Language-Guided Video Rendering in 3D Gaussian Splatting using Sparse Spatial-Semantic Attributes

Kiran Lekkala
Dolby Laboratories
kiran.lekkala@dolby.com

Tsung-Wei Huang
Dolby Laboratories
tsung-wei.huang@dolby.com

Guan-Ming Su
Dolby Laboratories
guanmingsu@ieee.org

***Abstract*— This paper presents a novel framework for generating realistic video footage at novel viewpoints from 3D Gaussian splats using text prompts. Current approaches face two key challenges: incorporating high-dimensional semantic information into dense Gaussian representations without costly annotations and efficiently passing representations to a language model to generate smooth, aesthetically pleasing camera trajectories. To address these, we propose a two-component pipeline. First, our Sparse Spatial-Semantic (SSS) attribute Augmentor (SAug) enriches Gaussian splats with spatial-semantic features learned from multi-view images alone, along with our sparsity control technique maintaining compact representations despite millions of splats. Second, our LLM-based Motion Generator (L-MR) leverages Large Language Models to produce smooth 6D camera pose trajectories directly from text prompts and compact object-based representations using SAug outputs. Together, these components enable text-driven generation of realistic novel-view videos, with SAug providing semantically-aware scene representations and L-MR translating natural language into cinematic camera motions. Experiments demonstrate that our method generates semantically and spatially more accurate trajectories while keeping the memory and computation footprint of the rendering pipeline low.**

Keywords—3DGS, LLM, Text-to-Motion, Sparse representation

I. Introduction

Recent advancements in 3D scene representation and rendering have led to the emergence of 3D Gaussian Splatting (3DGS) as a powerful technique for creating photorealistic digital environments [1]. This method uses millions of tiny, translucent ellipsoids called Gaussian splats to represent complex scenes with remarkable fidelity and efficiency. While 3DGS has shown great promise in capturing visual details from multi-view images, it currently lacks the ability to encode semantic information and spatial attributes beyond what is necessary for realistic rendering.

Our research addresses this limitation by developing a novel framework that enhances Gaussian splats with learned spatial and semantic attributes. This innovation is crucial for expanding the utility of 3DGS beyond mere visual reproduction, enabling more intelligent and context-aware applications in computer graphics and artificial intelligence. The core of our approach involves developing a method to incorporate spatial and semantic information into individual Gaussian splats, enriching their representation beyond visual characteristics. Given the high number of splats in a typical scene (often in the millions), we propose a procedure for sparsity control that learns a compact set of spatial and semantic attributes. This ensures efficiency by maintaining essential information needed for downstream tasks. We leverage a Large Language Model (LLMs) and Sequence models to interpret these enriched splats and generate sequences of keyframe poses, and thereby generating smooth and valid trajectory of 6D poses. This allows for text-prompted generation of realistic video footage rendered from explicit Gaussian splat representation. Our system incorporates a custom LLM-based approach to generate trajectories of poses. This integration bridges the gap between high-level semantic understanding and low-level path planning, enabling more natural and context-aware camera movements.

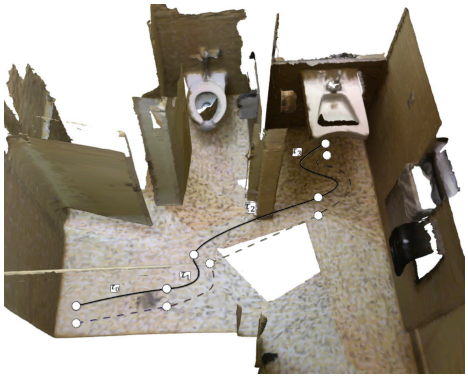

Figure 1. Visual representation of the ground-truth (solid line) and our model's predicted trajectory (dashed line). τ_i represents the interpolated ground-truth trajectory between two subsequent keyframes for the following text prompt: "*Start with a close up shot of the door<OBJ-24> showcasing arrangement of objects <OBJ-2>, <OBJ-11> and <OBJ-33>. Capture panoramic view from the door <OBJ-24> moving towards the mirror <OBJ-13> on the wall, capturing the dustbin<OBJ-23>. Focus on sink<OBJ-39> and finally pan towards the tap<OBJ-26>.*"

By combining these elements, our research aims to transform 3D Gaussian Splatting from a purely visual representation technique into a rich, spatial and semantic-aware framework capable of supporting advanced AI-driven content

979-8-3315-7111-5/26/$31.00 ©2026 IEEE

creation and scene understanding tasks. This work has significant implications for various fields, including virtual cinematography, autonomous navigation, and interactive digital environments. In the following sections, we detail the preliminaries, methodology, implementation details, and discuss the potential applications and future directions of this enhanced 3D Gaussian Splatting framework.

