Online Language Splatting

Saimouli Katragadda[1], Cho-Ying Wu[2], Yuliang Guo[2*], Xinyu Huang[2], Guoquan Huang[1], Liu Ren[2]
[1]University of Delaware, [2]Bosch Research North America & Bosch Center for Artificial Intelligence (BCAI)
{saimouli,ghuang}@udel.edu
{Cho-Ying.Wu,yuliang.guo2,xinyu.huang,liu.ren}@us.bosch.edu

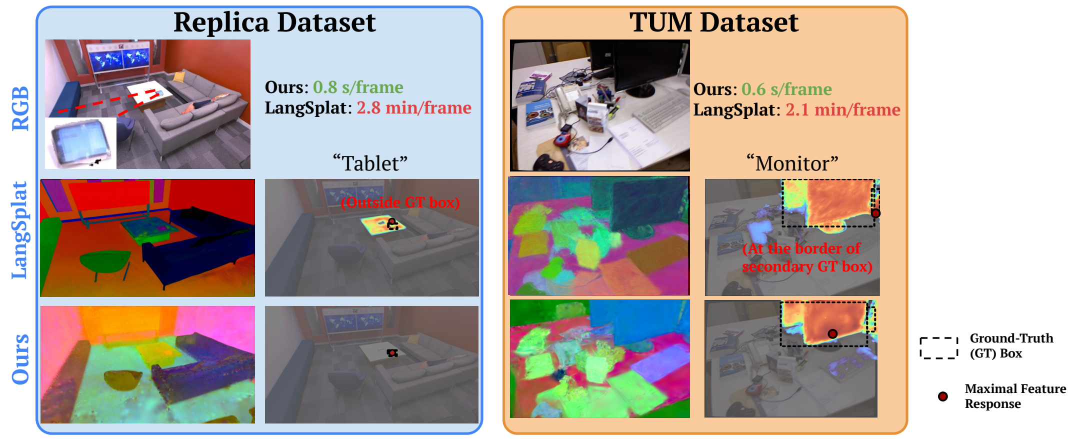

Figure 1. We introduce an **online language splatting** framework that incrementally constructs a 3D Gaussian-based language feature map using 3D Gaussian Splatting within a SLAM framework. Results are compared to the offline state-of-the-art LangSplat [36] across two datasets, presented in two panels. For each panel, the left column displays rendered language feature maps, and the right column shows target localization results. Our method not only outperforms in localization accuracy, but also achieve more than 40× improved efficiency.

Abstract

*To enable AI agents to interact seamlessly with both humans and 3D environments, they must not only perceive the 3D world accurately but also align human language with 3D spatial representations. While prior work has made significant progress by integrating language features into geometrically detailed 3D scene representations using 3D Gaussian Splatting (GS), these approaches rely on computationally intensive offline preprocessing of language features for each input image, limiting adaptability to new environments. In this work, we introduce **Online Language Splatting**, the first framework to achieve online, near real-time, open-vocabulary language mapping within a 3DGS-SLAM system without requiring pre-generated language features. The key challenge lies in efficiently fusing high-dimensional language features into 3D representations while balancing the computation speed, memory usage, rendering quality and open-vocabulary capability. To this end, we innovatively design: (1) a high-resolution CLIP embedding module capable of generating detailed language feature maps in 18ms per frame, (2) a two-stage online auto-encoder that compresses 768-dimensional CLIP features to 15 dimensions while preserving open-vocabulary capabilities, and (3) a color-language disentangled optimization approach to improve rendering quality. Experimental results show that our online method not only surpasses the state-of-the-art offline methods in accuracy but also achieves more than 40× efficiency boost, demonstrating the potential for dynamic and interactive AI applications.*

1. Introduction

Radiance Fields [17, 32, 39] have emerged as a transformative technology for 3D scene representation. Among them, 3D Gaussian Splatting (GS) [17] has become particularly popular due to its high rendering quality and efficiency in differentiable rendering research. While radiance fields provide detailed geometric and textured 3D representations for photorealistic image rendering, they lack the semantic information necessary for interaction with humans.

The integration of language features into 3D scene representations has recently enabled open-vocabulary language queries, improving both interpretability and interactivity in human-computer interaction [18, 36, 42, 60]. For example,

*Project Lead.

LangSplat [36] embeds CLIP-based language features [37] into 3DGS, including both RGB and language channels per Gaussian. However, existing Lang-GS methods typically rely on computationally intensive preprocessing to generate pixel-wise language features using multimodal foundation models like SAM+CLIP, which can require minutes per frame. This substantial computational overhead limits their applicability to offline scenarios, where language features must be precomputed for each frame.

While offline language mapping is sufficient for static, predefined environments, many real-world applications demand immediate scene understanding. For instance, a service robot entering a new environment must quickly perceive the 3D surroundings to follow commands, and augmented reality (AR) systems need to deliver instant, interactive feedback as users explore new spaces. Recent advancements in combining Gaussian Splatting with online mapping [6, 9, 15, 27, 56] have enabled detailed geometric and textured maps to be created in near real-time. However, these approaches do not incorporate language features, focusing solely on geometry and texture. Alternatively, methods that use pre-annotated ground-truth semantic maps [13, 21, 22] simplify the problem but are limited to closed-vocabulary settings, lacking the flexibility required for open-vocabulary commanding.

The key challenge in online 3D language mapping lies in efficiently integrating language features into 3D representations while preserving open-vocabulary capabilities. To address this, we introduce **Online Language Splatting**, the first framework to achieve near real-time, open-vocabulary 3D language mapping within a SLAM-GS system, eliminating the need for pre-generated language maps. Fig. 1 illustrates the proposed framework. In particular, our method addresses three core sub-challenges: (1) **Real-time High-Resolution CLIP Embedding:** Since offline, segment-centric CLIP feature preparation is a major runtime bottleneck, we replace it with a single-stage CLIP embedding and a Super-Resolution Decoder (SRD) module, enabling the generation of detailed, pixel-aligned CLIP maps in 18 ms per frame (Sec. 4.1). (2) **Open-Vocabulary-Preserving Feature Compression in Novel Scenes:** Unlike offline methods, which allow feature compression modules to be trained on the test scene, online methods rely on a pre-trained feature compressor to operate directly on unseen data. However, due to domain gaps, a single pre-trained autoencoder may struggle to maintain open-vocabulary capabilities when compressing CLIP features for online mapping. To address this online-specific generalization challenge, we introduce a two-stage autoencoder, where the second stage, an Online-Learned AutoEncoder (OLAE), dynamically adapts to the dominant data variance of the current scene. This further reduces feature dimensions while preserving critical information (Sec. 4.2). (3) **Color-Language Optimization:** Existing Lang-GS systems jointly optimized color and language using the same GS parameters, but these modalities inherently prefer distinct GS parameters (see Fig. 4). Prior work [36, 42, 60] jointly optimized RGB and language using shared GS parameters, which failed to achieve optimal performance for either modality. To address this, we disentangle RGB and language backpropagation paths, designing a set of separate GS parameters to effectively render high-quality outputs for both modalities (Sec. 4.3).

