ManipuLang: Object Manipulation Through Language

Sanchit Tanwar
Georgia Institute of Technology
350 Ferst Dr, Atlanta, GA 30308
stanwar8@gatech.edu

Pushkar Raj Singh
Georgia Institute of Technology
350 Ferst Dr, Atlanta, GA 30308
psingh428@gatech.edu

Milad Heydari
Georgia Institute of Technology
350 Ferst Dr, Atlanta, GA 30308
miladgheydari@gmail.com

Pranav Datta
Georgia Institute of Technology
350 Ferst Dr, Atlanta, GA 30308
datta@gatech.edu

Abstract

Rendering and editing real environments virtually typically involve advanced software that requires expert users. Methods such as Gaussian Splatting have simplified the conversion of real-world images to 3D virtual scenes, but current processes of editing these scenes involve manual specification of tasks, identification of objects, and localization of interest regions (in the form of bounding boxes) for addition tasks. Therefore, modifying 3D scenes using open-vocabulary queries is an open problem. We propose a solution called ManipuLang, which integrates LLMs (GPT-4o and LLaVA-3D) for interpreting arbitrary language prompts for task and object identification. It then uses GroundedSAM to localize the object and removes the associated Gaussians in the scene. We also explore options for performing addition into the scene. We hope this offers an avenue for easily and accessibly manipulating 3D environments generated from real-world images.

1. Introduction

Editing 3D scenes is typically a high barrier of entry task. It requires a rendering of the environment, normally done using advanced modeling software to capture the various objects within the scene virtually. This process takes time, as each object may have to be manually created within the virtual world. The software that does so is typically proprietary and requires expertise, making the task limited to a small number of skilled users. The ability to edit 3D scenes through language by simply using pictures taken from multiple angles can increase the accessibility for people attempting tasks such as interior design, AR/VR modeling of environments, or synthetic data generation for robotics.

We built on the Gaussian Splatting method, which synthesizes and virtualizes multi-view images into a series of 3D Gaussians [9]. Two prior works have branched from this method, and also served as inspiration for our solution: LangSplat [17] and Gaussian Grouping [24]. The former embeds language priors into the Gaussian Splat, allowing for open-vocabulary queries of objects within the scene. The latter developed a method of editing Gaussian Splats in 5 different ways: removal, inpainting, colorization, style transfer, and scene recomposition. These two methods, however, had limitations that made them unsuitable for applying directly to our intended task. LangSplat has no editing capabilities and requires distinct, specific labels of objects; it cannot reason about relativity or localize objects respective to others. Gaussian Grouping has no language field and requires the manual specification of tasks and objects.

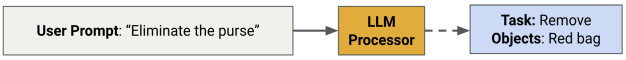


Figure 1: LLM parsing open-vocabulary prompt diagram.

Considering these methods and their limitations, our goal was to enable open-vocabulary editing of 3D Gaussian Splats, something not yet achieved by current prior works that we could find. For our scope, we focused on the removal and addition editing tasks. To enable open-vocabulary editing, we sought to integrate LLMs to parse and interpret arbitrary language prompts (Figure 1).

Our solution, known as ManipuLang, takes an arbitrary language query as input. Using GPT-4o [15], it identifies the task and subject of the task. It then localizes the subject to identify the name of the object using LLaVA-3D [25] and finally performs removal of that object using GroundedSAM (Figure 2) [19]. For addition, we explored current methods and evaluated various options for integrating the

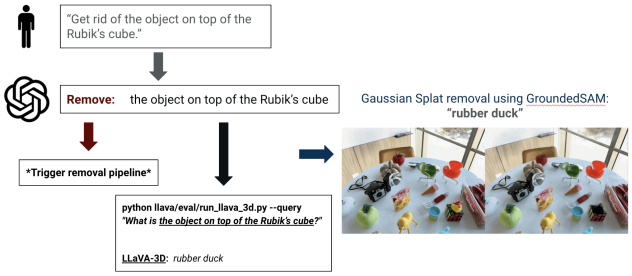


Figure 2: Process for performing removal.

task into our solution. We hope that this is a step forward toward making editing 3D environments, generated from real-world images, less complex and more accessible to non-expert users.

2. Related Work

Background: In this work, we primarily use **Gaussians** [9] as our fundamental building block in 3D scenes - similar to pixels in a 2D image. Each Gaussian is defined by a 3D position, scale, color, opacity, and rotation - representing an ellipsoid in 3D space. By placing many Gaussians in a scene and optimizing their properties to match the scene, 3D environments can represented in a way that is much easier to work with than traditional methods like Neural Radiance Fields (NeRF): Gaussians are explicit (directly manipulatable), efficient to render, and naturally handle varying levels of detail in scenes. Gaussian representations for a scene are created/optimized through rasterization-based rendering, which "trains" to create a good representation by projecting the 3D Gaussians onto 2D images and comparing against ground truth views and adjusting the Gaussians until the scene is both efficiently rendered and represents the scene well. We also briefly experiment with **3D point clouds**, which consist of discrete points in 3D space. These provide an initial structure for the scene but lack the smooth, continuous representation needed for high-quality rendering since each point is a distinct object and "dot" in the 3D space.