II. Background

As detailed in the comprehensive survey [4], **3D Gaussian Splatting (3DGS)** has evolved beyond its initial rendering capabilities to support more complex scene representations as a powerful method for representing complex 3D scenes. To this end, recent methodologies including LangSplat [13] and the decoupled feature embedding techniques presented in [12] have integrated semantic features directly into the Gaussian primitives. These advancements are pivotal, enabling real-time 3D scene understanding with rich contextual information. Recently chat applications like SplatTalk [11] that enables users to interact with 3DGS have also been proposed. However, retaining high fidelity of 3DGS while keeping the memory footprint low for downstream tasks is yet to be explored.

Multi-modal **Large Language Models** and 3D Understanding Large Language Models (LLMs) have made significant strides in natural language processing and understanding [7,8]. Recent research has explored the application of LLMs to 3D data [5], demonstrating their potential for processing and interpreting spatial information alongside textual inputs. Concurrently, research in point cloud understanding using LLMs has made significant strides [3] by generating context-aware responses to user queries. Recent LLMs [2] understanding 3D world further bridge this gap by demonstrating the ability to process object identifiers as a token placeholder for instance-level point clouds, generating responses to a specific instance of a semantic class. These advancements suggest the potential for LLMs to comprehend and manipulate 3D spatial data effectively.

Following are our novel contributions that we propose:

- We propose a method, Sparse Spatial-Semantic attribute Augmentor (SAug) to learn and incorporate spatial-semantic attributes directly into Gaussian splats, without any additional information apart from the dataset of multi-view images used for training 3DGS. This supports fine-grained spatial-semantic understanding, crucial for language-guided rendering.

- Since 3DGS are inherently heavy with a large memory footprint, we introduce a novel technique for adaptively controlling sparsity depending on the spatial complexity of an object. This process is coupled with spatial-semantic attributes for Gaussian splats, significantly reducing memory requirements and other computational challenges.

III. Overall Framework

The initial input to our system is a sequence of N multi-view RGB images denoted by $D = \{I_i \in \mathbb{R}^{(3,H,W)} \mid i = 0, \dots, N-1\}$, where i, H, W is the index, height and width of the image frame. We first train a Gaussian splat G using a training procedure $\hat{Q}$ that uses all the images in D. Mathematically $G = \hat{Q}(D)$. Although G is used for rendering images, we obtain Sparse Spatial-Semantic (SSS) attributes S that are used for the motion generation process. SSS augmented 3DGS enables systems to understand the relationship between spatial and semantic attributes of the scene. We obtain $S = \hat{A}(D)$, where $\hat{A}$ is the SSS attributes Augmentation (SAug) procedure. During deployment, user can provide a text prompt T to obtain a trajectory of 6D poses P using $\hat{L}$ using $P = \hat{L}(S, T)$, where $\hat{L}$ is LLM-based Motion Generator (L-MR). This stream or trajectory of poses P is passed to the 3DGS rendering procedure $\hat{R}$ to obtain a video V, i.e. $V = \hat{R}(G, P, \bar{C})$, where $\bar{C}$ is the camera parameters relating to the rendered video. Figure 2 shows the high-level flow chart of the system, through an easy-to-understand block diagram.

A. Training 3D Gaussian Splats ($\hat{Q}$)

3DGS uses a 3D sparse point cloud initialization from COLMAP [9] and iteratively learns the parameterized splats $\theta_l = \{p_l, s_l, r_l, c_l, \gamma_l\}$, representing position, scale, rotation, color, and opacity respectively. The parameters θ_l are obtained by optimizing the following loss-function:

$$\mathcal{L}_{3DGS}(\boldsymbol{\theta}) = \sum_{i=0}^{N-1} \| R(G(\boldsymbol{\theta}), C_i) - I_i \|$$