Building on these designs, our extensive experiments demonstrate that our approach not only surpasses prior state-of-the-art (SoTA) offline Lang-GS methods in text-queried 2D and 3D object localization and segmentation but also delivers a $40\times$ to $200\times$ boost in efficiency.

In summary, the main contributions of this paper include:

- We introduce the first near real-time, open-vocabulary, online language splatting framework, enabling flexible interaction with human language.
- We tackle key challenges in online language splatting by proposing a real-time high-resolution CLIP embedding, an open-vocabulary-preserving feature compressor, and a color-language disentangled optimization strategy.
- Through comprehensive evaluation, we demonstrate that our method outperforms prior state-of-the-art offline approaches across most of key metrics while achieving over 40× efficiency gains.

2. Related Work

2.1. SLAM with Differentiable Rendering

Dense visual SLAM builds 3D maps in an online fashion, typically using classical representations such as voxel grids [5, 28, 29, 33, 52], Octrees [47, 48, 55], or point cloud [1, 2, 16, 40, 41, 53]. In recent years, differentiable rendering has gained popularity in SLAM, enabling the joint optimization of camera poses, maps, and implicit representations such as neural fields [11, 14, 39, 45, 49, 61] or explicit 3D Gaussians [6, 9, 10, 15, 27, 35, 56] with manageable computational overhead. In particular, MonoGS [27] introduces a highly efficient pipeline for online camera tracking and mapping with high-quality rendering [9], leveraging CUDA-based gradient updates to optimize camera poses. While these methods excel in rendering quality and accurate camera pose estimation, they focus solely on geometric and photometric optimization. We emphasize that our approach complements existing SLAM-GS methods [15, 27, 35] by introducing a novel capability: 3D open-vocabulary language mapping, extending the utility of SLAM-GS systems for more interactive applications. For further details refer to [3].

2.2. Language 3D Gaussian Splatting

Lang-GS methods [36, 42, 60] have recently emerged to integrate language mapping into the GS framework. Given

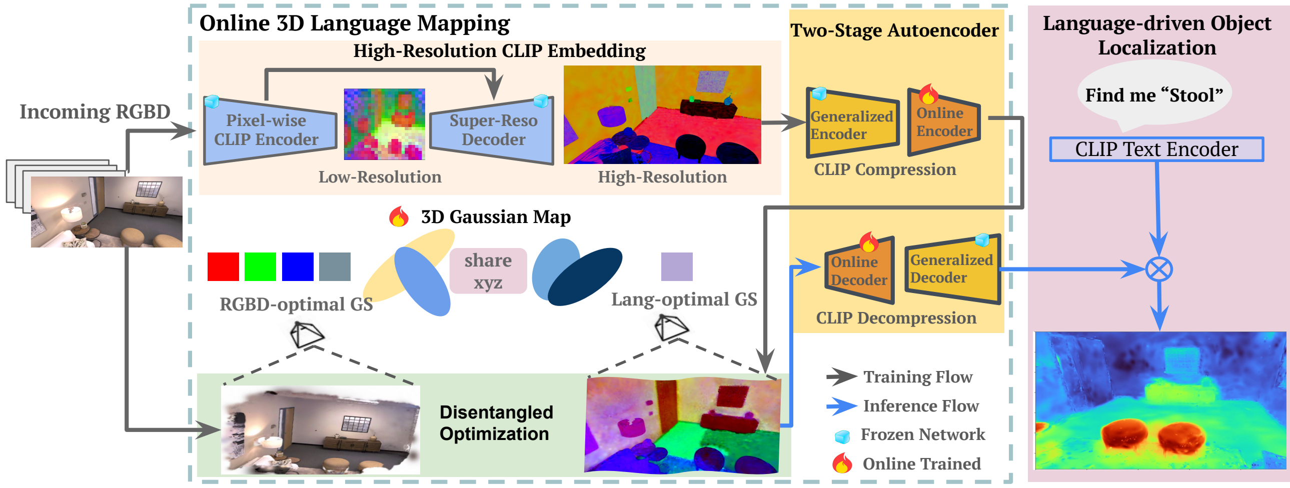

Figure 2. **Online Language Splatting Pipeline.** Our pipeline integrates 3D Gaussian Splatting with SLAM, using 3D Gaussians as the sole mapping elements. **Left:** During training, raw images are processed through a High-Resolution (HR) CLIP embedding module, which generates HR language features in real-time. These features are compressed via a two-stage CLIP compression module into low-dimensional maps for efficient optimization while preserving open-vocabulary capabilities. RGB and language parameters are optimized separately through disentangled optimization to accommodate distinct preference in update 3D Gaussian map. **Right:** At inference, the rendered low-dimensional language map undergoes a two-stage decoding process to reconstruct the full-resolution CLIP feature map, enabling open-vocabulary queries to locate target objects.

the effectiveness of GS, embedding both language features and RGB channels into 3D Gaussians has been shown to outperform previous NeRF-based methods [18] in both accuracy and rendering efficiency. However, most of these methods rely on offline-prepared language maps generated by SAM+CLIP, leading to significant processing time for open-vocabulary segmentation. Some GS-SLAM methods [13, 21, 22] support semantic map rendering but simplify the task by using dataset-provided semantic maps as ground truth. This approach limits them to closed vocabularies, contradicts the online nature of SLAM, and reduces adaptability to new scenes. In contrast, our method is the first to achieve online open-vocabulary 3D language mapping, enabling seamless adaptation to novel environments.

2.3. Open-Vocabulary Detection and Segmentation

Open-Vocabulary Detection (OV-D) and Segmentation (OV-S) have gained traction with the advent of CLIP [37] and large vision-language models, enabling natural language prompts for querying. OV-D typically employs a large backbone encoder and probing heads to predict bounding boxes and classes based on cosine similarity with text embeddings [4, 20, 25, 30, 31, 51]. OV-S is more challenging, requiring fine-grained masks and pixel-level semantics. The encoders for vision-language models generally suffer from limited bottleneck feature resolution [34, 37]. Many OV-S approaches [23, 36, 38, 46, 50, 57, 58] use class-agnostic mask generation via proposal networks or SAM [19], followed by vision-language model processing on proposal regions. However, these methods are computationally intensive and unsuitable for online applications. As a consequence, offline Lang-GS methods [36, 42, 60] adopting SAM-based CLIP input typically takes several minutes per image on high-end GPUs to label language field ground-truth. In contrast, we adopt a highly efficient OV-S encoding approach [54] that directly embeds CLIP features into the network's bottleneck feature map at a low spatial resolution. To overcome the spatial resolution for precise 3D language mapping, our proposed SRD module not only reconstructs high-resolution CLIP maps but operates in real-time.