LangSplat: LangSplat [17] focuses on language-based 3D scene understanding. The method enables open-vocabulary querying by distilling CLIP embeddings into 3D Gaussian representations. LangSplat augments each 3D Gaussian in the scene with a compact Identity Encoding, which is trained through differentiable rendering to align with CLIP features extracted from 2D views, that can then be queried to find relevant objects in a scene. To improve efficiency, it introduces a scene-specific autoencoder during training that compresses the 512-dimensional CLIP embeddings into a low (3) dimensional latent space - this is one of the key contributions of the paper as it shows great performance along with massive efficiency gains. The method also addresses point ambiguity through SAM's hierarchical semantics by generating three distinct masks (whole, part, subpart) during training to handle different semantic levels and labels each Gaussian with information for each level of masking. LangSplat achieves a remarkable 199× speedup compared to NeRF-based approaches while maintaining superior performance in tasks like open-vocabulary localization and semantic segmentation. Furthermore, its reliance on SAM's scales means that relevancy maps cannot be queried at arbitrary granularity, limiting flexibility.

Gaussian Grouping: Gaussian Grouping [24] focuses on object-centric 3D scene representation. Each Gaussian is augmented with a learnable Identity Encoding. The method uses an explicit representation where 3D Gaussians are grouped according to their similarity with nearby Gaussians in the scene (at a simplified level, Gaussians that are touching are grouped to form an object). Instead of relying on expensive 3D labels, the method leverages 2D mask predictions from SAM during rendering, combined with 3D spatial consistency regularization to maintain accuracy. Grouping spatially coherent Gaussians enables direct manipulation of scene elements as discrete units, supporting various editing applications like object removal, inpainting, and scene recomposition. This approach ensures that objects and background elements can be fully decoupled, making complex scene modifications efficient and straightforward. However, Gaussian Grouping lacks direct language modeling capabilities (each object is just given a number-based ID), so it is less versatile than LangSplat for open-vocabulary querying. Additionally, the masks generated by the method do not inherently include semantic language information, which limits its use for text-driven scene editing. While it excels in segmentation quality and downstream editing tasks, its dependence on predefined semantic categories constrains its adaptability in dynamic, open-world scenarios where interaction through natural language is required.

Addition of objects in 3D Gaussian splats has been studied by many works recently. Some works frame this problem by first masking the region in a rendered 2D view, a text-based inpainting model [1] is then used to inpaint the masked region in these rendered views, and Gaussian splats are retrained with these updated views [4]. This creates an issue of inconsistency as 2D diffusion models are stochastic leading to differences in the output between views even with the same text prompt. A solution recently explored multi-view consistent diffusion models for inpainting [2].

Another body of work explores addition as a composition [7], [22] by directly adding "object" Gaussians into "scene" Gaussians. In this work, we model the problem as composition and especially look into text-based composition where we try to find the ideal location to add objects

based on text queries.

In summary, LangSplat works by, when given a language query, using CLIP embeddings trained for each Gaussian to compare with each possible Gaussian and return the ones that match the query. Since, during the querying processes, this evaluates the similarity between each Gaussian and the query without considering the nearby Gaussians, This process does not always inherently return a continuous group of Gaussians and is often noisy around the boundaries, so it is not useful for things like editing/moving/removal. In contrast, Gaussian Grouping does not operate on language and instead maps IDs to a set of Gaussians that are grouped and continuous in space and well-formed to create an object. These can therefore easily be edited/removed in the scene while maintaining consistency. The goal of this work is to combine the two approaches in such a way that we can both 1. use a language query and 2. map that to a continuous object grouping of Gaussians that can be edited as in the Gaussian grouping work. We also aim to allow for more complex language-based querying that involves reasoning about which objects to edit - something that LangSplat does not support.

3. Methods

3.1. Language Pre-Processor

To be able to handle more open-ended queries (for example, "there is something blocking the TV" rather than "remove the table"), we wanted to build processes for more open-ended queries. To do this, we found it was sufficient to use the custom GPT tool from OpenAI [16]. Building on a standard GPT 4o, we provided it with a custom prompt telling it to parse open-ended queries and gave it a set of 64 human-made examples. As the pre-processor is scene-unaware, it simply outputs 1. An action (remove/add) and 2. a description of the object from the query or context. The custom GPT tool also allows us to limit the outputs of the language model to a certain format using JSON fields, so we can ensure that it always outputs an action followed by the object description. Some examples are shown below:

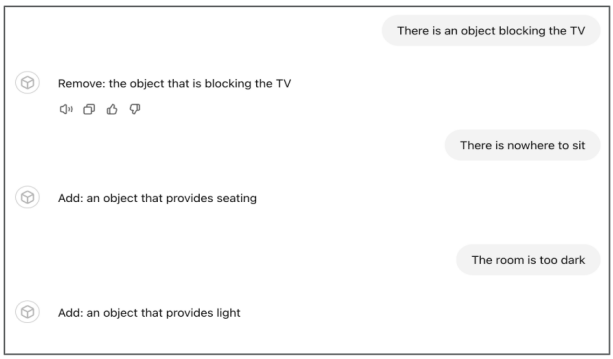


Figure 4: Custom GPT pre-processor

The output of the pre-possessor can is then parsed for:

1. action (add or remove which) triggers separate downstream pipelines

2. an object description

In the case of a non-exact object description, we then pass the description to a language model capable of reasoning. This step does not need to occur in 3D so we can simply use a VLM and pass in images of the scene. During our main experiments, we did this using LLaVA-3D [25] and processed the object description to get the actual object name. For example, the prompt "the object in front of the TV" returns an answer of "basket". Therefore, when running our entire language pre-processor, we were able to take a query like "there is an object blocking the TV" and output both an action, "remove", and the object, "basket", which we could then use to perform the action via our downstream processes. We also experimented with VLMs other than LLaVA-3D for our reasoning-based object detector in our ablations.