In the above loss-function, R corresponds to a differentiable rasterization function, $G(\boldsymbol{\theta})$ represents the scene described by the Gaussian splats, I_i and C_i corresponds to the image and the camera parameters, respectively. C_i comprises both the intrinsics and the extrinsics of the viewpoint used to capture the image. Note that in practice, the loss function $\mathcal{L}_{3DGS}$ also involves an additional term comprising the Structural Similarity Index Measure (SSIM). Since a dense parameterization of Gaussian splats $G(\boldsymbol{\theta})$ is only required for realistic rendering, and not for understanding the spatial-semantic relationships in the 3D scene, we obtain *Point splats* $\bar{G}(\bar{\boldsymbol{\theta}})$ from $G(\boldsymbol{\theta})$, where $\bar{\theta}_l = \{p_l\}$, to preserve only the position information of Gaussian splats. For simplification purposes, we refer to Point splats as $\bar{G}$ and use as input to the Semantic Feature extractor (SF) module.

B. Semantic Feature extractor (SF)

Inspired by the work of [2] and [3], we implement a 3D feature extraction pipeline to obtain semantic features for 3DGS. We first obtain all semantic-level instances using a large-pretrained instance segmentation model H as follows: $H(\bar{G}) = \{\bar{G}^0, \bar{G}^1, \bar{G}^{M-2}, .., \bar{G}^{M-1}\}$, where M corresponds to max-number of semantic instances. All semantic instances are then projected to multiple views present in the 3D scene, to obtain projected

binary masks $B_i^j = \{B_i^j \in \{0,1\}^{(H,W)} \,|\, i = 0, \ldots, N-1;\ j = 0, \ldots, M-1\}$, where M corresponds to max-number of semantic instances and N corresponds to the total number RGB images present in the train-dataset that is used to obtain the 3DGS.

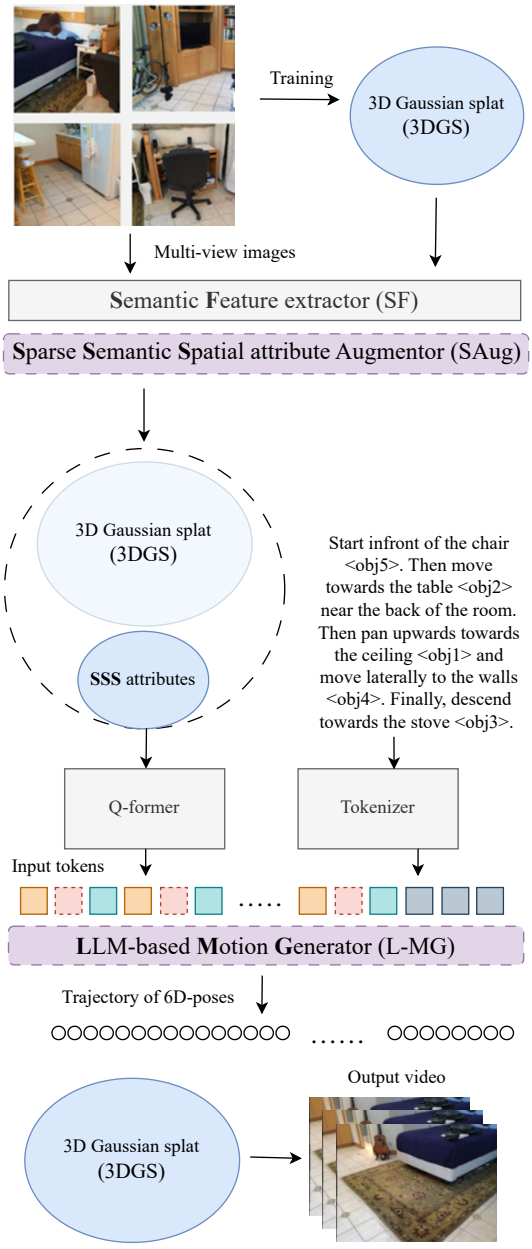

Figure 2. High-level overview of the proposed system. Our main proposed contributions are further described in detail in Figures. 3. <obj-*> are object-identifiers that prevents ambiguity in referring to which specific semantic instance.

As stated before, B_i^j is obtained by projecting all object-level point splats $\bar{G}^j$ associated with object j onto image I_i having camera parameters C_i, consisting of both intrinsics and extrinsics, using the projection function $\pi(\bar{G}^j, C_i)$.

We then extract dense 2D features f_v using a pre-trained foundation model (DinoV2) [6] from cut-outs of the original RGB images using these projected masks. All the points represented by x, y and z pertaining to a specific 3D instance that are projected to an image are now associated to a semantic feature f_v.

