise $\sim$ 200 images. The provided poses of images from Ploycam app have significant noises in some scenes. Thus we regenerate the poses of images by running COLMAP (Schonberger & Frahm, 2016), which also yields sparse 3D visual points serving as input to initialize 3D Gaussians in our method. The poses of the officially-provided test images are also properly transferred to our COLMAP trajectory by Sim(3) alignment between officially-provided image poses and our COLMAP poses.

Evaluation Protocol: Following LERF (Kerbl et al., 2023), the evaluation metric for object detection is the accuracy rate. We redeem the query as a success if the highest relevancy pixel is located inside the target box. The relevancy score at each pixel is obtained by matching the rendered CLIP feature map with the language embedding of the given text query as described in Sec. 4.3.

Baselines: We compare against FFD-LSeg that embeds pixel-aligned LSeg feature (Li et al., 2022) into NeRF (NeuralStudio 'neurfacto' implementation by feature fields distillation method (Kobayashi et al., 2022), OWL-ViT (Minderer et al., 2022) that is a 2D method based on Vision Transformer encoder and fine-tuned for object detection, LERF (Kerr et al., 2023) that embeds CLIP and DINO features into NeRF. The 3D methods, FFD-LSeg and LERF, share the same evaluation protocol as our FMGS. For the 2D method, OWL-ViT, we regard it as a success if the center of the predicted bounding box is located in the target box.

Evaluation Results: The quantitative evaluation results on all sequences of LERF dataset are presented in Table 1, and representative relevancy score maps of the proposed method are shown in Fig. 7. The detailed results demonstrate significant advantages of FMGS's integration of language embeddings in detecting objects associated with long-tail prompts. While LSeg (Li et al., 2022), trained on a small dataset to learn pixel-aligned CLIP features, exhibits diminished open-vocabulary language understanding capabilities, the approach of FFD-LSeg, which distills LSeg features into radiance fields, struggles with comprehending long-tail queries and consequently exhibits poorer performance. In terms of open-vocabulary 2D detection, Owl-ViT, which utilizes full-HD NeRF views and selects bounding boxes based on the highest confidence scores for text queries, outperforms FFD-Lseg. However, when faced with long-tail queries, Owl-ViT's performance falls short in comparison to the robust and versatile FMGS.

We also conducted a comparison with the closest method, LERF, which distills DINO and CLIP features into neural radiance fields represented solely by MHEs. As depicted in Table 1, our FMGS outperforms LERF significantly, achieving an accuracy improvement of 10.2the officially released code, slightly surpass those reported in the original paper (Kerr et al., 2023).

In Fig. 8, we present side-by-side comparisons with LERF (Kerr et al., 2023). The object detection results are visualized, highlighting the superior quality of the relevance map produced by our FMGS. It notably focuses more on the queried target objects, as opposed to LERF. This outcome stems from our hybrid representation, which combines 3D Gaussians and MHEs for semantic scene representation. The 3D Gaussians represent both the geometry and appearance of the scene, naturally dividing 3D structures of objects and the scene into distinct Gaussian volumes. This partitioning feature aids in distinguishing objects from each other and from the background. In FMGS, we assign an identical MHE embedding to a Gaussian volume, further promoting semantic consistency in local proximity. This, in turn, contributes to focusing of relevance on the target object. Taking the query 'Pikachu' in Fig. 8 as an example, where 'Pikachu' is depicted on the side of a paper bag. Even when observing from a challenging

