UVGS: Reimagining Unstructured 3D Gaussian Splatting using UV Mapping

Aashish Rai[1,2] Dilin Wang[2] Mihir Jain[2] Nikolaos Sarafianos[2] Kefan Chen[1,2]
Srinath Sridhar[1] Aayush Prakash[2]

[1]Brown University [2]Meta Reality Labs

https://ivl.cs.brown.edu/uvgs

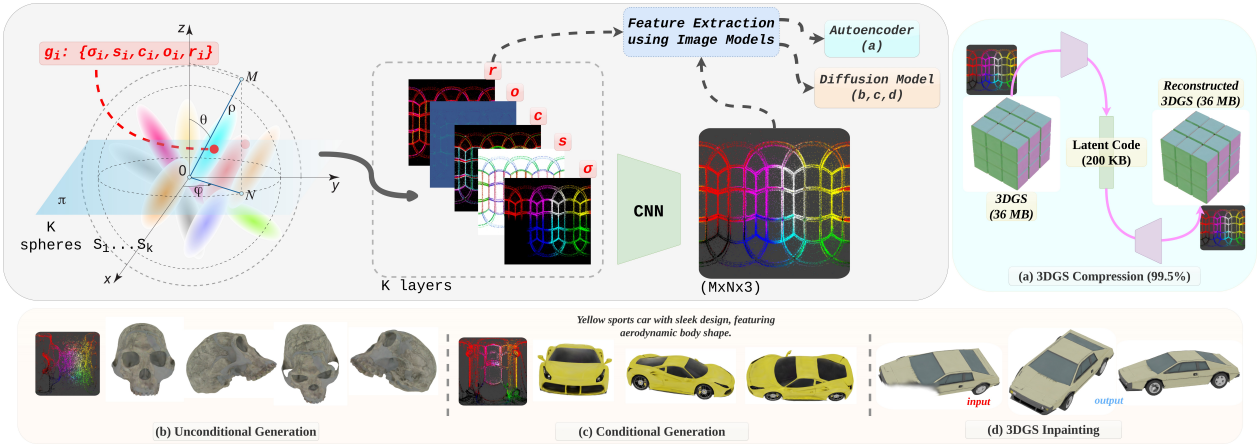

Figure 1. *We propose UVGS - an structured image-like representation for 3DGS obtained by spherical mapping of its primitives. The obtained UVGS maps can be further unified to a 3-channel Super UVGS image to bridging the gap between 3DGS and existing image foundation models. We show Super UVGS can compress the 3DGS assets using pretrained image Autoencoders, and for direct unconditional and conditional 3DGS object generation using diffusion models. We also present one of the first experiments on 3DGS inpainting.*

Abstract

3D Gaussian Splatting (3DGS) has demonstrated superior quality in modeling 3D objects and scenes. However, generating 3DGS remains challenging due to their discrete, unstructured, and permutation-invariant nature. In this work, we present a simple yet effective method to overcome these challenges. We utilize spherical mapping to transform 3DGS into a structured 2D representation, termed UVGS. UVGS can be viewed as multi-channel images, with feature dimensions as a concatenation of Gaussian attributes such as position, scale, color, opacity, and rotation. We further find that these heterogeneous features can be compressed into a lower-dimensional (e.g., 3-channel) shared feature space using a carefully designed multi-branch network. The compressed UVGS can be treated as typical RGB images. Remarkably, we discover that typical VAEs trained with latent diffusion models can directly generalize to this new representation without additional training. Our novel representation makes it effortless to leverage foundational 2D models, such as diffusion models, to directly model 3DGS. Additionally, one can simply increase the 2D UV resolution to accommodate more Gaussians, making UVGS a scalable solution compared to typical 3D backbones. This approach immediately unlocks various novel generation applications of 3DGS by inherently utilizing the already developed superior 2D generation capabilities. In our experiments, we demonstrate various unconditional, conditional generation, and inpainting applications of 3DGS based on diffusion models, which were previously non-trivial.

1. Introduction

The creation of high-quality 3D content is essential in applications like virtual reality, game design, robotics, and movie production, where realistic 3D representations play a critical role. Typical 3D representations like Neural Radiance Fields (NeRF) [29] are promising but require substantial computational resources, limiting their scalability for real-time applications. Moreover, NeRF is an implicit representation, which makes editing and manipulation challenging. Recently, 3D Gaussian Splatting (3DGS) [19] emerged as a compelling alternative, enabling efficient and high-fidelity 3D rendering through a large set of Gaussian primitives that model spatial and visual properties. As an explicit represen-

tation, 3DGS offers several advantages over NeRF. However, while 3DGS offers benefits in terms of speed and visual quality, its unstructured, permutation-invariant nature presents significant challenges for generative tasks. Much like point clouds, it lacks a coherent spatial structure, impeding its integration with conventional image-based generative models. This lack of structure and coherence among primitives hinders the application of image-based generative models [27, 61, 67], which rely on structured data representations.

Previous methods have tackled these challenges by transforming 3DGS into structured formats, such as voxel grids [9, 16, 61] or image-based representations like Splatter Image [46] or triplanes [70]. Other approaches employ diffusion models to directly predict 3DGS attributes [32]. These methods, while achieving impressive visual results, often require substantial computational resources, memory-intensive multi-view rendering, complex architectures limiting their scalability and flexibility for high-fidelity generation. Generating and processing 3DGS directly by efficiently utilizing modern generative models like Variational Autoencoders (VAEs) and diffusion models is limited as the neural networks are not permutation invariant.

To address these shortcomings, we introduce **UV Gaussian Splatting (UVGS)**, which provides a structured transformation of 3D Gaussian primitives into a 2D representation while preserving essential 3D information. We use spherical mapping [43] that inscribes Gaussian splats in a spherical surface, and projects attributes like position, rotation, scale, opacity, and color into an organized 14-channel image-like UV map. This mapping introduces spatial structure, resolving issues of permutation invariance by introducing local correspondences between neighboring Gaussians and global coherence across the entire 3D object. The result is a representation that functions as a "3D representation" structured in a 2D map format, enabling compatibility with powerful image-based neural network architectures.