Although referring objects by their semantics (chair, table, ceiling, wall etc.) seems trivial, in many real-world situations, there exist many variations and instances of each of these semantics. Scenes could certainly get complex when many such instances are present in complex configurations. To avoid object ambiguous object references, one straight-forward way is to have object IDs for the instances that automatically show up in the output of the instance segmentor H. These object-IDs, also known as object-identifiers, could be textually shown as <obj-0>, <obj-1>, <obj-2>, …, <obj-99> etc. are essentially special tokens that are added in the vocabulary and are used to refer the corresponding semantic instance in the text prompt. This process makes it easy for the LLM to understand which specific semantic instance is being referred. Since the object-IDs need to exist in the vocabulary to make sense for the LLM, we add these into the vocabulary and allocate a learnable embedding that is obtained during the end-to-end training.

C. *Sparse Spatial-Semantic attributes Augmenter (SAug) with Q-former*

Using the object-level 3D semantic features f_v obtained in the previous section, our proposed SAug module selectively prunes out redundant features attributed to 3D points, without sacrificing on the shape of the object. We utilize the concepts from contour estimation by computing a special subset that approximates the original set of points and is known as the convex hull. More details on this module can be found in Section IV and Figure 3.

We finally obtain *spatial and semantic features* $S^j = \{x, y, z, f_v\}$ from the SSS attributes from the SAug module. Before passing *spatial and semantic features* S^j, for each object, into a Q-former [14] we apply positional encoding to the feature vectors, that enables the network to understand the Spatial-Semantic relationship pertaining to a specific object. We use a similar positional-encoding procedure used by [3] that involves splitting the semantic feature f_v into three parts and apply positional encoding to each of the three parts using x, y and z locations of a specific splat. But instead of computing the absolute x, y and z values, we compute relative to the centroid of a specific object. We concatenate corresponding positional encodings PE_x, PE_y, PE_z, and pass it through the Q-former to infer a concise set of tokens *for every object.* These are all concatenated before getting passed onto the LLM, along with the tokenized language and *object-identifiers*, which would be described in the next sub-sections.

IV. Sparse Spatial-Semantic attributes augmenter (SAug)

SAug is designed to sparsify a large set of attributes present in 3DGS, while maintaining essential geometric information. This module addresses the key-challenge of which splats to have spatial-semantic attributes when processing large 3DGS into external systems, like our LLM based Motion Generator.

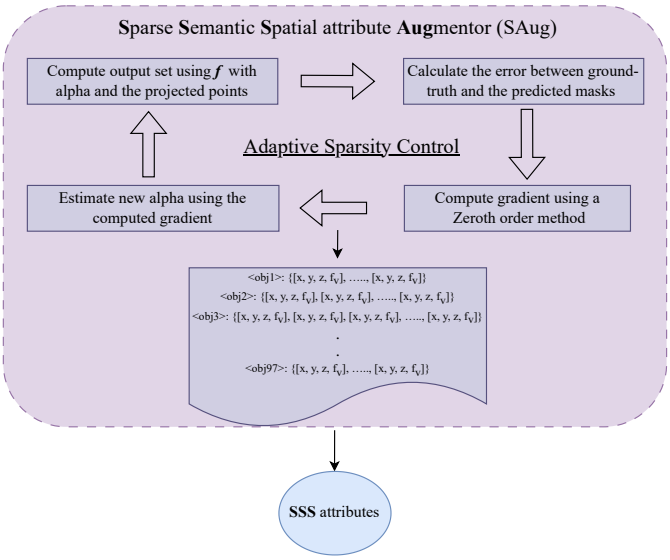

Figure 3. Internals of the SAug module. The obtained Sparse Spatial-Semantic (SSS) attributes from the SAug module is associated with a corresponding 3DGS.

While detailed Gaussian splats provide rich information about scene geometry and texture, the task of motion trajectory generation primarily relies on understanding the overall shape and spatial arrangement of objects. Texture information, while crucial for rendering, is less critical for path planning. By selectively reducing the density of the associated SSS attributes for a 3DGS, we can significantly decrease the computational load on the LLM based Motion Generator, without compromising its ability to generate appropriate motion trajectories. To reduce the number of additional attributes while preserving the essential shape information, our Adaptive Sparsity Control module employs a novel convex hull based approach using 2D-3D correspondences of objects in the 3D scene. Our approach adjusts the sparsity level by retaining more detail for objects having complex shapes. The simplified point cloud emphasizes structural information, potentially making it easier for the both the perceiver and the LLM to understand and reason about spatial relationships.