3.2. Scene Editing Pipeline

3.2.1 Dataset Curation

Object-level editing of a 3D scene requires a dataset with the decomposition of all the objects in the scene. We prepare this by using a zero-shot object tracking module, DEVA[6], on the multi-view images on the scene. DEVA uses SAM to automatically generate masks for each image of the multi-view collection. The 2D masks are individually produced per image. DEVA can associate the masks of the same identity across different views and we use this ability to assign all 2D masks of the same object a unique ID. This way of multi-view annotation and association results in over 60× speed up compared to linear assignment[20] based 2D mask matching as analysed in [24]

For the 3D language feature distillation version, we use DEVA in conjunction with GroundingDINO[12] and some predefined text prompts. We use the object embeddings from GroundingDino as the unique object ID in this case.

3.2.2 3D Consistent Mask Identities

To generate 3D-consistent mask identities across views of the scene, we group 3D Gaussians belonging to the same instance by adding an identity encoding parameter to each Gaussian similar to [24]. The Identity Encoding is a learnable and compact vector of length 16. During training, similar to the Spherical Harmonic coefficients of each Gaussian, we optimize the Identity Encoding vector to represent its instance ID in the scene. The instance ID is a view-independent parameter.

Following [9], the influence of all Gaussians on a single pixel location is computed by sorting the Gaussians in-depth order and blending $\mathcal{N}$ ordered points overlapping the

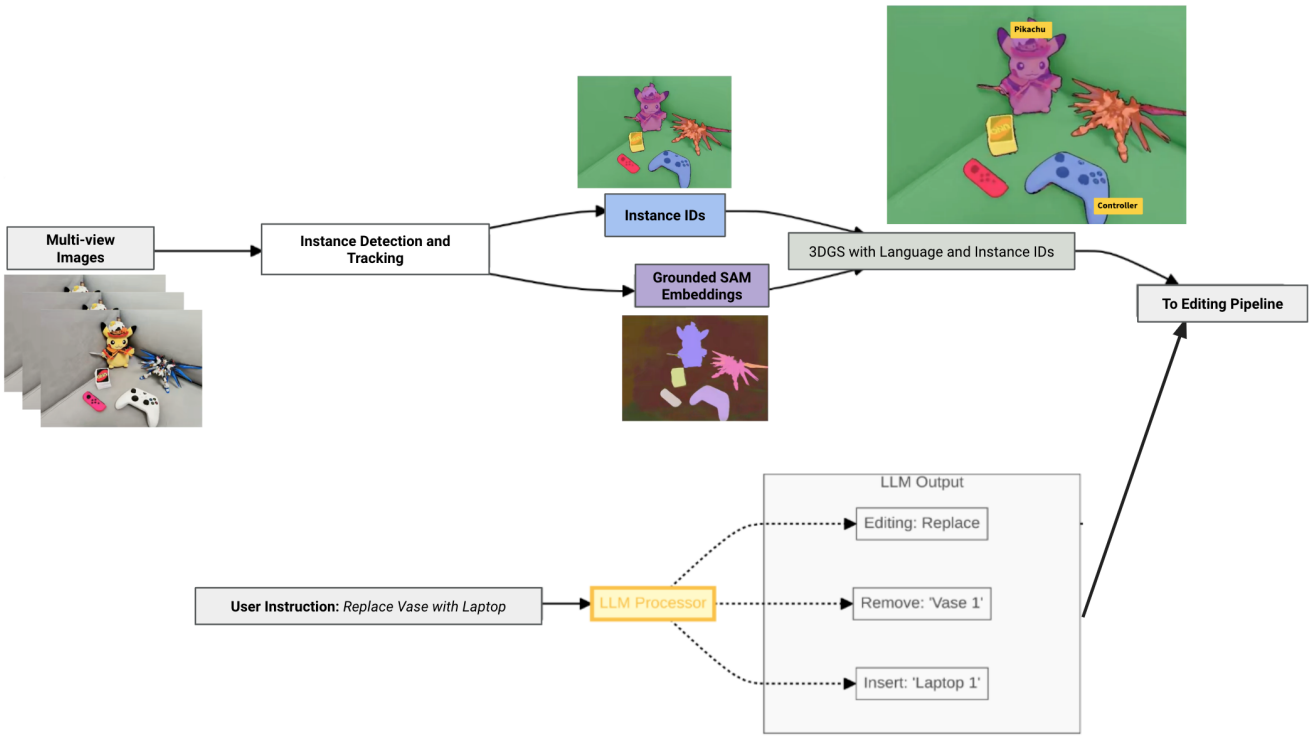


Figure 3: Block diagram of our editing pipeline. Our method consists of (i) a language pre-processor that reasons from the input user query the editing task to be performed and the objects to be edited, (ii) 3D langauge feature distilled Gaussian Splat representation which allows for open-vocabulary localization of objects. The identified objects to be edited are segmented in the 3D scene and the relevant edit is performed on the Gaussian contained within the segmented object.

pixels [13]:

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

where the final rendered 2D mask identity feature E_{id} for each pixel is a weighted sum over the Identity Encoding e_i of length 16 for each Gaussian, weighted by the Gaussian's influence factor α'_i on that pixel.