While UVGS introduces structure into 3D Gaussian Splatting, its full 14-channel attribute-specific representation presents challenges for direct integration with pretrained 2D generative models, as these models typically expect a simpler, image-compatible data. Each of its heterogeneous attributes—position, color, and transformation—has its own distinct distribution and resides in a separate feature space, making it challenging to represent the 3D object in a unified shared space. To address this, we introduce **Super UVGS**, a compact 3-channel representation that unifies these diverse attributes into a cohesive format. Using a carefully designed multi-branch mapping network, Super UVGS consolidates the distinct attribute spaces into a shared feature space, enabling a more collective representation of the object. This unified transformation not only facilitates zero-shot compatibility with pretrained 2D models but also optimizes memory usage and computational efficiency, making Super UVGS highly practical for large-scale 3D tasks. Unlike previous approaches that use Triplanes, voxels, occupancy grid, neural fields etc. and require specialized 3D architectures to train on 3D data, UVGS effortlessly leverages widely available pretrained 2D foundational models. This zero-shot generalization capability allows UVGS to fully benefit from priors learned in 2D domains from large amount of data, improving both flexibility and scalability. To sum up our main contributions are:

- *Efficient Structured Representation of 3DGS*: We present UVGS, an image-like representation that solves permutation invariance and unstructured nature of discrete 3DGS through spherical mapping, making direct feature extraction possible by organizing unordered points into a coherent 2D representation compatible with 2D models.
- *Compact and Scalable Super UVGS Representation*: To address scalability while dealing with large scale 3DGS points and enabling the direct integration of pre-trained 2D foundation models, we introduce Super UVGS - a low-dimensional version of UVGS maps that retains high fidelity features while reducing memory overhead.
- *Diverse 3D Applications*: Our approach unlocks seamless integration of 3DGS with pre-trained 2D foundation models for various tasks, including unconditional and conditional generation of 3DGS.

2. Related Work

3D Generative/Reconstruction Models for Objects: Generation or reconstruction of 3D assets has been a long standing task [4, 24, 28, 30, 31, 35, 39, 40, 45, 47, 49, 56, 60]. Previous reconstruction approaches like NeRF [2, 29] are often slow and do not provide a defining geometry [6, 13, 18, 23, 26, 28, 36, 40, 54, 66, 68]. Advancements in the field led to the emergence of explicit voxel grid based representations that encode colors and opacities directly [9, 33, 51]. These approaches achieve significant speed ups compared to the NeRF based approaches, but they can't produce high fidelity assets due to the low resolution of voxel grids. On the other hand, triplane representation [15, 44, 53, 69] provides a trade-off between the quality and memory utilization. Another line of work [11, 59] splits the input mesh into different patches and simplifies the object generation problem to an image generation problem. However such methods rely on either cutting through the mesh to create a geometry image [11] or rely on an existing UV representation of the geometry and utilize subset of existing UV islands [59] resulting in loss of details. Recently, there has been a notable advancement in 3D Gaussian Splatting (3DGS) for the representation of objects and scenes leding to the emergence of 3DGS showcasing impressive real-time results in reconstruction and generation tasks [16, 19, 22, 34, 48, 55, 57, 58, 60, 62, 69]. Recent

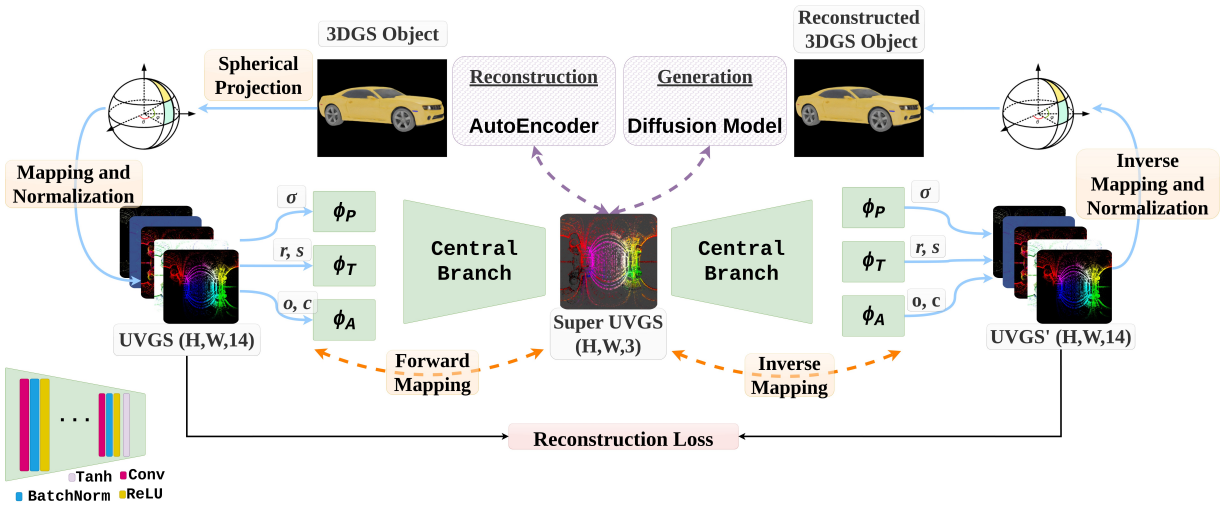

Figure 2. The input 3DGS object is first converted to UVGS maps through spherical mapping. We use a multibranch forward mapping network to convert the obtained 14-channel UVGS to a compact 3-channel Super UVGS image. This represents the 3DGS object in a structured manner and can be used with image foundation models for reconstruction or generation. The Super UVGS is mapped back to UVGS through branched inverse mapping, which in turn can be reconstructed back to the 3DGS object through inverse spherical mapping.

advances in the 3D generative models for asset synthesis using the existing geometries like NeRF, voxel grids, or triplane geometries [3, 14, 48, 58, 62, 69] leverage generative models [17, 41] and the existing 3D datasets [5, 10]. However, most of the works employing 3DGS or other representations use multiview rendering and Score Distillation Sampling (SDS) to achieve convincing generation and reconstruction capabilities [8, 35]. These approaches demand high memory and compute resources and are often quite slow in optimizing due to per scene optimization.