3. Preliminaries

3DGS and Rendering In 3D Guassian fields, each Gaussian $\mathcal{G}_i, i \in [1, \mathcal{N}]$ is represented by its 3D world-coordinate positions $\boldsymbol{\mu}_i \in \mathbb{R}^3$, covariance matrix $\boldsymbol{\Sigma}_i \in \mathbb{R}^{3\times3}$, colors $\boldsymbol{c}_i \in \mathbb{R}^3$, and opacity $\alpha_i \in \mathbb{R}$. We drop spherical harmonics the same as in the prior online 3DGS [15, 27, 56]. The pixel color C is rendered by front-to-back composition of overlapping Gaussians sorted by depth:

$$C = \sum_{i\in\mathcal{N}} \boldsymbol{c}_i\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j) =: \sum_{i\in\mathcal{N}} \boldsymbol{c}_i\alpha_i T \tag{1}$$

where T denotes the transmittance. Note that opacity α_i is after decay by the Gaussian function w.r.t. projected 2D Gaussian: $\boldsymbol{\mu}_{2D} = \mathbf{KP}\boldsymbol{\mu}$, $\boldsymbol{\Sigma}_{2D} = \mathbf{JR\Sigma R}^T\mathbf{J}^T$, where $\mathbf{K}$ is the camera intrinsics, $\mathbf{P}$ is the world-to-camera projection matrix, $\mathbf{R}$ is the rotation, and $\mathbf{J}$ is the Jacobian of the affine approximation of the projective transformation.

SLAM Tracking and Keyframing We adopt tracking and keyframing mechanism in MonoGS [27]. For each

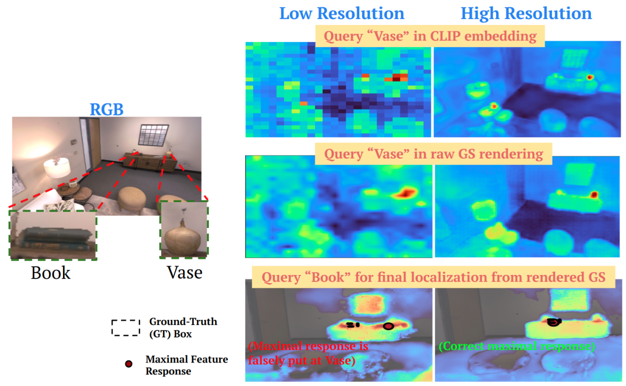

Figure 3. **Importance of Feature Map Resolution. Top:** Compared to the Low Resolution (LR) query heatmap from the pixel-wise encoder output (left), the High Resolution (HR) heatmap from SRD output (right) improves localization and differentiation. **Middle & Bottom:** Query heatmaps from rendered maps after GS mapping. GS mapping from LR exhibits feature bleeding, while mapping from HR preserves structural details, better localization.

tracked frame, current camera pose is optimized with appearance and geometry loss.

$$\mathcal{L} = \lambda |C^r - C^{gt}| + (1-\lambda)|D^r - D^{gt}|, \quad (2)$$

where C^r and D^r are rendered image and depth by alpha composition in Eq. (1). (For depth rendering, the color term is replaced by z-direction distance at the center of a Gaussian.) Tracked frames are selected as keyframes within a local window after a co-visibility check, ensuring sufficient novel regions are visible in each keyframe and avoiding redundant optimization. For each keyframe, new Gaussians are created in the 3D maps with $\boldsymbol{\mu}$ initialized by back-projecting depth to cover the areas. The 3DGS parameters are optimized by the maintained keyframe window with appearance and geometry loss Eq. (2) plus scale-isotropic regularization to prevent serious needle-like artifacts.

Multi-Channel Optimization Prior work (e.g., [36]) that embeds semantics or language features into 3DGS caches additional channels per Gaussian. The language map rendering follows alpha-blending rules:

$$F = \sum_{i \in \mathcal{N}} \boldsymbol{f}_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (3)$$

where $\boldsymbol{f}_i \in \mathbb{R}^3$ is language feature in each Gaussian. During the backward pass, language gradients are entangled with color and depth gradients:

$$\frac{\partial \mathcal{L}}{\partial \alpha_i} = \frac{\partial \mathcal{L}}{\partial C}\frac{\partial C}{\partial \alpha_i} + \frac{\partial \mathcal{L}}{\partial D}\frac{\partial D}{\partial \alpha_i} + \frac{\partial \mathcal{L}}{\partial F}\frac{\partial F}{\partial \alpha_i}, \quad (4)$$

where the loss $\mathcal{L}$ is based on Eq.(2) with an additional L1 loss using groundtruth language maps via SAM+CLIP. Some works [13, 21, 22] use gradients in Eq.(4) to train with multi-modality online. To not let language features interfere with color Gaussians, LangSplat's offline optimization first trains color Gaussians on the whole sequence without the last gradient term, and in the second stage they use the last term to pass gradient to only train language features.

4. Online Language Splatting

Our approach enables near real-time, high-resolution, Open-Vocabulary (OV) language mapping within a 3D Gaussian Splatting (3DGS) framework, facilitating language-driven spatial understanding for robotics and AR applications. As shown in Fig. 2, our pipeline consists of three main components during the training and optimization phase, addressing the key challenges outlined in Sec. 1.

The pipeline begins with standard RGB-D SLAM input streams. Color images are processed through a pixel-wise CLIP encoder to generate low-resolution language features. These features, combined with hierarchical encoder outputs, are then refined by a Super-Resolution Decoder (SRD) to produce pixel-aligned, high-resolution language maps. Next, the CLIP Compression module, implemented as a Two-Stage Autoencoder, significantly reduces the dimensionality of CLIP features for efficient online mapping while preserving essential information for OV queries. The second stage, an Online-Learned Autoencoder (OLAE), further enhances generalization to novel scenes. Finally, Disentangled Optimization separates gradient flows for color and language, enabling independent optimization of Gaussian parameters. This improves rendering quality across both modalities. During inference, the rendered low-dimensional language map can be passed through the Two-Stage Autoencoder to reconstruct full CLIP features, allowing OV queries for locating target objects.

4.1. High-Resolution CLIP Embedding

Unlike offline methods that require multiple passes and complex mask generation, our approach leverages a ConvNeXt-based pixel-wise CLIP Encoder [54] to generate a coarse CLIP embedding map, which is then refined by a lightweight **Super-Resolution Decoder (SRD)** to produce dense, high-quality language maps. This design preserves conceptual integrity while enabling real-time operation. The SRD takes a coarse CLIP map along with the intermediate outputs from layers 1 and 2 of the pixel-wise encoder as inputs, progressively enhancing the CLIP feature map resolution through two convolutional upsampling blocks that align with hierarchical encoder features. The detailed architecture is illustrated in Supplemental Fig. 8.