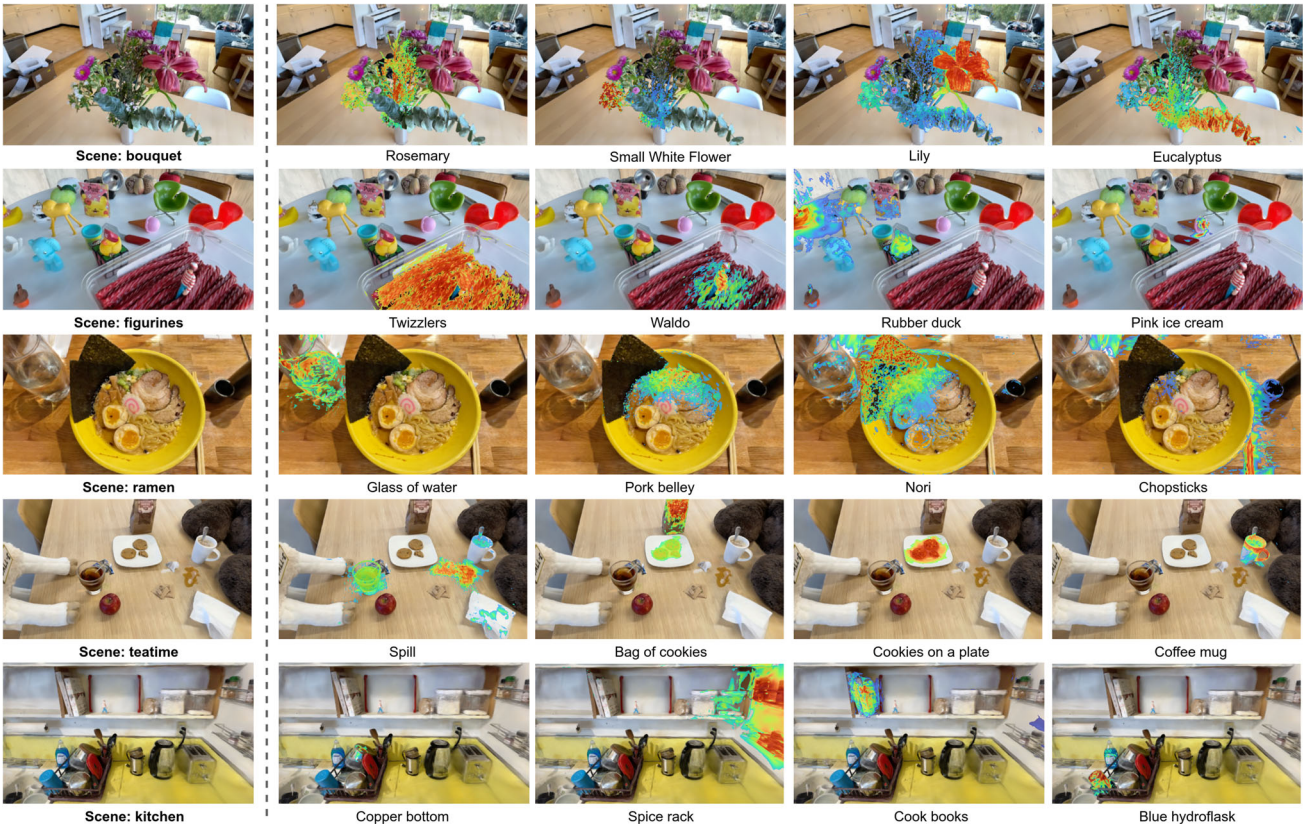

Fig. 7 Relevancy score for object detection. Left: The rendered RGB image at novel view from 5 scenes on LERF dataset (Kerr et al., 2023). Right: Visualization of relevancy scores with the given text queries shown below the figures. We overlay them on the RGB images

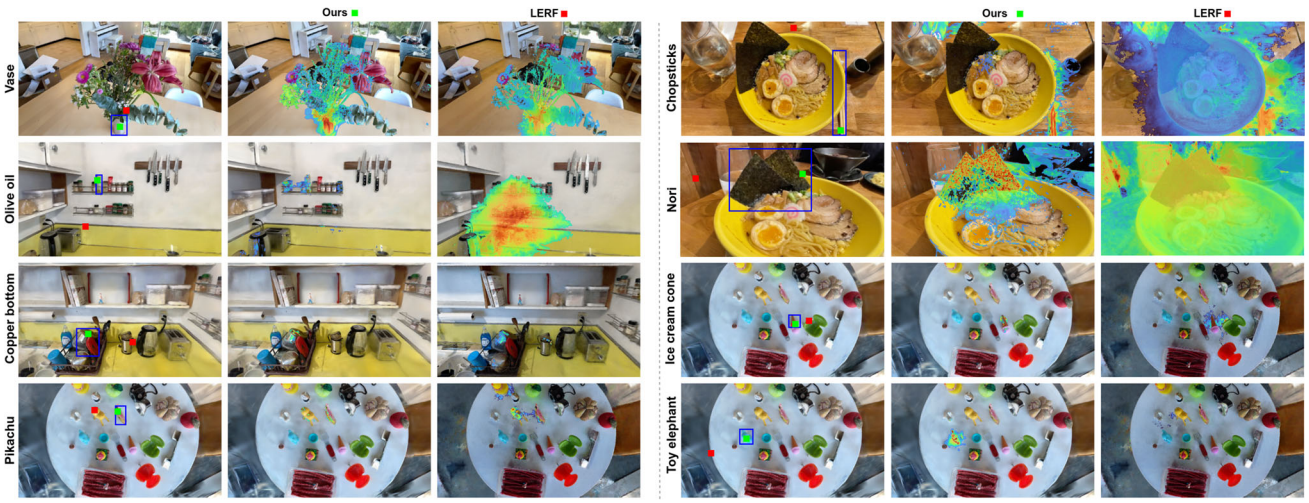

Fig. 8 Object detection results. The left and right groups of figures illustrate the detection results for four open-vocabulary queries, respectively. Each group comprises three subfigures: Left displays the ground-truth bounding boxes (blue), our detected highest-relevancy pixel (green) and the one detected by LERF (red) (Kerr et al., 2023). Middle showcases our relevancy score corresponding to the given text query. The text query is shown at the far left of each row. Right showcases LERF's relevancy score corresponding to the given text query. Our computed relevancy score is more focused on the target objects linked to the query (Color figure online)

Table 1 Accuracy and runtime efficiency (Frames Per Second, FPS) of object detection with open-vocabulary queries.