Giving Structures to Discrete Gaussians: Although, 3DGS has led to breakthrough in the reconstruction field by demonstrating superior performance in multiple domains, the generation of 3DGS directly remains challenging due to its discreteness and unstructured nature [61, 67]. These characteristics present substantial challenges when integrating them with conventional computer vision models, like Autoencoders and generative models [67]. The research in direct learning of trained 3DGS primitives is largely unexplored [27]. Some efforts attempt to address this by directly predicting 3DGS attributes using diffusion models [32] while others like Splatter Image [46] project Gaussian objects into image-based representations through direct 3D-unaware projection. These methods struggle with maintaining multiview consistency, as the model only infers seen poses correctly, while hallucinating for unseen poses.

Concurrent works [16, 61] follow the voxel-based representations to transport Gaussians into structural voxel grids with volume generation models for generating Gaussians. However, these methods are computationally expensive for high-resolution voxels, face difficulties in preserving high-quality Gaussian reconstructions due to information loss during voxelization. DiffGS [67] tries to solve the above issues by proposing three continuous functions to represent 3DGS. However, it is limited to only category-level generation and learning generic probability functions for all the categories poses significant compute and design challenges. In contrary, we introduce an efficient way to give structures to discrete Gaussians by taking inspiration from the developments in 3D graphics. Our method does not require any learning to map an unstructured set of Gaussians to this efficient and structured representation (termed UVGS). The proposed representation provides local and global correspondence among different Gaussian points making the widely available existing computer vision frameworks learn and extract underlying features from them.

3. Methodology

Preliminaries: 3DGS represents an object or a scene with a collection of Gaussians primitives to model the geometry and view-dependent appearance. For a 3DGS set, $G = \{g_i\}_{i=1}^{N}$, representing an object with N individual Gaussians, the geometry of the i^{th} Gaussian is explicitly parameterized via 3D covariance matrix Σ_i and it's center $\sigma_i \in \mathbb{R}^3$ as: $g_i(x) = e^{(-\frac{1}{2}(x-\sigma_i)^T\Sigma^{-1}(x-\sigma_i))}$ where, the covariance matrix $\Sigma_i = r_i s_i s_i^T r_i^T$ is factorized into a rotation matrix $r_i \in \mathbb{R}^4$ and a scale matrix $s_i \in \mathbb{R}^3$. The appearance of the $i-th$ Gaussian is represented by a color value $c_i \in \mathbb{R}^3$ and an opacity value $o_i \in R$. In practice, the color is represented by a series of Spherical Harmonics (SH) coefficients, but for simplicity, we represent the view-independent color by just RGB values. Thus, a single Gaussian can be represented by a set of five attributes as $g_i = \{\sigma_i,\ r_i,\ s_i,\ o_i,\ c_i\} \in \mathbb{R}^{14}$, and the entire 3DGS can be represented by a set of N such Gaussians as: $G = \{\{\sigma_i,\ r_i,\ s_i,\ o_i,\ c_i\}\}_{i=1}^{N}$.

3.1. Spherical Mapping

3DGS is represented as a permutation invariant set with no structural correspondence among different Gaussians g_i, making it challenging to extract meaningful features from this set containing a few hundred thousands of them using neural networks. To address this, we introduce a novel representation that gives structure to this unstructured set of points and solves the permutation invariance issue for faster and better feature extraction. We propose to accomplish this by employing spherical mapping to map the 3DGS primitives to an image-like representation that is both invariant to random shuffling of 3DGS points and well structured.

We begin the mapping by inscribing the 3DGS object into a sphere with the same center as the object in the canonical space. Inscribing a 3DGS object into a sphere involves enclosing the object within a sphere. This begins by determining the geometric center of the object. The next step is to calculate the radius of the sphere, which is achieved by measuring the Euclidean distance from the center to the farthest point on the object. The radius of the sphere is defined such that the sphere fully encloses the object. The sphere acts as a bounding volume for the entire object.

We consider each Gaussian g_i in 3D to be centered at the mean position represented by σ_i with Cartesian coordinates (x_i, y_i, z_i). The aim is to get the spherical coordinates $(\rho_i, \theta_i, \phi_i)$ for each Gaussian g_i. To do so, we calculate the azimuthal θ_i and polar ϕ_i angles for each g_i along with the distance from the origin to the point, ρ_i. The spherical radius is defined as $\rho_i = \sqrt{x_i^2 + y_i^2 + z_i^2}$, the azimuthal angle as $\theta_i = \tan^{-1}(y_i, x_i)$, while the polar angle as $\phi_i = \cos^{-1}(z_i, \rho_i)$. The azimuthal and polar angles are then normalized, such that we can map them on a 2D UV map of $M \times N$ dimensionality with 14-channels. θ_i and ϕ_i are converted to degrees and mapped to UV image coordinates: $\theta_{i\text{ scaled}} = \left\lfloor \frac{\pi+\theta_i}{2\pi} \times \mathrm{M} \right\rfloor$, $\phi_{i\text{ scaled}} = \left\lfloor \frac{\phi_i}{\pi} \times \mathrm{N} \right\rfloor$ Each channel in the UV map stores 3DGS attributes, including $\{\sigma_i,\ r_i,\ s_i,\ o_i,\ c_i\} \in \mathbb{R}^{14}$. We refer this 14-channel UV map as *UVGS*, $U \in \mathbb{R}^{M \times N \times 14}$ defined as:

$$\mathrm{U}[\phi_{i\text{ scaled}}, \theta_{i\text{ scaled}}, :] = [\sigma_i,\ r_i,\ s_i,\ o_i,\ c_i]. \quad (1)$$

This transformed UVGS representation provides spatial coherence and solves the permutation invariance problem as any random arrangement of points will now map to the same UVGS representation U. It should be noted that this kind of transformation will also preserve the spatial correlation between the Gaussian points in 3D and transform them to 2D UV maps by mapping them to neighboring pixels. This provides both the local level correspondence among the neighboring Gaussians and the overall global correspondence for the object. Thus, solving the unstructured and discreteness problems. This enables standard neural network architectures (*e.g.* CNNs) to effectively capture correlations among neighboring Gaussians for efficient feature extraction.