A. Computing α-shape for 2D-projections

As stated before, the input to our novel SAug module would consist of object level semantic features and their associated discretized 3D voxels taking discrete values from 0 to 255. Along with these, every voxel would also consist of a semantic feature vector f_v.

The procedure to evaluate α-shape, can be represented using the equation $\omega = \boldsymbol{f}(\sigma, \alpha)$; where σ is a set of input 2D points and α determines compactness of the set of output points ω to contain all points in σ. Note that for simplicity purposes, ω is assumed to be an ordered set of points listed pairwise based on the prevalent edges between the pair of points. Specific to our case, all the points σ are two-dimensional consisting of (x, y). These are obtained by applying the projection function π on a set of 3D points $\bar{G}^j$ using $\pi(\bar{G}^j, C_i)$ with camera parameters of a specific view-point C_i that has a corresponding image I_i. Lastly, j corresponds to a semantic class that exists in image I_i.

B. Adaptive Sparsity control

We use α-shape [8] for controlling sparsity of the points. Since α-shape is not differentiable, we can utilize *Zeroth-order methods* for optimizing the value of α. We find the α, such that it minimizes the following cost function:

$$\mathcal{J}(\alpha_j) = \sum_{i=0}^{N-1} \| \boldsymbol{f}(\pi(\bar{G}^j, C_i), \alpha_j) - B_i^j \|^2$$

Essentially, the above cost function is a measure of how similar the projected masks from the *"sparsified"* object-level point clouds are from the original ground-truth binary masks B_i^j where i and j are the indices of images and the semantic instance, respectively. Note that the above $L2$ or Euclidian distance can be replaced with a binary cross-entropy loss since we are dealing with binary masks. As stated before, $\boldsymbol{f}(\sigma, \alpha)$ computes the α-shape of the points σ using a specific α.

In the iterative procedure of optimizing α using the cost function $\mathcal{J}(\alpha)$, the initial α can be set close to 0 at the start. Note that the number of points, i.e. total sum of sizes of all resultant sets ω_i^j obtained from $\boldsymbol{f}(\pi(\bar{G}^j, C_i), \alpha)$, increases with α. Since we are attempting to prune points directly in 3D space, there exists an alpha for each object-level semantic instance $\bar{G}^j$. Furthermore, all the semantic classes are independent, and we can independently optimize the combined set of α's, $\{\alpha_j\}$, each corresponding to a semantic instance j. This makes our approach highly parallelizable and efficient.

V. Experimental results

We trained the L-MG system and the Q-former end-to-end on the train-set of our curated dataset that we built upon Scannet [5]. We use GPT-4o [8] to first generate the dataset of key-frame poses. Following the text-command and the corresponding 3D scene our system should output a trajectory that can then be rendered into a video from the 3DGS.

Evaluation on the test-set. We set aside 10% of 5700 samples randomly selected data as the test set. We show the experimental results on the test set.