Scene	FFD-LSeg (Kobayashi et al., 2022)	OWL-ViT (Minderer et al., 2022)	LERF (Kerr et al., 2023)	Ours
Bouquet	50.0%	66.7%	*83.3%*	**100.0 %**
Figurines	8.9%	38.5%	*87.2%*	**89.7%**
Ramen	15.0%	**92.5%**	62.5%	*90.0 %*
Teatime	28.1%	75.0%	**96.9%**	*93.8%*
Kitchen	13.0%	42.6%	*85.2%*	**92.6 %**
Average Acc.	18.0%	54.8%	*83.0%*	**93.2%**
Inference FPS	–	–	0.1214	103.4

Comparison between Feature Fields Distillation (Kobayashi et al., 2022) using LSeg (Li et al., 2022) features (FFD-Lseg), OWL-ViT (Minderer et al., 2022), LERF (Kerr et al., 2023) and Ours FMGS. Please find more details on scenes and text queries for LERF dataset in Kerr et al. (2023)
Best marked with bold and second best marked with italics

Table 2 Segmentation evaluation

Methods	*Bed*		*Sofa*		*Lawn*		*Room*		*Bench*		*Table*	
	mIoU	mAP	mIoU	mAP	mIoU	mAP	mIoU	mAP	mIoU	mAP	mIoU	mAP
OV-Seg (Liang et al., 2022)	79.8	40.4	66.1	69.6	81.2	92.1	71.4	49.1	88.9	89.2	80.6	65.3
3D-OVS (Liu et al., 2023)	89.5	96.7	74.0	91.6	88.2	97.3	92.8	98.9	89.3	96.3	88.8	96.5
LERF (Kerr et al., 2023)	33.5	25.6	28.1	45.6	49.8	82.0	26.3	49.1	55.2	79.5	31.1	33.3
Ours	**38.0**	**50.1**	**56.6**	**82.0**	**64.9**	**90.5**	**57.0**	**85.3**	**62.1**	**84.1**	**63.6**	**85.3**
Ours-Refined	**80.6**	**85.5**	**90.8**	**97.4**	**92.6**	**98.5**	**87.9**	**97.8**	**84.5**	**94.8**	**89.4**	**97.2**

We report the mIoU(↑) scores and the mAP(↑) scores of the following methods in 6 scenes of 3D-OVS dataset (Liu et al., 2023). Note that 3D-OVS is a weakly supervised method, which knows the segmentation annotations in training and specially designed for segmentation task. Our method and LERF are 3D method training without any segmentation annotations, relying only on the relevancy between class query and the rendered CLIP features. OV-Seg (Liang et al., 2022) is a supervised method for segmentation task. Our method and LERF are unsupervised methods, under apple-to-apple comparison. We can further post-process and refine our 2D segmentation results by SAM (Kirillov et al., 2023) and get the results shown as 'Ours-Refined'
Best marked with bold

Table 3 Ablation study.

Methods	Bouquet	Figurines	Ramen	Teatime	Kitchen	Average
Ours	100.0	89.7	90.0	93.8	92.6	**93.2**
W/O dotpsim	100.0	91.0	85.0	90.6	85.2	90.4
W/O hybrid CLIP	54.2	32.1	52.5	6.3	9.3	30.8
W/ LERF CLIP	91.7	70.5	72.5	72.5	87.0	78.8
W/O MHE	91.7	71.8	90.0	90.6	77.8	84.4

Object detection comparison between our full method, ours without dot product similarity (dotpsim) loss, and ours without hybrid CLIP features by averaging at multiple scales for supervision, using single scale CLIP feature at the finest-resolution instead, as well as ours with LERF CLIP at multiple individual scales (Kerr et al., 2023)
Best marked with bold

viewpoint with almost no visibility of 'Pikachu', FMGS successfully maintains high relevance at the target location, due to its 3D consistency and fine-grained scene understanding. In contrast, LERF fails to detect 'Pikachu' and mistakenly identifies a visually similar object.

Inference Runtime: Our FMGS, relying on 3D Gaussian Splatting rendering (Kerbl et al., 2023), excels in efficiently rendering RGB images. We've implemented our rendering method for CLIP and DINO feature maps based on a CUDA implementation of Gaussian Splatting rendering. Even when rendering deep features with high dimensions, which can significantly increase computation time, our FMGS remains remarkably fast. It can render the 480 × 270 CLIP feature map, DINO feature map, and RGB image jointly at an impressively high rate of 103.4 FPS during inference, even with our unoptimized implementation. In contrast, LERF operates at a significantly slower pace, achieving a mere 0.1214 FPS during inference (see Table 1). This slowness stems from

LERF's need to perform a brute-force search for the best scales when rendering CLIP features, spanning a range from 0 to 2 ms with 30 increments. Consequently, we are **851.73 times faster** than LERF in rendering CLIP features, enabling efficient real-time open-vocabulary queries after our scene representation is trained.

5.2 Unsupervised Segmentation

In following experiments we use FMGS to segment queries and evaluate their segmentation masks. **Note** that our method is not delicately designed for the segmentation task. We lack a dedicated segmentation header for predicting segmentation masks, nor do we explicitly partition the scene at the object level. We have examined the open-vocabulary language understanding capability of FMGS in the object detection experiments discussed in the above section. Our primary objective for doing this segmentation evaluation is to assess the pixel-level accuracy of the rendered CLIP features obtained from the trained scene representation. Segmentation relies on matching these rendered CLIP features to the embeddings of the provided semantic labels.

Dataset: We conducted our segmentation evaluation using the 3D-OVS dataset (Liu et al., 2023), which consists of six scenes with labeled ground-truth semantic segmentation masks for test image views. These scenes are characterized by their cleanliness, with clear backgrounds and well-defined foreground objects. Each scene comprises approximately 30 images with predefined poses and sparse points computed using COLMAP (Schonberger & Frahm, 2016). The dataset includes a variety of objects, including many long-tail objects like 'Gundam,' 'Pikachu,' 'Stapler', and more. For further details about the scenes and semantic labels, please refer to Liu et al. (2023).