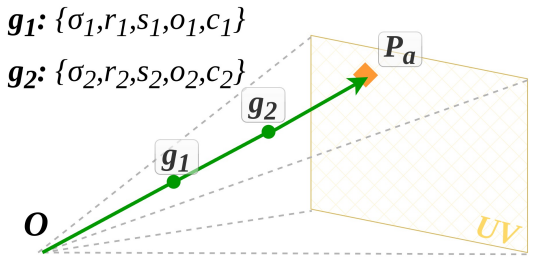

Figure 3. **Dynamic Selection**. In spherical mapping of 3DGS points to UV maps, multiple points may map to the same pixel, creating a many-to-one issue. Our Dynamic Selection approach addresses this by retaining the attributes of the point with the highest opacity per pixel on the same ray.

To further unify the extracted position (σ), transformation (r, s), and color (c, o) maps in a same feature space and use the existing image foundation models, we map the obtained UVGS $U \in \mathbb{R}^{M \times N \times 14}$ further to a 3-channel image $S \in \mathbb{R}^{M \times N \times 3}$ (termed as Super UVGS), using a Convolutional Neural Network (CNN). A multi-branch forward mapping network is employed to map $U \in \mathbb{R}^{M \times N \times 14}$ to the 3-channel Super UVGS $S \in \mathbb{R}^{M \times N \times 3}$. We provide all technical details in Sec. 3.3 and the supplementary material.

The Super UVGS representation effectively retains all the details of 3DGS attributes and can be directly utilized with existing widely available image-based models. We demonstrate this by showing perfect reconstruction of 3DGS object from Super UVGS image in the experiments section. This semantically structured representation S offers both local and global correspondence in representing Gaussian attributes and needs relatively less storage.

3.2. Dynamic GS Selection and Multiple Layers

When projecting 3DGS points to UV maps using spherical mapping, multiple points may map to the same pixel in UV space as shown in Fig. 3. Two 3DGS points (g_1) and (g_2) map to the same pixel on UV map (P_a) causing many-to-one mapping issue. To address this, we propose a Dynamic Selection approach where each UV pixel retains the 3DGS attribute with the highest opacity intersecting the same ray. Using the same example in Fig. 3, if opacity o_1 of Gaussian g_1 is less than opacity o_2 of g_2. Then only g_2 will be stored in the UV map at pixel P_a. We observed that this method helps maintain the geometry and appearance of the 3DGS object while resolving many-to-one mapping issues with minimal quality loss. For more complex objects or real-world scene representation, we stack multiple such layers of UV maps, where each UVGS pixel now holds attributes of the top-K opacity values of 3DGS primitives. This can be accomplished by inscribing the 3DGS object inside multiple spheres where each sphere maps the 3DGS attribute corresponding to the top-K^{th} opacity value along the same ray. More details on this are presented in the supplementary. To show the effectiveness of proposed UVGS

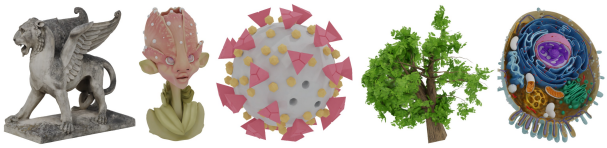

Figure 4. Complex object reconstructions (K=4) using pretrained image-based autoencoder.

maps in capturing the intricacies of a complex objects, we used a pretrained image based autoencoder to reconstruct objects using a 4 layer UVGS as shown in Fig. 4.

3.3. Mapping Networks

Our goal is to bring the extracted UVGS maps to a common feature space to better represent the object collectively and to make the 14-channel UVGS representations $U \in \mathbb{R}^{M\times N\times 14}$ work with the widely available image based foundation models. To accomplish this, we map it to a 3-channel image which can be easily processed by the existing architectures while also maintaining the spatial correspondence. We design a simple yet effective multi-branch CNN to extract features from different UVGS attributes and map them to a 3-channel feature-rich image, termed Super UVGS. The structured UVGS maps provides local and global features that can be learned by a CNN.

Forward Mapping The first layer is a set of three mapping branches for position, transform, and appearance ($\phi_P^f,\ \phi_T^f,\ \phi_A^f$) respectively. We refer to them as position, transformation, and appearance branch. The position branch takes the mean position (σ) as an input and processes it to give a position feature map M_P. Similarly, the transformation branch takes the rotation (r) and scale (s) together to generate another feature map M_T. The last, appearance branch takes the color (c) and opacity (o) together to produce another feature map M_A. All the three features maps from position, transformation, and appearance branch are concatenated to get a final feature map, before passing them to the next module, called the Central Branch. The central branch (ϕ_C^f) is composed of multiple hidden Convolution layers, where each layer is followed by BatchNorm and ReLu activation. The last layer of the central branch is activated using $tanh$ to ensure the Super UVGS does not take any ambiguous value resulting in gradient explosion or undesired artifacts. The obtained Super UVGS S representation squeezes all the 3DGS attributes to a 3 dimensional image while also maintaining local and global structural correspondence among them.

Inverse Mapping We design an inverse mapping network that aims to map the obtained 3-channel Super UVGS image $S \in \mathbb{R}^{M\times N\times 3}$ back to the UVGS maps to obtain each of the five different 3DGS attributes $\{\sigma,\ r,\ s,\ o,\ c\}$. The inverse mapping network simply follows the forward mapping network architecture in the reverse order, where at first, we put the Central Branch (ϕ_{iC}) followed by attribute specific position, transformation, and appearance branches ($\phi_{iP},\ \phi_{iT},\ \phi_{iA}$). We provide more details on mapping network in the supplementary.

Branched mapping layers: The rationale behind using branched mapping layers in both forward and reverse mapping networks is to prevent the incompatibility issues arising due the the different value distribution of 3DGS attributes. Note that the disparate distributions of values within each set of attributes in 3D Gaussian Splatting, (*i.e.*, mean position, transformation, and color), pose a challenge to the model when processed collectively. For instance, neighboring Gaussians in UVGS maps show smooth changes in position and color values but typically have large variations in rotation, scale, and opacity values. This results in gradient anomalies and slow convergence. To address this, we propose a multi-branch network architecture, where attribute-specific branches implicitly learn to process these distinct attribute specific properties, focusing on their unique features before passing them to the central branch. The central branch receives a concatenated stack of processed attributes and exploits the correlation between them by extracting local feature correspondences. This information is then mapped to a 3-channel Super UVGS image, effectively capturing the complex relationships between the various attributes. This approach enables our network to manage diverse attribute distributions, resulting in faster convergence, improved accuracy, and specialized processing for each attribute set.