The Super-Resolution Decoder (SRD) is trained with supervision from offline high-resolution CLIP feature maps. Following a procedure similar to [36], we generate high-resolution language features to serve as training labels for our lightweight SRD. Our training is not restricted to a specific dataset, as we only require diverse images to cover a broad range of concepts without relying on their original annotations. When training images encompass diverse concepts (e.g., COCO [24]), and our training focuses on the simplified task of upsampling the feature map, the OV capability of the CLIP features is expected to be preserved.

The resulting high-resolution CLIP embedding module (pixel-wise CLIP Encoder + SRD) operates highly efficiently, achieving a runtime of 18 ms on an RTX-3090 GPU while using only 1.6 GB of GPU memory. The SRD submodule contributes only 2 ms to this runtime, significantly improving feature quality with minimal overhead. These enhancements in turn result in improved accuracy and IoU (see Table 1, Fig. 7). The benefits of high-resolution CLIP maps are further illustrated at the feature level in Fig. 3.

Note that our SRD design shares certain similarities with FeatUp [8], as both approaches focus on upsampling feature maps. However, unlike their unsupervised method, we employ a simpler and more efficient strategy by using hierarchical feature supervision to guide upsampling in a supervised manner. Our approach not only enhances accuracy for high-resolution images but also improves computational efficiency (see supplemental Fig. 11 for comparison).

4.2. Two-Stage Online CLIP Compression

Since CLIP features are high-dimensional (768) vectors, a key challenge is how to effectively compress them to enable real-time integration while preserving OV capabilities.

To address this, we first develop a generalized language compressor that exploits the inherent redundancy in language feature embeddings. Using diverse images from a large dataset (e.g., COCO), we train a simple autoencoder baseline with a multi-layer MLP to compress the dimensionality from 768 to a 32-dimensional code. This code size is carefully chosen to balance semantic preservation and data compression. Due to the domain gap between the pretraining dataset and the test scenes, the output dimension cannot be too low, as excessive compression may compromise OV capabilities when applied to new domains. Supplemental Table 11 provides a detailed analysis of code size selection and its impact on performance.

While the generalized language compressor effectively reduces dimensionality, the resulting code size remains too large for efficient integration into an online Lang-GS framework. To further compress the CLIP feature while preserving their OV capability, we introduce an **Online-Learned AutoEncoder (OLAE)** as a second-stage compressor, which adapts dynamically to testing scenes by compressing features into a smaller 15-dimensional code. This adaptation is based on the observation that data variance within a specific scene can often be captured by fewer dimensions, allowing less relevant directions from the generalized model to be disregarded. The OLAE starts with an initial training phase of 200 iterations (6 ms/iter) and incrementally updates using selected keyframes. For each iteration, two additional random keyframes are incorporated, ensuring retention of previously learned features and preventing catastrophic forgetting. By combining a generalized compressor (for broad vocabulary preservation) and an online-learned compressor (for scene adaptability), our approach maintains OV capabilities while significantly reducing memory cost, making real-time applications feasible.

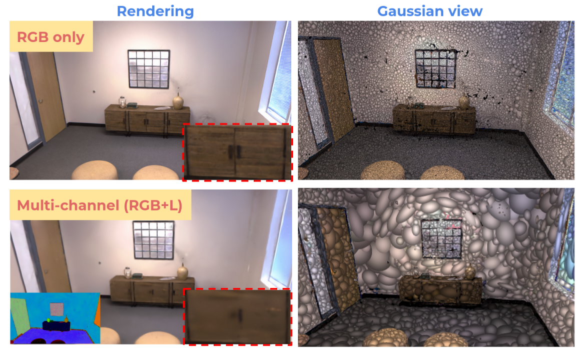

Figure 4. **Color optimization alone vs. color-language joint optimization.** Adding language channels leads to sub-optimal rendering quality with much different Gaussian maps.

4.3. Color-Language-Disentangled Optimization

One of the key challenges in online 3DGS is how to optimizing color and language modalities in the meantime. Prior methods like LangSplat use additional channels in each Gaussian to represent language features, as described in Sec. 3. Colors and language modality share the common 3DGS parameters, including α, $\boldsymbol{\mu}$, and $\boldsymbol{\Sigma}$ (or the same scale $\boldsymbol{S}$ and rotation $\boldsymbol{R}$). However, we find that sharing the common parameters will lead to suboptimal performance for both colors and language. To verify, we visualize the Gaussian fields for training colors alone and jointly optimizing RGB-L channels in Fig. 4. The color rendering deteriorates when jointly optimizing RGB-L. Observing Fig. 4, we find that this is because language features tend to stretch Gaussian scales and apply different rotations. Unlike color appearance with textural details, language features are homogeneous, such as wall areas are associated with the same language codes. Therefore, language Gaussians prefer much different GS parameters from RGB.

Losses of color and language are chained to the same GS parameters by Eq. (4), where $\partial\mathcal{L}/\partial\alpha_i$ further back-propagates to $\boldsymbol{\mu}, \boldsymbol{R}, \boldsymbol{S}$. However, if using different sets of whole GS parameters, the number of Gaussians can be high and takes up the memory, but the key rendering contributors could still be sparse. To efficiently represent colors and language, we adopt multi-mode $\boldsymbol{R}$, $\boldsymbol{S}$, and α for colors and languages, and they still share the same $\boldsymbol{\mu}$ in each Gaussian to prevent Gaussian duplication. The color and language rendering become

$$C = \sum_{i \in \mathcal{N}} \boldsymbol{c}_i \alpha_i^c \prod_{j=1}^{i-1}(1-\alpha_j^c), \quad F = \sum_{i \in \mathcal{N}} \boldsymbol{f}_i \alpha_i^f \prod_{j=1}^{i-1}(1-\alpha_j^f), \tag{5}$$

and for depth rendering we use α^c. The back-propagation becomes

$$\frac{\partial\mathcal{L}}{\partial\alpha_i^c} = \frac{\partial\mathcal{L}}{\partial C}\frac{\partial C}{\partial\alpha_i^c} + \frac{\partial\mathcal{L}}{\partial D}\frac{\partial D}{\partial\alpha_i^c}, \quad \frac{\partial\mathcal{L}}{\partial\alpha_i^f} = \frac{\partial\mathcal{L}}{\partial F}\frac{\partial F}{\partial\alpha_i^f}, \tag{6}$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{R}_i^{c/f}} = \frac{\partial \mathcal{L}}{\partial \alpha_i^{c/f}} \frac{\partial \alpha_i^{c/f}}{\partial \boldsymbol{\Sigma}_i^{c/f}} \frac{\partial \boldsymbol{\Sigma}_i^{c/f}}{\partial \boldsymbol{R}_i^{c/f}}, \quad \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}_i^{c/f}} = \frac{\partial \mathcal{L}}{\partial \alpha_i^{c/f}} \frac{\partial \alpha_i^{c/f}}{\partial \boldsymbol{\Sigma}_i^{c/f}} \frac{\partial \boldsymbol{\Sigma}_i^{c/f}}{\partial \boldsymbol{S}_i^{c/f}}, \tag{7}$$