Evaluation Protocol: In terms of our evaluation protocol, we rely on the annotated ground-truth masks for the test views. These masks serve as a reliable benchmark for both qualitative and quantitative assessments of segmentation performance. We calculate the mean Intersection over Union (mIOU) scores and mean Average Precision (AP) metrics by comparing the segmentation results with these ground-truth masks.

Baselines: We conduct a direct comparison of our method with LERF (Kerr et al., 2023). To perform semantic segmentation, we initially obtain relevancy scores by computing the cosine similarity between the rendered CLIP feature and the embeddings of all class labels (this is different from the relevancy score calculation with auxiliary canonical phrases involved in Sec. 4.3.). These relevancy scores serve as segmentation logits, and we subsequently apply the softmax function to convert them into probabilities. Each pixel is then assigned a semantic class label corresponding to the maximum probability. Note that LERF (Kerr et al., 2023) requires a scale factor when rendering CLIP features, and we report the best segmentation results that can be achieved by LERF by selecting the best scales for each ray. It's also important to note that both LERF and our method encounter challenges in discerning the semantic labels of backgrounds when presented with visibility-limited close views and lack of context. Therefore, we have replaced the original background labels, including 'white sheet', 'wood wall', 'grey sofa', and 'lime wall', with a more general label 'background' when testing LERF and our method.

Additionally, for comprehensive reference, we present results obtained using the dedicated 3D-OVS method (Liu et al., 2023) for the segmentation task. However, it is worth emphasizing that comparing object detection methods like ours and LERF (Kerr et al., 2023) to 3D-OVS is not entirely equitable, as acknowledged in the paper of 3D-OVS (Liu et al., 2023). 3D-OVS (Liu et al., 2023) has prior access to segmentation class labels and distill class-related information into the radiance field during training. In contrast, neither LERF nor our methods have access to class labels during scene representation training. Consequently, the trained 3D-OVS scene representation can only be effectively employed for querying the classes known before training, and does not support arbitrary semantic queries beyond the trained classes. Furthermore, we compare to a 2D ceiling approach (Liang et al., 2022), OV-Seg, which is directly trained for open-vocabulary semantic segmentation by fine-tuning CLIP on masked image regions and text descriptions. OV-Seg is supervised with mask-category pairs, while ours and LERF are completely unsupervised.

Evaluation Results The segmentation experiment results are presented in Table 2 and Fig. 9. Notably, our approach outperforms LERF (Kerr et al., 2023) by a significant margin across all cases. This superior performance can be attributed to the higher quality of our rendered CLIP feature compared to the one produced by LERF. Our method exhibits more concentrated high relevancy around the queried objects, showcasing the advantage of our semantic scene representation, which maintains high semantic consistency in local proximity. We refine our 2D segmentation results using SAM (Kirillov et al., 2023) by assigning class labels to SAM segments through a majority voting based on pixel labels obtained from our method. The refined result is displayed in the 'Ours-Refined' row of Table 2 and Fig. 9.

5.3 Ablations

We conducted an ablation study on the object detection task, as it serves as a key indicator of our method's open-vocabulary semantic understanding capabilities. The results are presented in Table 3.