We use the following losses during Gaussian Splat training to group each Gaussian by the instance ID, like [24]:

1. ***2D Identity Loss***: A standard cross-entropy loss $\mathcal{L}_{\text{2D}}$ for K categories classification on the rendered 2D features.

2. ***3D Regularization Loss***: This Loss, denoted $\mathcal{L}_{\text{3D}}$, leverages the 3D spatial consistency, which enforces the Identity Encodings of the top k-nearest 3D Gaussians to be close in their feature distance.

Combined with the conventional 3D Gaussian Reconstruction Loss $\mathcal{L}_{\text{rec}}$ on image rendering [9], the total loss $\mathcal{L}_{\text{render}}$ becomes

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

3.2.3 3D Object Localizaiton

We experimented with two different ways of Gaussian localization for object removal:

1. ***2D Mask Matching***: In this version, we use GroundedSAM[19] for open-vocabulary 2D object segmentation on the multi-view images of the scene. [19] combines the open-set object detection capabilities of GroundingDINO with promptable segmentation of SAM to detect and segment anything with text inputs. Since we use 2D masks generated using SAM in our dataset for training 3D consistent masks, there is almost a one-to-one correspondence between each 2D rendered mask from the Gaussian Splat and the corresponding open-vocabulary segmented masks from GroundedSAM. Given the user query, we measure the IOUs of the 2D GroundedSAM mask with the rendered masks from the Gaussian Splat and localize the identity encoding of the best overlapping object. Some open vocabulary 2D segmentation results from GroundedSAM are shown in Figure 5.

2. ***3D Language Feature Distillation***: Instead of using 2D mask matching, another alternative is to query the object in 3D directly. To do so, we train the identity encoding to

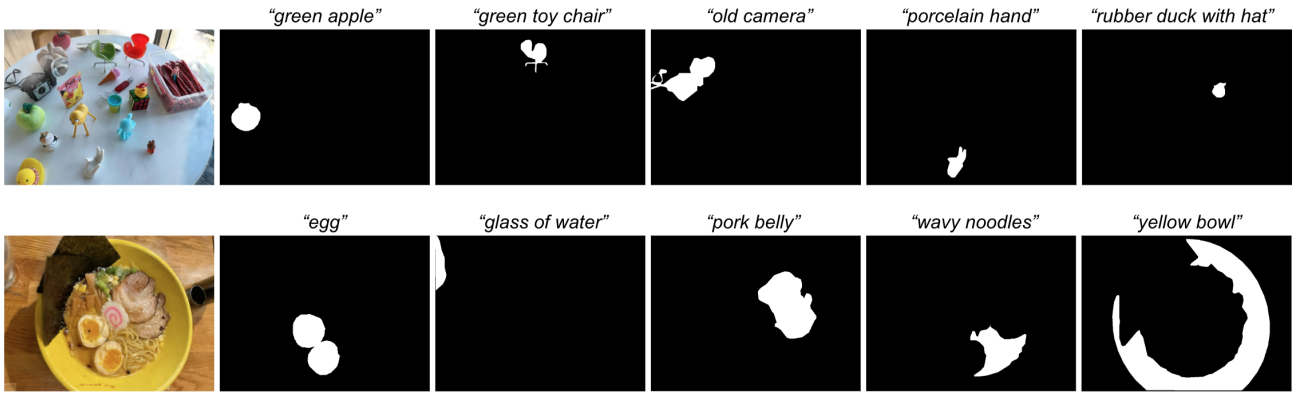


Figure 5: 2D segmentation masks with GroundedSAM on scenes from LERF-Mask dataset. In our 2D Mask Matching removal pipeline, the masks are matched with the rendered masks from the Gaussian Splat for object localization.

take the form of the 512-dimensional object embeddings, of GroundingDINO as a way to distill language features into the Gaussian Splat. To then match a user input text prompt involves passing the prompt through the GroundingDino text encoder based on BERT yielding a 512-dimensional text embedding. We then find the dot products of the text embedding with all the identity encodings to get the best matching group of Gaussians. The 3D distilled features result in slightly better localization as will be shown in the results section but are limited to the objects annotated during dataset preparation. This pipeline is shown in Figure 3.

3.2.4 3D Object Removal

For object removal, we simply delete the 3D Gaussians of the editing target with no additional Gaussian parameter tuning.

3.2.5 3D Object Inpainting

For object inpainting, we first delete the relevant 3D Gaussians and then add a small number of new Gaussians to fill the space left by the removed Gaussians. To find the region to be inpainted we use DEVA with the prompt "big blurry hole" on the rendered images of the scene. The new Gussianns are supervised by the 2D inpainting results by LaMa[21] during rendering.

3.2.6 3D Object Addition

To overcome the limitations of purely 2D generation pipelines, we reformulate the problem as a composition and semantic placement task grounded in 3D. First, we create a 3D Gaussian representation for the objects to be inserted into the scene. Although it is possible to rely on user input for placement (e.g., selecting a 3D location), we focus on evaluating the current 3D understanding capabilities of multi-modal large language models (MLMs). Thus, we treat placement as a language-guided semantic reasoning challenge. Recent works [18] have explored related tasks using monocular images, but here we adopt a 3DLLM approach. Specifically, we start by generating a 3D point cloud from the Gaussian representations learned, as shown in Figure 8.