where script c/f denotes different modes. We empirically simplify the back-propagation path to $\partial\mathcal{L}/\partial\boldsymbol{\mu}_i^{2D}$ by only computing $\partial\mathcal{L}/\partial\alpha_i^c \times \partial\alpha_i^c/\partial\boldsymbol{\mu}_i^{2D}$ without language. MonoGS [27] further computes gradients to camera poses ($\boldsymbol{P}$) by $\partial\boldsymbol{\mu}_i^{2D}/\partial\boldsymbol{P}$ and $\partial\boldsymbol{\Sigma}_i^{2D}/\partial\boldsymbol{P}$, and we also drop the language part and only use color mode's mean and covariance for camera poses. For co-visibility check in keyframing, we require both colors and language are with sufficient novel areas to join as keyframes. To prune a Gaussian, we also require both α_i^c and α_i^f are below a threshold. Last, to not let language mode learn skewed scales, we add a loss $|\boldsymbol{S}_i^f - \boldsymbol{S}_{i\perp}^c|$, where $_\perp$ denotes stop gradient.

5. Experiments

Baselines. Since we introduce the first online Language Gaussian Splatting (Lang-GS) method, we primarily compare our approach to state-of-the-art (SoTA) offline Lang-GS methods, including LangSplat [36], Feature3DGS [60], and LEGaussian [42][1] in text query-based object localization. Additionally, to demonstrate that integrating language mapping does not degrade SLAM system performance, we compare our method against SoTA SLAM-GS approaches in image rendering and camera pose tracking, including MonoGS [27], SplaTAM [15], and RTG-SLAM [35].

Datasets. We conduct evaluations on two widely used datasets for the SLAM setup: the synthetic Replica [43] and the challenging real-world TUM RGB-D [44], both qualitatively and quantitatively. In Replica, we evaluate the top 10 most frequent classes, sampling 21 frames randomly from each sequence as test frames. In TUM RGB-D, we manually annotate test frames to create ground-truth masks for language queries, serving as evaluation targets. In training, we utilize COCO [24] and Omnidata [7] datasets due to their diverse range of concepts, ensuring broad generalization across various scenes and objects.

Evaluation Metrics. To assess object localization via text queries, we follow LangSplat to use mIoU and localization accuracy (Loc) on rendered language maps. Localization is considered successful if the highest-relevancy pixel falls within the ground-truth bounding box. To evaluate image rendering quality, we use PSNR, SSIM, and LPIPS, while camera pose tracking is assessed via ATE RMSE, following the evaluation protocol in [27]. We exclude the post-stage global refinement for strict online requirements. Each method's runtime is measured by per-frame processing time. Besides these known metrics, we also evaluate text-query 3D localization via Chamfer Distance (CD) and Earth Mover's Distance (EMD) between the ground-truth and localized point sets.

[1]LEGaussian's reported Replica results are based on re-annotated and simplified groundtruth. We re-evaluate it for fair comparison.

Table 1. **Comparison to Lang-GS SoTA on Replica**. Our method is compared to the SoTA Lang-GS methods on the Replica dataset in terms of image-based localization accuracy and per-frame running time. We also analyze the impact of key introduced modules, including Super-Resolution Decoder (SRD) in CLIP Embedding and Online Learning of AutoEncoder (OLAE) in feature compression. The variants without OLAE train a single AE from other scenes in Replica Dataset.

Method	**Modules**		**Query Loc.**		**Time**
	SRD	OLAE	mIOU	Loc	
LangSplat [36]	–	–	0.417	0.720	2.8 min/fr
Feature3DGS [60]	–	–	0.359	0.755	2.3 min/fr
LEGaussian [42]	–	–	0.245	0.682	32 s/fr
Ours	✗	✗	0.400	0.754	**0.8 s/fr**
	COCO	✗	0.475	0.782	
	COCO	✓	0.479	0.759	
	Omni	✗	0.485	0.802	
	Omni	✓	**0.487**	**0.826**	

Table 2. **Comparison to Lang-GS SoTA on TUM RGB-D.** Our method is compared to the Lang-GS SoTA method LangSplat on image-based localization accuracy and running time.

TUM RGB-D	**Scene1**		**Scene2**		**Time**
	mIOU	**Loc**	**mIOU**	**Loc**	
LangSplat [36]	**0.646**	0.850	**0.538**	0.7825	2.1 min/fr
Ours	0.599	**0.917**	0.535	**0.7905**	**0.6 s/fr**

Implementation Details. We utilize a pre-trained CLIP ViT-L model [37] alongside a ConvNeXt-L-based hierarchical encoder from [54] to extract 768-dimensional feature representations from input images [26]. The input to this module is an RGBD image with dimensions 640×640×3. The module processes the input and produces a feature map of size 24×24×768. Subsequently, the SRD enhances this feature map to an output resolution of 192×192×768, maintaining the semantic context of the input data. We train two separate SRD models: one on 7% of the COCO dataset [24] and another on 30% of the Omnidata-Tiny dataset [7]. Both models are trained on four A5000 GPUs with a batch size of 12 images per GPU. We utilize the AdamW optimizer with an initial learning rate of 2×10^{-4} and a weight decay of 1×10^{-4} for a total of 180 training epochs. For the generalized auto-encoder, a 8-layer MLP architecture was used to compress language features to a code size of 32. For the online compressor, a 2 layer MLP with a encoder code of size 15 is trained online using a scheduler with a reduction on plateau and a threshold of 1×10^{-4}, and optimized with Adam using a learning rate of 1×10^{-3}. The online training takes 10 key frames with 200 iterations in initialization, and update 1 iteration in the consecutive frames. To ensure a fair comparison, we upgrade LangSplat's OpenCLIP [12] model to match our feature dimensions (768, up from 512), use a code size of 15 (from 3), and train the pipeline offline on the entire Replica and TUM RGBD dataset, sampling every 10th image from each sequence.

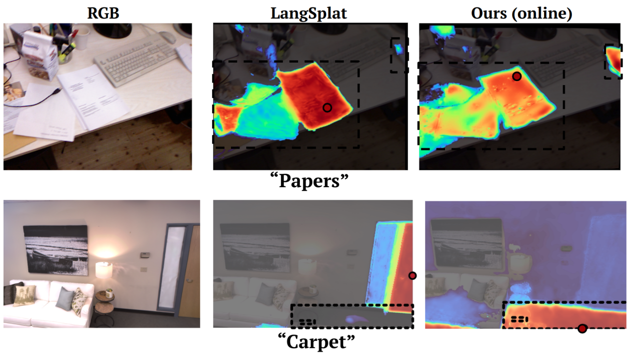

Figure 5. **Qualitative comparison with offline SoTA: Top**: On the TUM RGB-D dataset, our method successfully segments the paper in the top-right corner, which LangSplat fails to detect. **Bottom**: On the Replica dataset, we accurately localize the carpet, whereas LangSplat misidentifies a different object. Black box: ground-truth box; red dot: maximal feature response as the predicted localization.