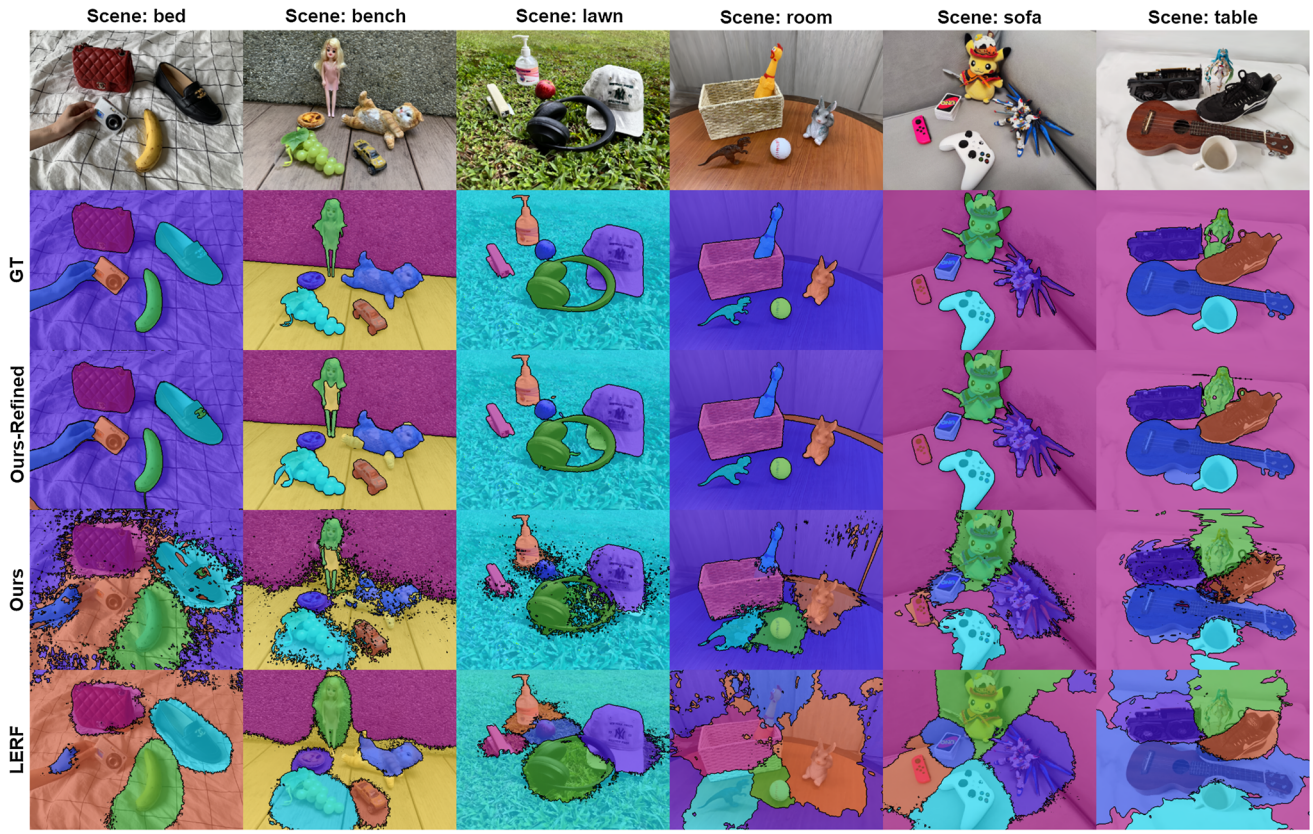

Fig. 9 Semantic segmentation results. In the rows from top to bottom, we display RGB images, ground-truth (GT) segmentation masks, our refined segmentation results, our segmentation results, and the segmentation results obtained by LERF (Kerr et al., 2023) scene representation. It's essential to note that neither our method nor LERF was initially intended for the segmentation task. Our primary aim is to evaluate the pixel accuracy of the relevance map computed from the rendered CLIP features. We can further post-process and refine our 2D segmentation results by SAM (Kirillov et al., 2023) and get the results shown as 'Ours-Refined'

5.3.1 Hybrid CLIP Feature

In this ablation study, we investigated using a single scale of CLIP features at the finest scale level, rather than our hybrid CLIP features, which are obtained by averaging multiple-scale CLIP features extracted from patches at different resolutions. As demonstrated in Table 3, the hybrid CLIP feature for supervision is greatly important. The scene understanding capability is severely compromised when employing only a single-scale CLIP feature for supervision (denoted as *'W/O Hybrid CLIP'*). The inferior performance observed with the use of a single-scale CLIP feature stems from its potential inadequacy in capturing sufficient contextual information within the image. By contrast, our hybrid CLIP feature encompasses a significantly larger receptive field, enhancing its contextual awareness.

To conduct a comprehensive analysis, we also compare our method of using a single hybrid CLIP supervision (*'Ours'*) to the approach of utilizing CLIP supervision at multiple individual scales, as proposed by LERF (Kerr et al., 2023), which we denote as *'W/ LERF CLIP'*. During its training, randomly sampled scale factors are concatenated with the intermediate MHE feature vector $\mathbf{q}$ (refer to Sec. 4.1) to decode CLIP features at each scale. At inference time, the optimal scale factors at each pixel location are determined by an exhaustive search among 30 uniformly distributed scales. The best scale factor is selected based on its relevance score to the query. Consequently, this approach is at least 30 times slower than our method with a single hybrid CLIP supervision during inference. Despite its inefficiency, the accuracy achieved by *' W/ LERF CLIP'* (78.8%) significantly lags behind that of *'Ours'* (93.2%), as shown in Table 3, underscoring the efficacy and superior efficiency of our proposed method. It is worth noting that while it is plausible that 'W/ LERF CLIP' could achieve better performance with much more training iterations, this aspect falls beyond the scope of our current investigation.

5.3.2 Pixel-Alignment Loss

To assess the effectiveness of our proposed pixel alignment loss, we conducted an ablation study by training our seman-

tic scene representation without this loss. The impact of omitting the pixel alignment loss on the accuracy of the object detection task is shown in Table 3. Furthermore, we provide qualitative results in Fig. 6, which indicates that CLIP features from a scene representation trained with pixel-alignment loss are better at distinguishing between different objects and separating objects from the background.

5.3.3 Scene Representation

To evaluate the effectiveness of our hybrid scene presentation, which integrates 3D Gaussians for geometry and appearance representation alongside MHE for efficient semantic embedding, we compare it with a vanilla scene representation method that solely employs Gaussians without MHE by attaching semantic embeddings to each Gaussian (*'W/O MHE'*). Notably, for fair comparisons, we consider only the selected Gaussians with sufficient opacity and 2D radius in the vanilla method (as detailed in Sec. 4.4) identical to our method (*'Ours'*). These attached semantic embeddings share the same dimensionality as the intermediate MHE feature vector **q** in our method (see Sec. 4.1). They are also decoded by two MLPs, MLP_{ϕ}^{CLIP} and MLP_{ψ}^{DINO}, to obtain CLIP and DINO features while rendering the feature maps.