In this manner, our methodology shifts from 2D-only reconstruction to a 3D-aware pipeline that incorporates semantic cues provided through language. By combining explicit 3D representations and textual guidance, we aim to exploit MLMs as semantic reasoning engines that can theoretically identify suitable locations for object placement within a scene. The goal is to achieve consistency across viewpoints and arrive at a coherent, spatially informed solution that surpasses the limitations of current purely 2D methods. We further draft prompts for semantic placement similar to [18].

4. Experiments

4.1. Settings

Datasets. To measure the open vocabulary localization and segmentation accuracies in 3D scenes we use the LERF-Mask dataset proposed in [24]. LERF-Mask contains three manually annotated scenes with accurate segmentation masks. Each 3D scene contains an average of 7 text queries with corresponding GT mask labels. Additionally, we also test removal performance on 2 scenes presented in Mip-NeRF 360[14].

Implemetation Details. In training, we set $\lambda_{2d} = 1.0$ and $\lambda_{3d} = 2.0$. Also, we use the Adam optimizer with a learning rate of 2.5e-3 for identity encoding.

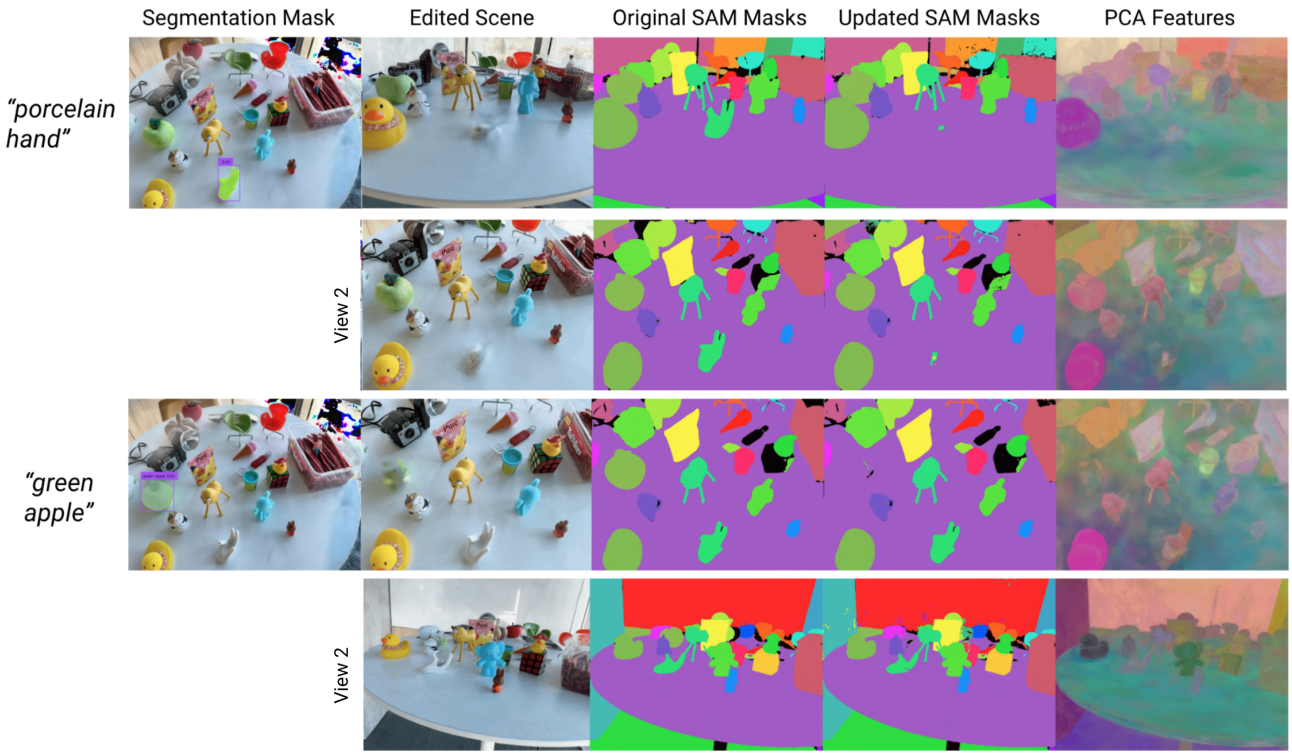


Figure 6: 3D Object Removal results on the Figurines scene from the LERF-Mask dataset using our method.