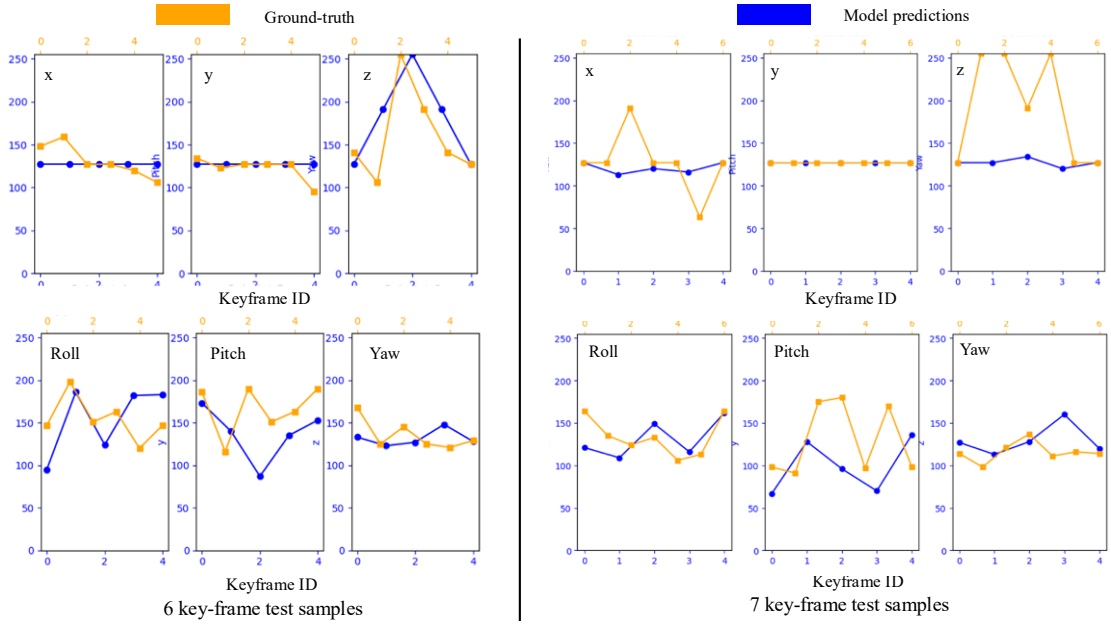

Figure 4. Comparison of the model predictions and corresponding ground-truth of 2 different test sets. The left and the right test sets contain 6 key-frames and 7 key-frames respectively and constitute some of the challenging test samples in the dataset. Each plot corresponds to the *x, y, z, roll, pitch, yaw* of each of the keyframe pose in the trajectory.

Evaluating the performance of a model in terms of prediction accuracy is challenging especially due to the number of poses being subjective. Hence we plot each of the key-frame outputs for each pose element as shown in Figure 4 as estimates for empirical evaluation. The test set was sampled randomly.

Our LLM inside our L-MG system was initialized using Vicuna [6] model weights. We trained on 5130 data samples in our curated dataset on an NVIDIA A100 GPU. Following are the hyperparameters used in our experiments:

Hyperparameter	Value
Number of epochs	3
Base learning rate	5e-6
Batch size	32
Lora R	16
Lora alpha	16
Max grad norm	0.01

VI. Conclusion

This research presents a novel approach to enhancing Gaussian splats with spatial and semantic attributes, significantly advancing the field of 3D scene representation and generation. Our work addresses a critical gap in the current state of Gaussian splats, which traditionally only contain information necessary for realistic rendering. By incorporating spatial and semantic relationships into individual Gaussian splats, we have created a more comprehensive and versatile representation of 3D scenes. This enhanced representation serves as a powerful foundation for downstream tasks, particularly in the domain of text-guided video generation. The proposed system opens up new possibilities in fields such as virtual cinematography, augmented reality and robotics.

References

[1] Kerbl, Bernhard, et al., "3D Gaussian Splatting for Real-Time Radiance Field Rendering." ACM Trans. Graph. 42.4 (2023): 139-1.

[2] Huang, Haifeng et al., "Chat-scene: Bridging 3d scene and large language models with object identifiers." Proceedings of Neural Information Processing Systems, December, 2024.

[3] Hong, Yining, et al., "3d-llm: Injecting the 3d world into large language models." Advances in NeurIPS (2023): 20482-20494.

[4] Kathariya et al., Gaussian Splatting: State-of-The-Arts and Future Trends, ICNC (2025)

[5] Dai, Angela, et al., "Scannet: Richly-annotated 3d reconstructions of indoor scenes." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[6] Oquab, Maxime, et al., "Dinov2: Learning robust visual features without supervision." arXiv preprint arXiv:2304.07193 (2023).

[7] Chiang, Wei-Lin, et al., "Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality." https://vicuna.lmsys.org (2023):

[8] Hurst, Aaron., et al., "Gpt-4o system card." arXiv preprint arXiv:2410.21276 (2024).

[9] Gardiner et al., "Alpha shapes: determining 3D shape complexity across morphologically diverse structures", BMC Evolutionary Biology 2018

[10] Schonberger et al., "Structure-from-Motion Revisited" Conference on Computer Vision and Pattern Recognition (CVPR) (2016).

[11] Thai et al., "SplatTalk: 3D VQA with Gaussian Splatting", (ICCV) (2024)

[12] Dai et al., "Efficient Decoupled Feature 3D Gaussian Splatting via Hierarchical Compression" Conference on Computer Vision and Pattern Recognition (CVPR) (2025).

[13] Qin et al., "LangSplat: 3D Language Gaussian Splatting" Conference on Computer Vision and Pattern Recognition (CVPR) (2024)

[14] Li et al., "BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models" (CVPR) (2024)