As indicated in Table 3, *'W/O MHE'* achieves an object detection accuracy of 84.4%, which is inferior to *'Ours'* with its hybrid scene representation. The superior performance of *'Ours'* can be attributed to its ability to render CLIP feature maps at a higher quality over *'W/O MHE'*. Additionally, in terms of scene representation complexity, the MHE component of our method maintains a constant number of parameters across the five scenes of the LERF dataset, whereas the parameter count of the semantic embeddings in *'W/O MHE'* varies with the number of Gaussians and increases by 84.1%, 113.5%, 2.14%, 174.2%, and 136.9% compared to *'Ours'*. This substantial memory demand of *'W/O MHE'* can pose even greater challenges in large-scale scenarios.

6 Discussion and Limitations

When comparing FMGS to LERF (Kerr et al., 2023), both methods distill Clip and Dino features from foundation models into 3D scene representations. However, their rendering algorithms and scene representations differ significantly. These distinctions lead to rapid and high-quality language feature acquisition using common hyperparameters, such as the feature field architecture. An additional key advantage of FMGS is that it employs the same feature embedding for each Gaussian, regardless of the viewing direction. This feature enables direct 3D localization of vision-language queries. It's important to note that FMGS not only facilitates the localization of language queries in 3D but also allows for finding a given image of the scene using the 3D Gaussian embeddings. LERF, on the other hand, does not offer such 3D localization capabilities out of the box.

In terms of limitations, FMGS currently relies heavily on the presence of high-quality and calibrated input images, a limitation shared with NeRF-based approaches. Additionally, the performance of FMGS is entirely contingent on the quality of the base foundation models used for training the feature fields. It is conceivable that a model better suited for localizing language within images could yield improved feature field quality. Furthermore, to enhance performance in semantic segmentation tasks, it is advisable to embed a specialized segmentation foundation model, such as SAM (Kirillov et al., 2023; Cen et al., 2023), into our scene representation. Unlike using SAM solely for post-refinement of our 2D segmentation results, directly distilling SAM results into the 3D space offers the advantage of enforcing multi-view consistency, potentially leading to improved performance.

7 Conclusions

Foundation Model Embedded Gaussian Splatting (FMGS) contributes to scene understanding by seamlessly merging vision-language embeddings and 3D representation. This novel 3D scene representation achieves multi-view semantic consistency through self-supervised distillation and pixel alignment of CLIP features. The resulting feature-embedded 3D Gaussians achieve state-of-the-art performance in comparison to previous methods. By bridging vision, language, and 3D, FMGS paves the way for unprecedented object comprehension in real-world environments, opening exciting possibilities for augmented reality, robotics, and beyond.

Acknowledgements We are very grateful to Juan J. Gómez Rodríguez and Francis Engelmann for their advice and insightful discussions about this work.

Data Availability Figures. 3, 5, 6, 7, 8, and Tables 1, 3 are experimental results on the the publicly available data shared in Kerr et al. (2023). Figure 9 and Table 2 are experimental results on the publicly available data shared in Liu et al. (2023). All the intermediate experimental results and data shown in this paper can be accessed from the first author by request [https://xingxingzuo.github.io/fmgs/].

References

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. (2022). Flamingo: A visual language model for few-shot learning. *Advances in Neural Information Processing Systems, 35*, 23716–23736.

Amir, S., Gandelsman, Y., Bagon, S., & Dekel, T. (2021). *Deep vit features as dense visual descriptors., 2*(3), 4. arXiv:2112.05814.

Azuma, D., Miyanishi, T., Kurita, S., & Kawanabe, M. (2022). Scanqa: 3d question answering for spatial scene understanding. *In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 19129–19139).

Barron, J. T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., & Srinivasan, P. P. (2021). Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. *In Proceedings of the IEEE/CVF international conference on computer vision* (pp. 5855–5864).

Barron, J. T., Mildenhall, B., Verbin, D., Srinivasan, P. P., & Hedman, P. (2022). Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5470–5479).

Barron, J. T., Mildenhall, B., Verbin, D., Srinivasan, P. P., Hedman, P. (2023). Zip-nerf: Anti-aliased grid-based neural radiance fields. In *ICCV*.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., & Joulin, A. (2021). Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 9650–9660).

Cascante-Bonilla, P., Hui, W., Wang, L., Feris, R. S., & Ordonez, V. (2022). Simvqa: Exploring simulated environments for visual question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5056–5066).

Cen, J., Zhou, Z., Fang, J., Yang, C., Shen, W., Xie, L., Zhang, X., & Tian, Q. (2023). Segment anything in 3D with nerfs. In *NeurIPS* .

Chen, A., Xu, Z., Geiger, A., Yu, J., & Su, H. (2022). Tensorf: Tensorial radiance fields. In *European conference on computer vision* (pp. 333–350). Springer.

Chen, G., & Wang, W. (2024). A survey on 3D Gaussian splatting. arXiv preprint arXiv:2401.03890.

Chen, H., Blomqvist, K., Milano, F., & Siegwart, R. (2023). Panoptic vision-language feature fields. arXiv preprint arXiv:2309.05448.

Cherti, M., Beaumont, R., Wightman, R., Wortsman, M., Ilharco, G., Gordon, C., Schuhmann, C., Schmidt, L., & Jitsev, J. (2022). Reproducible scaling laws for contrastive language-image learning. arXiv preprint arXiv:2212.07143.