5.1. Comparison with the State of the Art

Comparison to Lang-GS SoTA Methods. The comparison between our method and previous SoTA offline Lang-GS methods is presented in Table 1 and Table 2. As observed, our method establishes a new SoTA performance on the Replica dataset, significantly surpassing offline methods, regardless of whether SRD is trained on COCO or Omnidata datasets. It also leads to improved localization accuracy and competitive mIoU scores on the TUM-RGBD dataset upon LangSplat. As an online method, our approach is $40\times$ to $200\times$ more efficient than SoTA offline methods. A detailed per-scene evaluation on the Replica dataset is provided in supplemental Table 7. Qualitatively, Fig. 5 shows that our method correctly identifies objects that LangSplat either misses or misidentifies. The ablation study of key modules are further discussed in Sec 5.2.

On the other hand, our performance advantage on the TUM-RGBD dataset is less pronounced. This is primarily due to challenges such as motion blur and lower image quality, which complicate online camera tracking. These conditions favor offline approaches that rely on extensive global optimization (e.g., 30k iterations) of both 3D Gaussian parameters and camera poses.

Comparison to SLAM-GS SoTA Methods. We compare our method to recent SoTA SLAM-GS approaches in Table 3. As observed, despite integrating language feature mapping into a SLAM-GS system, our framework preserves novel view rendering performance and localization is on par to MonoGS [27], the SLAM framework on which we build. Compared to other SoTA SLAM-GS methods, our approach achieves the best overall performance in novel view rendering while incorporating language mapping. RTG-SLAM achieves superior ATE due to the inclusion of additional classical SLAM modules, which increases system complexity. Per-scene results are provided in supplemental Table 8.

Table 3. **SLAM-GS Evaluation on Replica**. Our method is evaluated against other SLAM-GS approaches based on novel view rendering quality and camera localization error (ATE in cm).

Method	Lang.	PSNR ↑	SSIM ↑	LPIPS ↓	ATE (cm) ↓
SplaTAM [15]	✗	33.39	0.968	0.101	0.392
RTG-SLAM [35]	✗	35.77	**0.982**	0.106	**0.182**
MonoGS [27]	✗	35.72	0.950	0.075	0.420
Ours	✓	**35.81**	0.950	**0.072**	0.397

3D Localization Evaluation. For 3D localization, we first fuse multi-view language renderings into voxels by truncated signed distance field (TSDF). Each voxel size is 20cm. Then, we extract mesh by marching cubes. Each vertex is described by language features. We pass the batched point features into the CLIP decompressor, reconstruct the 768 dimensions, and compare with the mesh build by TSDF with semantic mask groundtruth. We compare with LangSplat on the devised 3D localization protocol. Table 4 demonstrates that our framework can run in an online setting with better 3D language mapping. Fig. 6 provides visualization of the 3D localization. See the supplementary Sec 14 for more results.

Table 4. **3D Localization Evaluation.** Our online method is compared to LangSplat in 3D localization from language query.

Method	Online	3D Language Localization							
		cabinet		cushion		stool		rug	
		CD↓	EMD↓	CD↓	EMD↓	CD↓	EMD↓	CD↓	EMD↓
LangSplat	✗	**0.03**	**0.02**	**0.23**	0.10	**0.16**	**0.13**	0.26	**0.38**
Ours	✓	0.04	0.05	0.55	**0.03**	0.20	0.40	**0.26**	1.49

Method	Online	3D Language Localization							
		lamp		wall		ceiling		**Average**	
		CD↓	EMD↓	CD↓	EMD↓	CD↓	EMD↓	CD↓	EMD↓
LangSplat	✗	1.64	38.3	0.14	0.18	0.55	**0.30**	0.43	5.63
Ours	✓	**1.22**	**4.08**	**0.09**	**0.05**	**0.27**	0.71	**0.38**	**0.97**

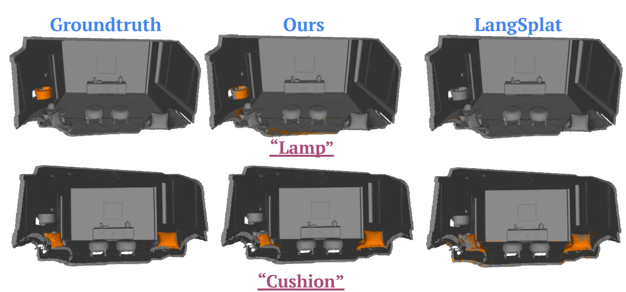

Figure 6. **Visualization of 3D localization results.**

Runtime Analysis. Our entire network module runs at 21ms per frame on an RTX-3090 GPU, including 15ms for CLIP encoding, 2ms for super-resolution decoding, and 6ms for online compression with online training. While the overall pipeline speed is currently bottlenecked by the MonoGS baseline—resulting in a runtime of 0.6–0.8 seconds per frame—significantly higher speeds are achievable with advancements in the SLAM-GS system. For instance, by integrating our method into Hi-SLAM [59], we achieve 7.05 FPS. In contrast, the offline method LangSplat requires approximately 168s per frame (2.8 minutes), including 35s for SAM, 10s for post-processing, and an additional 123s per frame (amortized) for training the dense CLIP autoencoder on the testing scene. This total runtime underscores the significant computational cost of an offline approach.

5.2. Ablation Study

Super-Reso Decoder (SRD) in CLIP Embedding We analyze the impact of SRD on the Replica dataset, with results summarized in Table 1 and individual scene results provided in supplemental Table 7. We observe that SRD significantly improves both mIoU and Loc metrics from our basic online baseline. The underlying reasons for these improvements are evident through visual comparisons in Fig. 3 and Fig. 7. From Fig. 3, we can see that high-resolution language maps greatly enhance localization of small or distant objects. Fig. 7 demonstrates improvements in semantic boundaries and even helps resolve ambiguities between visually similar but different classes. **Cross-view consistency:** In a new ablation study, we found that the generalized encoder improves cross-view consistency—without it, the average cosine similarity drops to 0.5. While frame-wise encoding alone cannot fully ensure cross-view consistency, the globally optimized 3D map from Gaussian Splatting (GS) inherently maintains consistency across views.