Reconstruction Losses: Since the obtained UVGS maps have both local and global features, we opted for image-based losses to train the overall architecture. We use a set of Mean Squared Error (MSE) and Learned Perceptual Image Patch Similarity (LPIPS) [65] between the obtained UVGS from spherical mapping U and the predicted UVGS $\hat{U}$ from inverse mapping network. We calculate the LPIPS loss over four attributes of 3DGS including mean position (σ), view independent color (c), scale (s), and rotation (r). The overall LPIPS loss for UV maps can be written as a linear sum of individual attribute loss terms as:

$$\mathcal{L}_{UV-lpips} = \mathcal{L}_\sigma + \mathcal{L}_s + \mathcal{L}_r + \mathcal{L}_c \tag{2}$$

The overall loss function for the training can be written as: $\mathcal{L}_{uvgs} = \mathcal{L}_{mse} + \lambda.\mathcal{L}_{UV-lpips}$ where λ is a scalar and varied from 0 to 10 during the course of training.

4. Experiments

3DGS Dataset and UV maps To train the mapping networks and learn a latent space for unconditional and conditional sampling, we need large amount of 3DGS assets. However, there's a lack of such a large-scale dataset for high quality 3DGS assets. To this end, we create a custom large scale dataset by converting the Objaverse [10] meshes into 3DGS representation [1]. We start by designing a scene of 88

[1] Sketchfab data was filtered out of training data due to its license

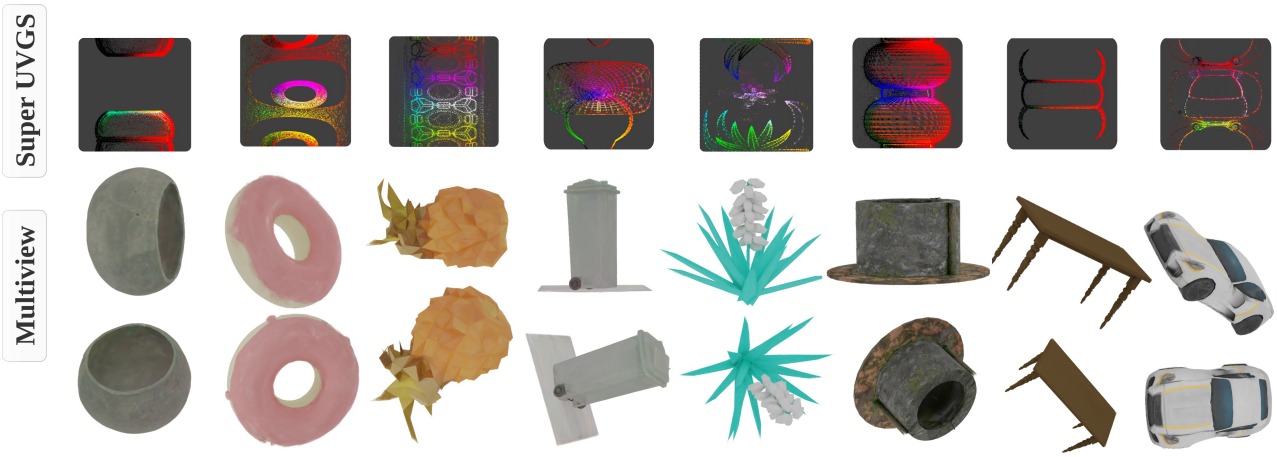

Figure 5. Figure shows a wide variety of high-quality unconditional generation result from our method using Latent Diffusion Model. We train an LDM to randomly sample Super UVGS images from random noise. The Super UVGS can be converted to 3DGS object using inverse mapping network and inverse spherical projection. The unconditional generation model was trained on Objaverse dataset.

cameras in a canonical space and use it to capture Objaverse objects from various angles covering all of the object views. The 88 rendered views from different angles are then used to train a 3DGS for 10K iterations using [19]. This way, we create a high-quality and large-scale 3DGS dataset of $\sim 400K$ objects and scenes from Objaverse. We only use static scenes or objects from Objaverse. After fitting all the object to 3DGS representation, we convert the objects to the corresponding UV maps (*i.e.*, UVGS) through Spherical Mapping as illustrated in Fig. 2. For the course of our experiments, we only map the objects to a single layer UV maps as it was sufficient to represent the general purpose Objaverse objects with minimal quality loss. Through mapping, we gathered a UVGS dataset of $\sim 400K$ maps. We fix the size of the UVGS maps to 512×512. Through our experiments, we found that UV maps of size 512×512 are sufficient to represent objects in our dataset and capable of storing upto $262K$ unique Gaussians. Table 1 compares 1-4 layer UV maps. We also did experiments on ShapeNet [5] cars dataset for evaluation purposes.

Baselines & Metrics: To evaluate the quality of reconstructed 3DGS objects from both Super UVGS image and Autoencoder latent space to 3D, we use PSNR and LPIPS [64]. The aim is to convert the given 3DGS object to UVGS, and then to Super UVGS, and further to Autoencoder's latent space and calculate the metrics from the reconstructions at every step to prove the proposed method doesn't significantly affect the quality of reconstructions, while also providing a structurally meaningful representation that is much compact and easier to use with existing image based models. We compare the generational capabilities of our method against various conditional and unconditional SOTA 3D object generation method including the ones using multiview rendering for optimization DiffTF [3], Get3D [14], methods trying to give structural representation to Gaussians, GaussianCube [61], and general purpose SOTA large 3D content generation models like DreamGaussian [48], LGM [50], and EG3D [4]. We also compare the quality of our generation results using FID and KID.

Table 1. PSNR and LPIPS comparison for various reconstruction methods using UVGS and Super UVGS representations on Objaverse Cars and Full datasets. AE, VAE, VQVAE are pretrained image based models. K is the number of UVGS layers used. We also report the compression % (CP) compared to the fitted 3DGS.