Corona, R., Zhu, S., Klein, D., & Darrell, T. (2022). Voxel-informed language grounding. arXiv preprint arXiv:2205.09710.

Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., & Nießner, M. (2017). Scannet: Richly-annotated 3D reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5828–5839).

Fei, B., Xu, J., Zhang, R., Zhou, Q., Yang, W., & He, Y. (2024). 3D Gaussian as a new vision era: A survey. arXiv preprint arXiv:2402.07181.

Ghiasi, G., Gu, X., Cui, Y., & Lin, T.-Y. (2021). Open-vocabulary image segmentation. arXiv preprint arXiv:2112.12143.

Gordon, D., Kembhavi, A., Rastegari, M., Redmon, J., Fox, D., & Farhadi, A. (2018). Iqa: Visual question answering in interactive environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4089–4098).

Grinvald, M., Furrer, F., Novkovic, T., Chung, J. J., Cadena, C., Siegwart, R., & Nieto, J. (2019). Volumetric instance-aware semantic mapping and 3d object discovery. *IEEE Robotics and Automation Letters, 4*(3), 3037–3044.

Gu, Q., Kuwajerwala, A., Morin, S., Jatavallabhula, K. M., Sen, B., Agarwal, A., Rivera, C., Paul, W., Ellis, K., Chellappa, R., Gan, C., Miguel de Melo, C., Tenenbaum, J. B., Torralba, A., Shkurti, F., & Paull, L. (2023). Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. arXiv.

Gu, X., Lin, T.-Y., Kuo, W., & Cui, Y. (2021). Open-vocabulary object detection via vision and language knowledge distillation. arXiv preprint arXiv:2104.13921.

Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., & Davison, A., et al. (2011). Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology* (pp. 559–568).

Jatavallabhula, K. M., Kuwajerwala, A., Gu, Q., Omama, M., Chen, T., Li, S., Iyer, G., Saryazdi, S., Keetha, N., Tewari, A., Tenenbaum, J. B., de Melo, C. M., Krishna, M., Paull, L., Shkurti, F., & Torralba, A. (2023). Conceptfusion: Open-set multimodal 3d mapping.

Karkus, P., Cai, S., & Hsu, D. (2021). Differentiable slam-net: Learning particle slam for visual navigation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 2815–2825).

Keetha, N., Karhade, J., Jatavallabhula, K. M., Yang, G., Scherer, S., Ramanan, D., & Luiten, J. (2023). Splatam: Splat, track & map 3d gaussians for dense rgb-d slam. arXiv preprint arXiv:2312.02126.

Kerbl, B., Kopanas, G., Leimkühler, T., & Drettakis, G. (2023). 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG), 42*(4), 1–14.

Kerr, J., Kim, C. M., Goldberg, K., Kanazawa, A., & Tancik, M. (2023). Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 19729–19739).

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, L. C., Tete X., Gustafson, W., Spencer, B., Alexander, C., & Wan-Yen, L., et al. (2023). Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 4015–4026).

Kobayashi, S., Matsumoto, E., & Sitzmann, V. (2022). Decomposing nerf for editing via feature field distillation. In *Advances in neural information processing systems* volume 35.

Li, B., Weinberger, K. Q., Belongie, S., Koltun, V., & Ranftl, R. (2022). Language-driven semantic segmentation. arXiv preprint arXiv:2201.03546.

Liang, F., Wu, B., Dai, X., Li, K., Zhao, Y., Zhang, H., Zhang, P., Vajda, P., & Marculescu, D. (2022). Open-vocabulary semantic segmentation with mask-adapted clip. arXiv preprint arXiv:2210.04150.

Lin, K., Wang, L., & Liu, Z. (2021). End-to-end human pose and mesh reconstruction with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 1954–1963).

Liu, K., Zhan, F., Zhang, J., Xu, M., Yu, Y., El Saddik, A., Theobalt, C., Xing, E., & Lu, S. (2023). Weakly supervised 3d open-vocabulary segmentation. In *Thirty-seventh conference on neural information processing systems*.

Liu, K., Zhan, F., Zhang, J., Xu, M., Yu, Y., Saddik, A. E., Theobalt, C., Xing, E., & Lu, S. (2023). 3d open-vocabulary segmentation with foundation models. arXiv preprint arXiv:2305.14093.

Lu, S., Chang, H., Jing, E. P., Boularias, A., & Bekris, K. (2023). OVIR-3d: Open-vocabulary 3d instance retrieval without training on 3d data. In *7th annual conference on robot learning*.

Lu, Y., Xu, C., Wei, X., Xie, X., Tomizuka, M., Keutzer, K., & Zhang, S. (2023). Open-vocabulary point-cloud object detection without 3d annotation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Lüddecke, T., & Ecker, A. (2022). Image segmentation using text and image prompts. In *CVPR* (pp. 7086–7096).

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., & Ramamoorthi, R. (2020). Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*.

Minderer, M., Gritsenko, A., Stone, A., Neumann, M., Weissenborn, D., Dosovitskiy, A., Mahendran, A., Arnab, A., Dehghani, M., & Shen, Z., et al. (2022). Simple open-vocabulary object detection with vision transformers. arXiv preprint arXiv:2205.06230.