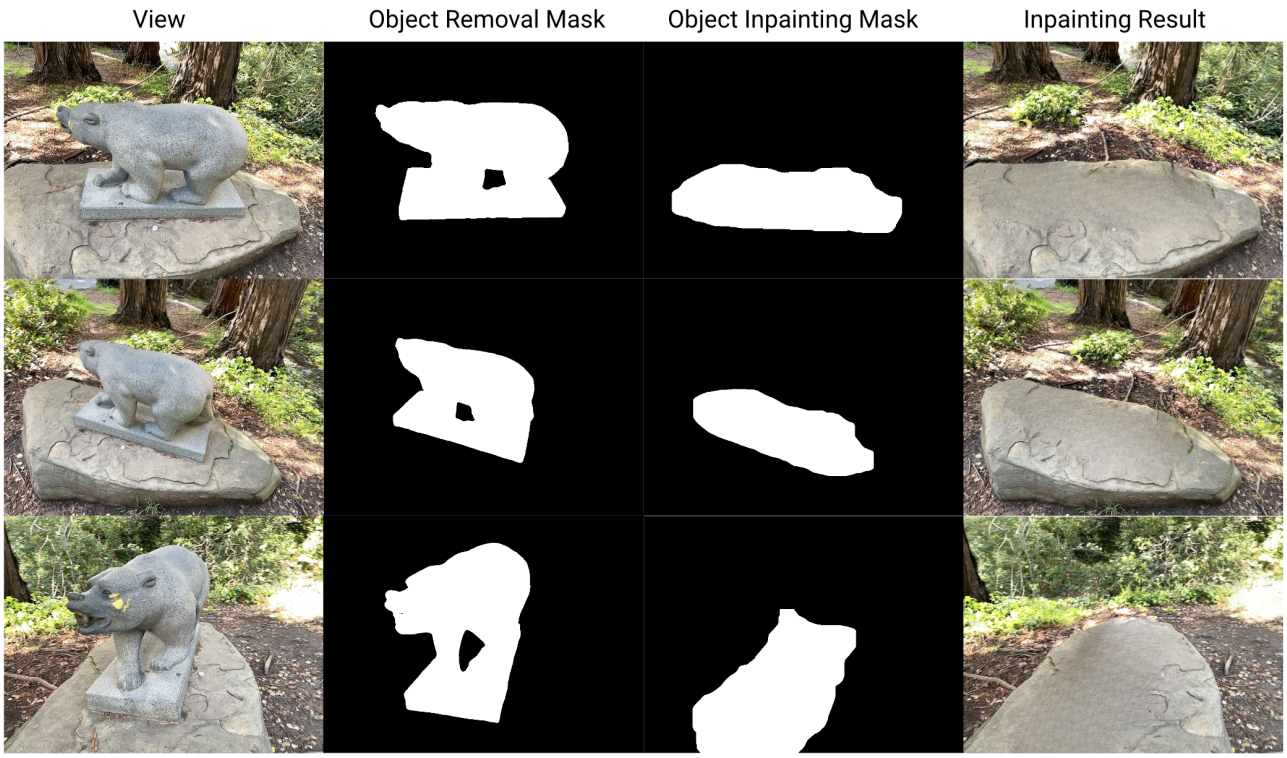


Figure 7: 3D Object Inpainting results on the Bear scene from the MipNeRF-360 dataset using our method. After object removal, the region to be inpainted is segmented using DEVA using the prompt "big blurry hole".

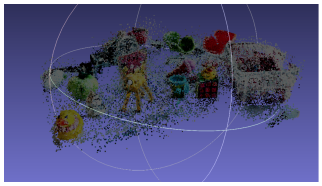


Figure 8: Sample of Gaussian splat to point cloud conversion on figurines scene

4.2. Results

Open-vocabulary Segmentation Comparison We compare the segmentation quality of our proposed methods with results of several state-of-the-art open set 3D segmentation methods, such as LangSplat[17], LERF[10] and SA3D[3] using the intersection-over-union metric. Our 3D language feature distillation works the best as shown in Table 1. We also show the per object IOUs of the same on the Figurines scene in Table 2.

Removal Quality Comparison. We use the CLIP Directional Similarity score to quantify the performance of removal methods. CLIP Directional Similarity measures the similarity of the direction of change from the original scene to the edited one between the image and text CLIP embeddings. We show in Table 3 that our method performs similarly to another language-based editing work [4] while also performing open-vocabulary localization. Some qualitative removal and inpainting results are shown in Figure 6 and 7 respectively.

Table 1: Comparison of Open Vocabulary Segmentation on LERF-Mask dataset. Our 3D Language Distillation version achieves the best performance but is limited to the categories annotated during dataset preparation with DEVA. Best result is shown in **bold** and second best is underlined.

Model	figurines		ramen		teatime	
	mIoU	mBIoU	mIoU	mBIoU	mIoU	mBIoU
LERF [10]	33.5	30.6	28.3	14.7	49.7	42.6
SA3D [3]	24.9	23.8	7.4	7.0	42.5	39.2
LangSplat [?]	52.8	50.5	50.4	44.7	69.5	65.6
Ours(2D Matching)	69.7	67.9	77.0	68.7	71.7	66.1
Ours(3D Matching)	**72.3**	**68.6**	**79.7**	**70.1**	**72.8**	**67.4**

4.3. Ablations

As an alternative to GroundedSAM [19] for object localization, we tried LangSplat [17] in an ablation study. While both are built on Segment Anything (SAM) [11], we expected the 3D language field from LangSplat to be more suitable for our use case than the 2D object detection outputs from GroundedSAM, especially considering the apparent success in object localization from the pretrained model (Figure 9). However, after training a new version of the model on a different dataset called "teatime", the results were suboptimal. While LangSplat appeared to segment the scene well, it could not localize most of the objects within the scene. The query "stuffed bear" was the only query that generated a reasonable mask. Other queries such as "plate" and "sheep" yielded mostly random masks (Figure 10). This outcome was curious considering LangSplat's claimed localization accuracy of 88.1% on the "teatime" dataset. We suspected the issue to be in the language autoencoder, but retraining the autoencoder with different learning rates and encoder/decoder dimensions generated the same results, and we were unable to replicate the paper's claimed accuracy.

Table 2: Per object IOUs of our 3D Language Distillation version on the Figurines scene from LERF-Mask Dataset.

Prompt	mIOU	bIOU
'green toy chair'	0.8707	0.8507
'green apple'	0.9155	0.8906
'old camera'	0.7336	0.6840
'porcelain hand'	0.8131	0.8113
'rubber duck with red hat'	0.6567	0.6541
'red toy chair'	0.8914	0.8626
Overall	0.6973	0.6790

Table 3: 3D Object Removal quality comparison using CLIP Directional Similarity score on scenes from MipNeRF-360.