Method	**PSNR**(C/F)	**LPIPS**(C/F)	**CP**(%)
3DGS	34.6 / 34.2	0.02 / 0.02	0
UVGS (@K=1)	31.3 / 31.1	0.06 / 0.06	53.0
UVGS (@K=2)	32.8 / 31.9	0.04 / 0.05	45.6
UVGS (@K=4)	34.2 / 33.2	0.02 / 0.03	33.3
Super UVGS (@K=1)	31.2 / 31.1	0.07 / 0.08	89.7
AE (@K=1)	30.9 / 30.8	0.07 / 0.09	99.5
VAE (@K=1)	30.6 / 30.9	0.07 / 0.09	99.5
VQVAE (@K=1)	30.3 / 30.1	0.08 / 0.10	99.7

Mapping Network Training Details We train the forward and inverse mapping networks to project the obtained UVGS maps $U \in \mathbb{R}^{M \times N \times 14}$ to Super UVGS image $S \in \mathbb{R}^{M \times N \times 3}$, and back to the reconstructed UVGS maps $\hat{U} \in \mathbb{R}^{M \times N \times 14}$. We provide an in-depth discussion of all implementation details in the supplementary material.

4.1. UVGS AutoEncoder and 3DGS Compression

The obtained Super UVGS image is a structurally meaningful representation that can have various applications in the generation and reconstruction of new 3D assets as it contains features that can be learned by the existing image based models. Through our experiments, we show that a 3-channel Super UVGS image can be directly reconstructed using a pretrained image based Autoencoders or VAEs without any fine-tuning. We tested on three different models in-

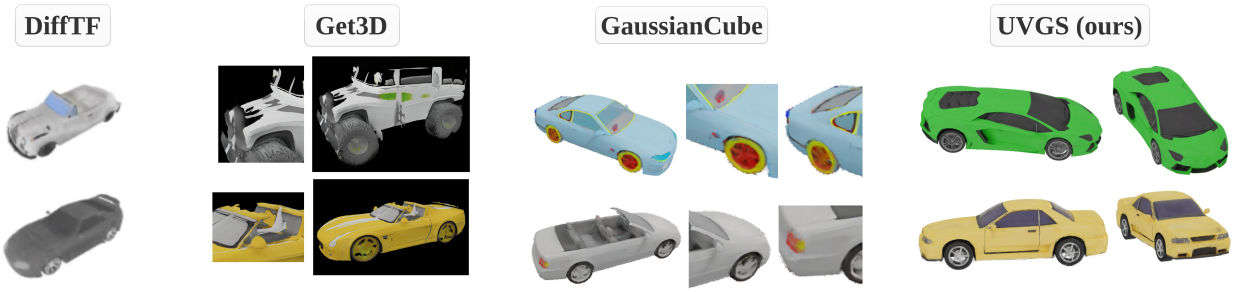

Figure 6. Comparison of unconditional 3D asset generation on the cars category with SOTA methods. Figure shows that DiffTF [3] produces low-quality, low-resolution cars lacking detail. While Get3D [14] achieve higher resolution, it suffers from 3D inconsistency, numerous artifacts, and lacks 3D detail. Similar issues are found in GaussianCube [61] along with symmetric inconsistency in the results. In contrast, our method generates high-quality, high-resolution objects that are 3D consistent with sharp and well-defined edges.

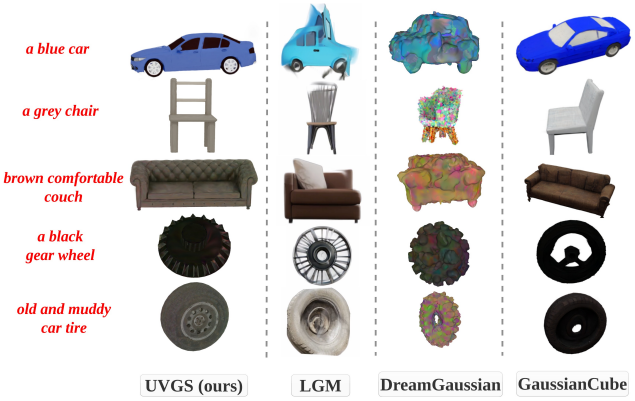

Figure 7. We compare the performance of our model against various SOTA methods for text-conditional object synthesis. Our method not only generates high-quality assets for simpler objects, but also for complicated objects with intricate geometry.

cluding image AE, KL-VAE [20], VQVAE [52] and each performed quite well without any significant quality loss. The reconstruction PSNR and LPIPS values are presented in Table 1. This means we can now leverage the powerful compression capabilities of image based Autoencoders to compress the storage requirements of 3DGS by more than 99%. We have shows the storage comparison results in Table 1. It is interesting to note that the Super UVGS representation itself can be used to compress the memory requirement for storing 3DGS object by up to 89.7%.

4.2. Unconditional & Conditional Generation

We aim to show the effectiveness of Super UVGS representation for directly generating 3DGS objects from a learned latent space. We consider Super UVGS images as a compact and structured proxy for representing 3DGS objects as it maintains the 3D object information while also providing learnable features. The existing methods fail to directly generate a large number of Gaussians (*e.g.* 100K+) to represent objects with sufficient quality either due to the lack of 3D generative model architectures that support such large number of unstructured points, or due to the lack of a large 3DGS dataset [32, 46, 50, 61, 69]. We leverage the Super UVGS representation and use the existing 2D image generative models like Diffusion Models [17] for this task. Specifically, to train a generative model capable of randomly sampling new high-quality 3DGS assets for various downstream tasks, we use an unconditional Latent Diffusion Model (LDM) [41] on the obtained Super UVGS images. As illustrated in Section 4.1, we can use a pretrained image VAE to map the Super UVGS image to a latent space and reconstruct back. Hence, we only train a LDM on the latent space. More implementation details are provided in the supplementary.

Unconditional LDM: To design a generative model capable of randomly sampling new high-quality 3DGS assets for various downstream tasks, we train an unconditional LDM [41] on the learned Super UVGS images. Following [38, 41, 63], we use DDIM [17] for faster and consistent sampling with up to 1000 time steps used in the forward diffusion process, and 20 during denoising. Once trained, the model is used to randomly sample Super UVGS images, resulting in high quality 3DGS assets through inverse mapping. Results are presented in Fig 5. We demonstrate that our method inherently learns to generate multiview consistent images due to the powerful Super UVGS representation unlike most prior works using rendering-based losses.