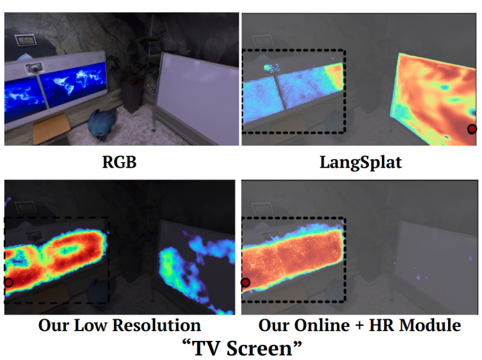

Figure 7. **Visual comparison in CLIP embedding:** Our High-resolution embedding allows for more complete object capture, reducing holes and resolving semantic ambiguities. The dotted black box represents the ground-truth, while the red dot indicates the maximal feature response as the predicted localization.

Online Learning of AutoEncoding (OLAE). The ablation study on Online Learning of AutoEncoder (OLAE) in CLIP Compression is summarized in Table 1 and supplemental Table 7. To evaluate the effect of removing the online encoder strategy, we train a single autoencoder using 4-fold cross-validation on the Replica dataset. In each fold, two sequences are held out for testing, while the remaining sequences are used for training. This setup ensures that the trained modules are exposed to data from the Replica domain while making that the testing scenes remain unseen.

Table 5. **Impact of Color-Language Disentanglement (disent.) versus Joint Multi-Channel Optimization (joint).**

Method	2D Loc.		3D Loc.		Image Rendering			Tracking
	mIOU↑	Loc↑	EMD↓	CD ↓	PSNR↑	SSIM↑	LPIPS↓	ATE[cm]↓
Joint	0.323	**0.633**	0.384	1.940	31.23	0.901	0.197	0.796
Disent.	**0.402**	0.622	**0.375**	**0.974**	**35.89**	**0.957**	**0.060**	**0.325**

As shown in Table 1, introducing OLAE even surpasses the in-domain fine-tuned single autoencoder, demonstrating its effectiveness in preserving semantic concepts upon compression. A more detailed per-scene analysis (supplemental Table 7) further reveals: Although in-domain fine-tuned pipelines tend to outperform on certain testing scenes similar to those observed during training, OLAE performs better on novel scenes and shows higher overall stability across all scenes. A more detailed ablation study on code size is provided in supplemental Sec. 12.1.

Color-Language Disentanglement. We evaluate the impact of disentanglement by comparing it to joint multi-channel optimization commonly used in other online multi-modality learning frameworks [13, 21, 22]. This study uses the Replica Room-0 subset to examine the design's impact on 2D / 3D localization accuracy, novel-view image rendering quality and SLAM tracking errors (ATE [cm]), with results shown in Table 5.

The results demonstrate that disentangling color and language significantly enhances 2D mIoU, both 3D metrics, color rendering, and camera tracking, while maintaining comparable 2D localization performance. A more detailed ablation study on the impact of GS parameters is provided in supplemental Sec 13, confirming that this strategy enables the two modalities to operate with their optimal GS parameters, minimizing interference between them.

6. Conclusion

In this work, we introduce Online Language Splatting, a framework that enables online language-aware 3D mapping through key innovations. First, a real-time Super-Resolution Decoder (SRD) enhances CLIP embeddings, generating detailed language maps. Second, an highly effective and efficient two-stage CLIP compression preserving open-vocabulary capabilities. Third, a color-language disentangled optimization improves rendering quality for both language and color images. Our experimental results demonstrate that our online approach not only outperforms offline SoTA Lang-GS methods, but also leads to orders of magnitude efficiency improvement.

References

[1] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021. 2

[2] Daniele Cattaneo, Matteo Vaghi, and Abhinav Valada. Lcdnet: Deep loop closure detection and point cloud registration for lidar slam. *IEEE Transactions on Robotics*, 38(4):2074–2093, 2022. 2

[3] Guikun Chen and Wenguan Wang. A survey on 3d gaussian splatting. 2025. 2

[4] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. Yolo-world: Real-time open-vocabulary object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16901–16911, 2024. 3

[5] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (ToG)*, 36(4): 1, 2017. 2

[6] Tianchen Deng, Yaohui Chen, Leyan Zhang, Jianfei Yang, Shenghai Yuan, Jiuming Liu, Danwei Wang, Hesheng Wang, and Weidong Chen. Compact 3d gaussian splatting for dense visual slam. *arXiv preprint arXiv:2403.11247*, 2024. 2

[7] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10786–10796, 2021. 6

[8] 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 (ICLR)*, 2024. 5, 12, 15

[9] Xinli Guo, Peng Han, Weidong Zhang, and Hongtian Chen. Motiongs: Compact gaussian splatting slam by motion filter. *arXiv preprint arXiv:2405.11129*, 2024. 2

[10] Jiarui Hu, Xianhao Chen, Boyin Feng, Guanglin Li, Liangjing Yang, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Cg-slam: Efficient dense rgb-d slam in a consistent uncertainty-aware 3d gaussian field. In *European Conference on Computer Vision (ECCV)*, pages 93–112. Springer, 2024. 2

[11] Huajian Huang, Longwei Li, Hui Cheng, and Sai-Kit Yeung. Photo-slam: Real-time simultaneous localization and photorealistic mapping for monocular stereo and rgb-d cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21584–21593, 2024. 2

[12] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021. 6, 14

[13] Yiming Ji, Yang Liu, Guanghu Xie, Boyu Ma, Zongwu Xie, and Hong Liu. Neds-slam: A neural explicit dense semantic slam framework using 3d gaussian splatting. *IEEE Robotics and Automation Letters*, 2024. 2, 3, 4, 8

[14] Mohammad Mahdi Johari, Camilla Carta, and François Fleuret. Eslam: Efficient dense slam system based on hybrid representation of signed distance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17408–17419, 2023. 2

[15] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat track & map 3d gaussians for dense rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21357–21366, 2024. 2, 3, 6, 7, 14

[16] Maik Keller, Damien Lefloch, Martin Lambers, Shahram Izadi, Tim Weyrich, and Andreas Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *International Conference on 3D Vision (3DV)*, pages 1–8. IEEE, 2013. 2

[17] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*, 2023. 1

[18] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19729–19739, 2023. 1, 3

[19] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026, 2023. 3

[20] Weicheng Kuo, Yin Cui, Xiuye Gu, AJ Piergiovanni, and Anelia Angelova. Open-vocabulary object detection upon frozen vision and language models. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023. 3

[21] Linfei Li, Lin Zhang, Zhong Wang, and Ying Shen. Gs3lam: Gaussian semantic splatting slam. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 3019–3027, 2024. 2, 3, 4, 8

[22] Mingrui Li, Shuhong Liu, Heng Zhou, Guohao Zhu, Na Cheng, Tianchen Deng, and Hongyu Wang. Sgs-slam: Semantic gaussian splatting for neural dense slam. In *European Conference on Computer Vision (ECCV)*, pages 163–179. Springer, 2024. 2, 3, 4, 8