Method	Kitchen	Bear
GaussianEditor[4]	0.1877	0.1804
Ours	0.1630	0.1415

4.4. Addition

To evaluate the proposed methodology, we test several MLMs that accept point clouds or depth information combined with camera poses as input, including [5], [25], and [23]. Each model is assessed based on its capacity to reason about object placement in 3D scenes. In our experiments, while these models demonstrate some ability to localize objects based on textual descriptions, they struggle to identify empty spaces or semantically appropriate placements. For example, when queried about placing a cushion or a blender, the models often default to identifying an existing object (e.g., "cushion on the chair," "keyboard on the desk") rather than suggesting a new, empty area. Figures 11a and 11b demonstrate typical localization and reasoning results. To further investigate semantic placement, we used a variety of prompts designed to elicit spatially ap-

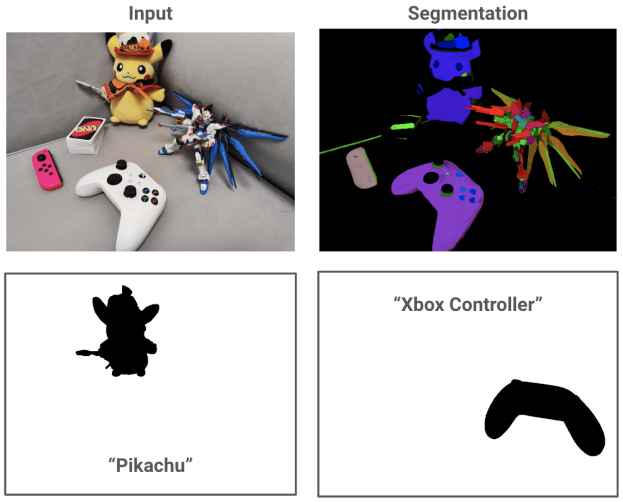


Figure 9: Results from pre-trained LangSplat model.

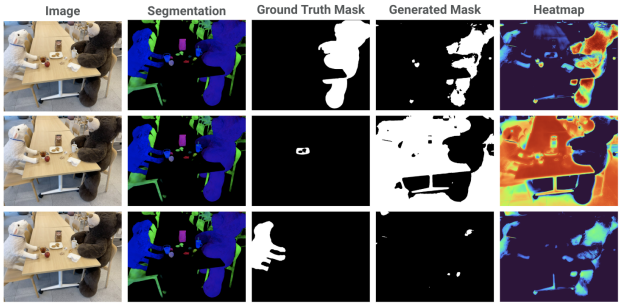


Figure 10: Results from our trained LangSplat model using different queries: "stuffed bear" (top), "plate" (middle), "sheep" (bottom).

propriate regions (see Figures 11c and 11d). While current MLMs often struggle to identify truly open spaces, their responses, such as suggesting a "chair" for cushion placement or a "desk" for a keyboard, indicate a rudimentary understanding of object-context relationships. However, they fail to localize unoccupied space. These responses suggest that their basic reasoning abilities could be improved with further instruction tuning or more advanced scene representations.

These findings highlight the challenges of semantic scene understanding and object placement in 3D. Improving these reasoning capabilities is essential for downstream applications in robotics and embodied AI, where models must interact physically and intelligently with their surroundings. While recent work [8] indicates progress toward this goal, we were unable to evaluate such systems within our current timeframe. Nevertheless, our results suggest that, with further refinement, 3DLLMs could eventually offer the spatial and semantic understanding required for effective real-world manipulation and decision-making.

5. Discussion and Conclusion

We presented ManipuLang: a system that enables open-vocabulary editing of 3D Gaussian Splats through natural language queries. Our solution integrates LLMs for task identification and object localization and uses prior works on object grouping/editing to enable users to manipulate 3D scenes through intuitive language commands. The system demonstrates effective performance in object removal tasks - achieving accuracy in both object identification and removal.

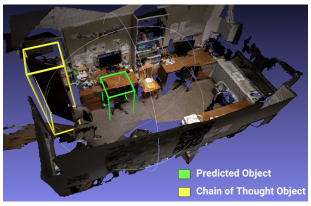
Our current implementation has several key limitations. First, the system relies on a pipeline of separate models (GPT-4o, LLaVA-3D, GroundedSAM) rather than a unified architecture. This sequential approach introduces potential bottlenecks and error propagation between stages. We attempted to develop a jointly trained model but encountered significant challenges as the compute resources that would have been required to do complex training of new join objectives that combined object grouping + CLIP embeddings were beyond what we had access to for this project. Additionally, some parts of this pipeline still don't work fully automated end-end and require human monitoring/guidance and debugging for example, when passing a query through the language pre-processor.

Also, for object addition, the system currently requires manual specification of object placement through clicks or bounding boxes and does not support finding an intuitive spot to place the object on its own.

Several promising directions exist for future work. First, developing a unified architecture could remove the need for multiple models and pipelines and potentially improve performance through end-to-end training. This would involve designing new architectures that can do both language understanding and object/grouping-based segmentation jointly fused during training instead of having these separate properties as we have now.