Conditional LDM: Similar to unconditional generation, we also trained a text-conditioned LDM following the SD's [38, 41, 63] pipeline and using the predicted text for our dataset. The trained model can be used to generate high-quality text-conditioned 3DGS assets that are multi-view consistent. The results are demonstrated in Fig 6.

The above experiments proves the effectiveness of our proposed Super UVGS representation in 3D object synthesis using widely available 2D image models. It also highlights that this compact 3-channel Super UVGS representation stores not just the spatial correspondence among different pixels, but also the rich 3D information of the objects. This way, we can easily convert a 3D asset generation problem into a 2D image generation problem without the use of any complex 3D architecture to handle large amount of un-

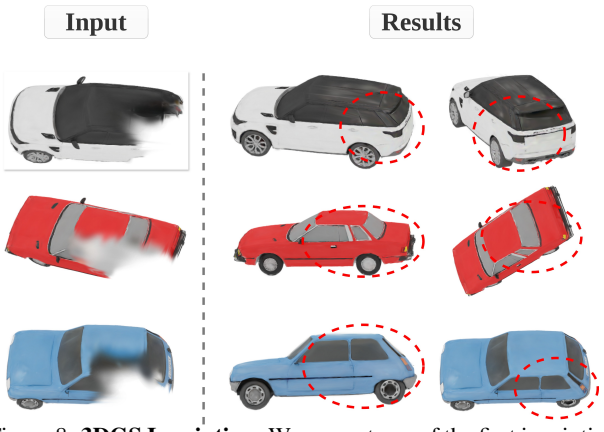

Figure 8. **3DGS Inpainting**: We present one of the first inpainting results on 3DGS directly leveraging the Super UVGS images and the denoising capabilities of diffusion models.

structured and permutation invariant 3DGS primitives, and neither relying upon computationally expensive multiview rendering or SDS loss. Baseline comparison for unconditional and conditional generation is presented in Table 2.

4.3. 3DGS Inpainting

Leveraging the powerful Super UVGS representation, we present in Fig. 8 one of the first experiments on inpainting 3DGS directly without using any multiview rendering or distilling information from diffusion models. We try to recover the missing Gaussians by leveraging the denoising capabilities of LDM and trying to predict the missing corresponding parts of the Super UVGS image. We believe, this can have potential applications in sparse view reconstruction. More details are given in the supplementary.

4.4. Ablation Studies & Discussion

We conduct exhaustive ablation studies to justify some of our framework's design choices including the effect of branching in mapping networks, the use of single layer UVGS maps, and the resolution of UVGS maps. We performed our experiments on our custom Objaverse [10] 3DGS dataset and evaluate the performance of our model in terms of PSNR, SSIM, and LPIPS. The results are presented in Table 3. From the table, it can be seen that by using four layers of UVGS maps (K=4), we can almost match the reconstruction quality of fitted 3DGS results, while we realized that simply with a single layer UVGS, we are able to maintain the overall geometry and appearance of the object for our dataset with a PSNR of more than 30. We also compared the reconstruction performance of our method with and without using branching in the mapping network. It can be clearly seen that using branching network significantly increases the reconstruction quality from Super UVGS space. The main reasoning behind this is the specialization of attributes that the branching provides to individually process each attribute first.

Table 2. We compare the FID and KID of unconditional generation using the current SOTA methods on 20K randomly generated samples from each method and ours. We also compare our method against SOTA text-conditioned generation frameworks on CLIP Score for 10K generated objects from each method.

Unconditional Generation			**Text-Conditioned Generation**	
Method	**FID ↓**	**KID ↓**	**Method**	**CLIP Score ↑**
Get3D [14]	53.17	4.19	DreamGaussian [48]	28.51
DiffTF [3]	84.57	8.73	Shap. E [7]	30.53
EG3D [4]	74.51	6.62	LGM [50]	30.74
GaussianCube	34.67	3.72	GaussianCube [61]	30.34
UVGS (Ours)	**26.20**	**3.24**	**UVGS (Ours)**	**32.62**

Table 3. We present quantitative ablation study for number of UVGS layers (K), UVGS map resolution, and the effect of branching in mapping network on the Objaverse 3DGS dataset.

Method	**PSNR**	**LPIPS**	**UVGS Size**	**PSNR**	**LPIPS**
UVGS @$K=1$	31.1	0.06	512×512 (@$K=1$)	31.1	0.08
UVGS @$K=2$	31.9	0.05	256×256 (@$K=1$)	28.2	0.23
UVGS @$K=4$	33.2	0.03	Without Branching	27.8	0.31

Limitations & Future Work: While single layer UVGS images can recover the geometry of the object, they sometimes suffer in terms of appearance and the generated objects might look washed out. We believe this can be solved by using a multi-layer UVGS maps. Similarly, the single layer UVGS map is limited to representing simpler everyday objects, and may not be sufficient to represent highly-detailed and complex objects or scenes. In the future, we want to extend this framework to learn features for real-world scenes and complex objects like a human head with multi-layer UV mapping. We also want to make this representation more efficient by better utilizing the empty pixels of UVGS maps and Super UVGS images while also maintaining the underlying features and 3D information.

5. Conclusion

We introduced a novel method to solve the underlying issues with 3D Gaussian Splatting (3DGS) that prevent the direct integration of them with the large number of existing image foundational models. We proposed UVGS - a structured representation for 3DGS obtained by spherical mapping of 3DGS primitives to UV maps. We further squeezed the multi-attribute UVGS maps to a 3-channel unified and structured Super UVGS image, which not only maintains the 3D structural information of the object, but also provides a compact feature space for 3DGS attributes. The obtained Super UVGS images are directly integrated with the existing image foundational models for 3DGS compression and unconditional and conditional generation using diffusion models. Leveraging these Super UVGS images, we showed one of the first inpainting experiments on 3DGS.