Müller, T., Evans, A., Schied, C., & Keller, A. (2022). Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG), 41*(4), 1–15.

Narita, G., Seno, T., Ishikawa, T., & Kaji, Y. (2019). Panopticfusion: Online volumetric semantic mapping at the level of stuff and things. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 4205–4212).

Qi, C. R,. Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 652–660).

Qin, M., Li, W., Zhou, J., Wang, H., & Pfister, H. (2024). Langsplat: 3d language gaussian splatting. In *Proceedings of the IEEE/CVF international conference on computer vision*.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., & Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning* (pp. 8748–8763). PMLR.

Schonberger, J. L., & Frahm, J.-M. (2016). Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4104–4113).

Schonberger, J. L., & Frahm, J.-M. (2016). Structure-from-motion revisited. *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4104–4113).

Schöps, T., Sattler, T., & Pollefeys, M. (2019). Surfelmeshing: Online surfel-based mesh reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 42*(10), 2494–2507.

Shafiullah, N. M. M., Paxton, C., Pinto, L., Chintala, S., & Szlam, A. (2022). Clip-fields: Weakly supervised semantic fields for robotic memory. arXiv preprint arXiv:2210.05663.

Shi, J.-C., Wang, M., Duan, H.-B., & Guan, S.-H. (2024). LEGaussians: Language embedded 3d gaussians for open-vocabulary scene understanding. In *Proceedings of the IEEE/CVF international conference on computer vision*.

Sun, J., Xie, Y., Chen, L., Zhou, X., & Bao, H. (2021). Neuralrecon: Real-time coherent 3d reconstruction from monocular video. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 15598–15607).

Takmaz, A., Fedele, E., Sumner, R. W., Pollefeys, M., Tombari, F., & Engelmann, F. (2023). OpenMask3D: Open-vocabulary 3D instance segmentation. In *Advances in neural information processing systems (NeurIPS)*.

Thomason, J., Shridhar, M., Bisk, Y., Paxton, C., & Zettlemoyer, L. (2022). Language grounding with 3d objects. In *Conference on robot learning* (pp. 1691–1701).

Tsagkas, N., Mac A. O., & Lu, C. X. (2023). Vl-fields: Towards language-grounded neural implicit spatial representations. arXiv preprint arXiv:2305.12427.

Tschernezki, V., Laina, I., Larlus, D., & Vedaldi, A. (2022). Neural feature fusion fields: 3D distillation of self-supervised 2D image representations. In *3DV*.

Wang, Z., Lu, Y., Li, Q., Tao, X., Guo, Y., Gong, M., & Liu, T. (2022). Cris: Clip-driven referring image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11686–11695).

Xu, Q., Xu, Z., Philip, J., Bi, S., Shu, Z., Sunkavalli, K., & Neumann, U. (2022). Point-nerf: Point-based neural radiance fields. *CoRR*. arXiv: 2201.08845.

Xue, L., Gao, M., Xing, C., Martín-Martín, R., Wu, J., Xiong, C., Xu, R., Niebles, J. C., & Savarese, S. (2022). Ulip: Learning unified representation of language, image and point cloud for 3d understanding. arXiv preprint arXiv:2212.05171.

Xue, L., Yu, N., Zhang, S., Li, J., Martín-Martín, R., Wu, J., Xiong, C., Xu, R., Niebles, J. C., & Savarese, S. (2023). Ulip-2: Towards scalable multimodal pre-training for 3d understanding.

Yang, X., Zhou, L., Jiang, H., Tang, Z., Wang, Y., Bao, H., & Zhang, G. (2020). Mobile3drecon: Real-time monocular 3d reconstruction on a mobile phone. *IEEE Transactions on Visualization and Computer Graphics, 26*(12), 3446–3456.

Ye, J., Wang, N., & Wang, X. (2023). Featurenerf: Learning generalizable nerfs by distilling pre-trained vision foundation models. arXiv preprint arXiv:2303.12786.

Yifan, W., Serena, F., Shihao, W., Öztireli, C., & Sorkine-Hornung, O. (2019). Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG), 38*(6), 1–14.

Ze, Y., Yan, G., Yueh-Hua, W., Macaluso, A., Ge, Y., Ye, J., & Hansen, N. (2023). CoRL: Multi-task real robot learning with generalizable neural feature fields.

Zhang, R., Guo, Z., Zhang, W., Li, K., Miao, X., Cui, B., Qiao, Y., Gao, P., & Li, H. (2021). Pointclip: Point cloud understanding by clip. arXiv preprint arXiv:2112.02413.

Zhou, S., Chang, H., Jiang, S., Fan, Z., Zhu, Z., Xu, D., Chari, P., You, S., Wang, Z., & Kadambi, A. (2024). Feature 3DGS: Supercharging 3d Gaussian splatting to enable distilled feature fields. In *Proceedings of the IEEE/CVF international conference on computer vision*.

Zhou, X., Girdhar, R., Joulin, A., Krähenbühl, P., & Misra, I. (2022). Detecting twenty-thousand classes using image-level supervision. In *ECCV* (pp. 350–368). Springer.

Zwicker, M., Pfister, H., Van Baar, J., & Gross, M. (2001). Ewa volume splatting. In *Proceedings visualization*.