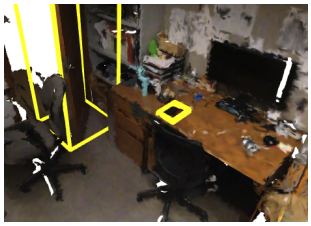
In addition, we currently use models like LLaVA-3D to get an object's name from a spatial description ("object on desk "basket") and then use a separate model trained using CLIP embeddings to find the location of the basket. This was necessary for our work as LLaVA-3D does not naively operate on Gaussian in such a way that it could produce the location of the object and a bounding box of it accurately and thus we had to use a pipeline of two different models. In the future, we would love to see this combined into a single step that works well via some model fusion or deeper integration.

Lastly, we would hope to build on the work in Seeing the Unseen [18] to make our addition pipeline capable of reasoning about where to place objects based on the scene without having to have the user specify the location - as is

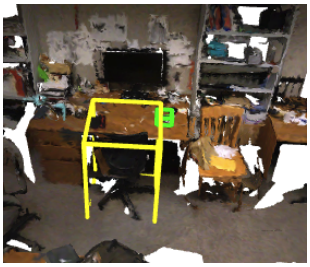


(a) Localization with prompt "chair near door".

(b) Localization with prompt "monitor near door".



(c) 3D-GRAND [23] results with prompt, perfect location to place a keyboard near door.



(d) 3D-GRAND [23] results with prompt, perfect location to place a cushion

Figure 11: Localization and semantic placement results of 3D-GRAND [23] on sample scene

currently required.

References

[1] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023. 2

[2] Chenjie Cao, Chaohui Yu, Fan Wang, Xiangyang Xue, and Yanwei Fu. Mvinpainter: Learning multi-view consistent inpainting to bridge 2d and 3d editing. *arXiv preprint arXiv:2408.08000*, 2024. 2

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

[4] Yiwen Chen, Zilong Chen, Chi Zhang, Feng Wang, Xiaofeng Yang, Yikai Wang, Zhongang Cai, Lei Yang, Huaping Liu, and Guosheng Lin. Gaussianeditor: Swift and controllable 3d editing with gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21476–21485, 2024. 2, 7

[5] Yilun Chen, Shuai Yang, Haifeng Huang, Tai Wang, Ruiyuan Lyu, Runsen Xu, Dahua Lin, and Jiangmiao Pang. Grounded 3d-llm with referent tokens. *arXiv preprint arXiv:2405.10370*, 2024. 7

[6] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Alexander Schwing, and Joon-Young Lee. Tracking anything with decoupled video segmentation. In *ICCV*, 2023. 3

[7] Hao-Yu Hsu, Zhi-Hao Lin, Albert Zhai, Hongchi Xia, and Shenlong Wang. Autovfx: Physically realistic video editing from natural language instructions. *arXiv preprint arXiv:2411.02394*, 2024. 2

[8] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023. 8

[9] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (SIGGRAPH Conference Proceedings)*, 42(4), July 2023. 1, 2, 3, 4

[10] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *International Conference on Computer Vision (ICCV)*, 2023. 7

[11] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv (Cornell University)*, 04 2023. 7

[12] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023. 3

[13] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 4

[14] Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, Peter Hedman, Ricardo Martin-Brualla, and Jonathan T. Barron. MultiNeRF: A Code Release for Mip-NeRF 360, Ref-NeRF, and RawNeRF, 2022. 5

[15] OpenAI. Gpt-4 technical report. *arXiv:2303.08774 [cs]*, 03 2023. 1

[16] OpenAI. Introducing gpts — openai, November 2023. 3

[17] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting, 2023. 1, 2, 7

[18] Ram Ramrakhya, Aniruddha Kembhavi, Dhruv Batra, Zsolt Kira, Kuo-Hao Zeng, and Luca Weihs. Seeing the unseen: Visual common sense for semantic placement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16273–16283, 2024. 5, 8

[19] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 01 2024. 1, 4, 7

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

[21] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with fourier convolutions. *arXiv preprint arXiv:2109.07161*, 2021. 5

[22] Alexander Vilesov, Pradyumna Chari, and Achuta Kadambi. Cg3d: Compositional generation for text-to-3d via gaussian splatting. *arXiv preprint arXiv:2311.17907*, 2023. 2

[23] Jianing Yang, Xuweiyi Chen, Nikhil Madaan, Madhavan Iyengar, Shengyi Qian, David F Fouhey, and Joyce Chai. 3d-grand: A million-scale dataset for 3d-llms with better grounding and less hallucination. *arXiv preprint arXiv:2406.05132*, 2024. 7, 9

[24] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *ECCV*, 2024. 1, 2, 3, 4, 5

[25] Chenming Zhu, Tai Wang, Wenwei Zhang, Jiangmiao Pang, and Xihui Liu. Llava-3d: A simple yet effective pathway to empowering lmms with 3d-awareness. *arXiv preprint arXiv:2409.18125*, 2024. 1, 3, 7

Student Name	Contributed Aspects	Details
Sanchit Tanwar	Implementation and Experimentation	Implemented a standalone inference and evaluation script for Langsplat along with Gaussian Grouping. Ran experiments for object addition with different MLM's.
Pushkar Raj Singh	Implementation and Analysis	(Implementation of GroundedSAM based object localization- 2D Mask Matching and 3D Distillation and subsequent integration to removal and integration pipelines.)
Milad Heydari	Implementation and Experimentation	GPT LLM Query Processor and LLaVA 3D object localization and pipeline. Initial LangSplat experimentation
Pranav Datta	Experimentation and Analysis	Trained and experimented different LangSplat models and analyzed results. Attempted to implement removal into LangSplat.

Table 4: Contributions of team members.