[23] Feng Liang, Bichen Wu, Xiaoliang Dai, Kunpeng Li, Yinan Zhao, Hang Zhang, Peizhao Zhang, Peter Vajda, and Diana Marculescu. Open-vocabulary semantic segmentation with mask-adapted clip. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7061–7070, 2023. 3

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence

Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014. 4, 6

[25] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. 2024. 3

[26] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 11976–11986, 2022. 6

[27] Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18039–18048, 2024. 2, 3, 6, 7, 14

[28] John McCormac, Ronald Clark, Michael Bloesch, Andrew Davison, and Stefan Leutenegger. Fusion++: Volumetric object-level slam. In *2018 international conference on 3D vision (3DV)*, pages 32–41. IEEE, 2018. 2

[29] Maxime Meilland and Andrew I Comport. On unifying keyframe and voxel-based dense visual slam at large scales. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3677–3683. IEEE, 2013. 2

[30] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, et al. Simple open-vocabulary object detection. In *European Conference on Computer Vision (ECCV)*, pages 728–755. Springer, 2022. 3

[31] Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling open-vocabulary object detection. *Advances in Neural Information Processing Systems (NeurIPS)*, 36, 2024. 3

[32] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, pages 102:1–102:15, 2022. 1

[33] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 127–136. IEEE, 2011. 2

[34] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research (TMLR)*, 2024. 3

[35] Zhexi Peng, Tianjia Shao, Yong Liu, Jingke Zhou, Yin Yang, Jingdong Wang, and Kun Zhou. Rtg-slam: Real-time 3d reconstruction at scale using gaussian splatting. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 2, 6, 7, 14

[36] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20051–20060, 2024. 1, 2, 3, 4, 6, 13, 15, 16

[37] 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 (ICML)*, pages 8748–8763. PMLR, 2021. 2, 3, 6

[38] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024. 3, 13

[39] Antoni Rosinol, John J Leonard, and Luca Carlone. Nerf-slam: Real-time dense monocular slam with neural radiance fields. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3437–3444. IEEE, 2023. 1, 2

[40] Erik Sandström, Yue Li, Luc Van Gool, and Martin R Oswald. Point-slam: Dense neural point cloud-based slam. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18433–18444, 2023. 2

[41] Thomas Schops, Torsten Sattler, and Marc Pollefeys. Bad slam: Bundle adjusted direct rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 134–144, 2019. 2

[42] Jin-Chuan Shi, Miao Wang, Hao-Bin Duan, and Shao-Hua Guan. Language embedded 3d gaussians for open-vocabulary scene understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5333–5343, 2024. 1, 2, 3, 6, 13, 15

[43] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 6

[44] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, 2012. 6

[45] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International conference on computer vision (ICCV)*, pages 6229–6238, 2021. 2

[46] Shuyang Sun, Runjia Li, Philip Torr, Xiuye Gu, and Siyang Li. Clip as rnn: Segment countless visual concepts without training endeavor. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13171–13182, 2024. 3

[47] Emanuele Vespa, Nikolay Nikolov, Marius Grimm, Luigi Nardi, Paul HJ Kelly, and Stefan Leutenegger. Efficient octree-based volumetric slam supporting signed-distance and occupancy mapping. *IEEE Robotics and Automation Letters*, 3(2):1144–1151, 2018. 2

[48] Emanuele Vespa, Nils Funk, Paul HJ Kelly, and Stefan Leutenegger. Adaptive-resolution octree-based volumetric slam. In *2019 International Conference on 3D Vision (3DV)*, pages 654–662. IEEE, 2019. 2

[49] Hengyi Wang, Jingwen Wang, and Lourdes Agapito. Co-slam: Joint coordinate and sparse parametric encodings for neural real-time slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13293–13302, 2023. 2

[50] Haoxiang Wang, Pavan Kumar Anasosalu Vasu, Fartash Faghri, Raviteja Vemulapalli, Mehrdad Farajtabar, Sachin Mehta, Mohammad Rastegari, Oncel Tuzel, and Hadi Pouransari. Sam-clip: Merging vision foundation models towards semantic and spatial understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3635–3647, 2024. 3

[51] Jiong Wang, Huiming Zhang, Haiwen Hong, Xuan Jin, Yuan He, Hui Xue, and Zhou Zhao. Open-vocabulary object detection with an open corpus. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6759–6769, 2023. 3

[52] Thomas Whelan, Michael Kaess, Hordur Johannsson, Maurice Fallon, John J Leonard, and John McDonald. Real-time large-scale dense rgb-d slam with volumetric fusion. *The International Journal of Robotics Research*, 34(4-5):598–626, 2015. 2

[53] Thomas Whelan, Stefan Leutenegger, Renato F Salas-Moreno, Ben Glocker, and Andrew J Davison. Elasticfusion: Dense slam without a pose graph. In *Robotics: science and systems (RSS)*, page 3. Rome, Italy, 2015. 2

[54] Bin Xie, Jiale Cao, Jin Xie, Fahad Shahbaz Khan, and Yanwei Pang. Sed: A simple encoder-decoder for open-vocabulary semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3426–3436, 2024. 3, 4, 6

[55] Binbin Xu, Wenbin Li, Dimos Tzoumanikas, Michael Bloesch, Andrew Davison, and Stefan Leutenegger. Mid-fusion: Octree-based object-level multi-instance dynamic slam. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5231–5237. IEEE, 2019. 2

[56] Chi Yan, Delin Qu, Dan Xu, Bin Zhao, Zhigang Wang, Dong Wang, and Xuelong Li. Gs-slam: Dense visual slam with 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19595–19604, 2024. 2, 3

[57] Haobo Yuan, Xiangtai Li, Chong Zhou, Yining Li, Kai Chen, and Chen Change Loy. Open-vocabulary sam: Segment and recognize twenty-thousand classes interactively. In *European Conference on Computer Vision (ECCV)*, pages 419–437. Springer, 2024. 3

[58] Hao Zhang, Feng Li, Xueyan Zou, Shilong Liu, Chunyuan Li, Jianwei Yang, and Lei Zhang. A simple framework for open-vocabulary segmentation and detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1020–1031, 2023. 3

[59] Wei Zhang, Qing Cheng, David Skuddis, Niclas Zeller, Daniel Cremers, and Norbert Haala. Hi-slam2: Geometry-aware gaussian slam for fast monocular scene reconstruction. *arXiv preprint arXiv:2411.17982*, 2024. 8

[60] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21676–21685, 2024. 1, 2, 3, 6, 13, 15

[61] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 12786–12796, 2022. 2

[62] Xueyan Zou, Jianwei Yang, Hao Zhang, Feng Li, Linjie Li, Jianfeng Wang, Lijuan Wang, Jianfeng Gao, and Yong Jae Lee. Segment everything everywhere all at once. *Advances in neural information processing systems (NeurIPS)*, 36:19769–19782, 2023. 15