ACKNOWLEDGEMENTS A part of this work was supported by NSF CAREER grant 2143576 and ONR DURIP grant N00014-23-1-2804.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 15, 16

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5855–5864, 2021. 2

[3] Ziang Cao, Fangzhou Hong, Tong Wu, Liang Pan, and Ziwei Liu. Large-vocabulary 3d diffusion model with transformer. *arXiv preprint arXiv:2309.07920*, 2023. 3, 6, 7, 8, 16, 17

[4] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16123–16133, 2022. 2, 6, 8, 16

[5] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 3, 6

[6] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European conference on computer vision*, pages 333–350. Springer, 2022. 2

[7] Minghao Chen, Junyu Xie, Iro Laina, and Andrea Vedaldi. Shap-editor: Instruction-guided latent 3d editing in seconds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26456–26466, 2024. 8, 16

[8] Zilong Chen, Feng Wang, Yikai Wang, and Huaping Liu. Text-to-3d using gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21401–21412, 2024. 3

[9] Yen-Chi Cheng, Hsin-Ying Lee, Sergey Tulyakov, Alexander G Schwing, and Liang-Yan Gui. Sdfusion: Multimodal 3d shape completion, reconstruction, and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4456–4465, 2023. 2

[10] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023. 3, 5, 8

[11] Slava Elizarov, Ciara Rowles, and Simon Donné. Geometry image diffusion: Fast and data-efficient text-to-3d with image-based surface representation. *arXiv preprint arXiv:2409.03718*, 2024. 2

[12] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021. 15

[13] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5501–5510, 2022. 2

[14] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *Advances In Neural Information Processing Systems*, 35:31841–31854, 2022. 3, 6, 7, 8, 16, 17

[15] Anchit Gupta, Wenhan Xiong, Yixin Nie, Ian Jones, and Barlas Oğuz. 3dgen: Triplane latent diffusion for textured mesh generation. *arXiv preprint arXiv:2303.05371*, 2023. 2

[16] Xianglong He, Junyi Chen, Sida Peng, Di Huang, Yangguang Li, Xiaoshui Huang, Chun Yuan, Wanli Ouyang, and Tong He. Gvgen: Text-to-3d generation with volumetric representation. In *European Conference on Computer Vision*, pages 463–479. Springer, 2025. 2, 3

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 3, 7, 15

[18] Han Huang, Yulun Wu, Junsheng Zhou, Ge Gao, Ming Gu, and Yu-Shen Liu. Neusurf: On-surface priors for neural surface reconstruction from sparse input views. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2312–2320, 2024. 2

[19] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2, 6, 14

[20] Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 7

[21] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023. 15, 16

[22] Jiahe Li, Jiawei Zhang, Xiao Bai, Jin Zheng, Xin Ning, Jun Zhou, and Lin Gu. Dngaussian: Optimizing sparse-view 3d gaussian radiance fields with global-local depth normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20775–20785, 2024. 2

[23] Shujuan Li, Junsheng Zhou, Baorui Ma, Yu-Shen Liu, and Zhizhong Han. Neaf: Learning neural angle fields for point normal estimation. In *Proceedings of the AAAI conference on artificial intelligence*, pages 1396–1404, 2023. 2

[24] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 300–309, 2023. 2

[25] Tiange Luo, Chris Rockwell, Honglak Lee, and Justin Johnson. Scalable 3d captioning with pretrained models. *Advances in Neural Information Processing Systems*, 36, 2024. 15

[26] Baorui Ma, Haoge Deng, Junsheng Zhou, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Geodream: Disentangling 2d and geometric priors for high-fidelity and consistent 3d generation. *arXiv preprint arXiv:2311.17971*, 2023. 2

[27] Qi Ma, Yue Li, Bin Ren, Nicu Sebe, Ender Konukoglu, Theo Gevers, Luc Van Gool, and Danda Pani Paudel. Shapesplat: A large-scale dataset of gaussian splats and their self-supervised pretraining. *arXiv preprint arXiv:2408.10906*, 2024. 2, 3

[28] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12663–12673, 2023. 2

[29] 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. *Communications of the ACM*, 65(1):99–106, 2021. 1, 2

[30] Chaerin Min and Srinath Sridhar. Genheld: Generating and editing handheld objects. *arXiv preprint arXiv:2406.05059*, 2024. 2

[31] Chaerin Min, Sehyun Cha, Changhee Won, and Jongwoo Lim. Tsdf-sampling: Efficient sampling for neural surface field using truncated signed distance field. *arXiv preprint arXiv:2311.17878*, 2023. 2

[32] Yuxuan Mu, Xinxin Zuo, Chuan Guo, Yilin Wang, Juwei Lu, Xiaofeng Wu, Songcen Xu, Peng Dai, Youliang Yan, and Li Cheng. Gsd: View-guided gaussian splatting diffusion for 3d reconstruction. *arXiv preprint arXiv:2407.04237*, 2024. 2, 3, 7

[33] Norman Müller, Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulo, Peter Kontschieder, and Matthias Nießner. Diffrf: Rendering-guided 3d radiance field diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4328–4338, 2023. 2

[34] Francesco Palandra, Andrea Sanchietti, Daniele Baieri, and Emanuele Rodolà. Gsedit: Efficient text-guided editing of 3d objects via gaussian splatting. *arXiv preprint arXiv:2403.05154*, 2024. 2

[35] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 2, 3

[36] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 2

[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*, pages 8748–8763. PMLR, 2021. 15, 16

[38] Aashish Rai and Srinath Sridhar. Egosonics: Generating synchronized audio for silent egocentric videos. *arXiv preprint arXiv:2407.20592*, 2024. 7

[39] Aashish Rai, Hiresh Gupta, Ayush Pandey, Francisco Vicente Carrasco, Shingo Jason Takagi, Amaury Aubel, Daeil Kim, Aayush Prakash, and Fernando De la Torre. Towards realistic generative 3d face models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3738–3748, 2024. 2

[40] Amit Raj, Srinivas Kaza, Ben Poole, Michael Niemeyer, Nataniel Ruiz, Ben Mildenhall, Shiran Zada, Kfir Aberman, Michael Rubinstein, Jonathan Barron, et al. Dreambooth3d: Subject-driven text-to-3d generation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2349–2359, 2023. 2

[41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 3, 7, 15, 16

[42] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015. 16

[43] Li Shen and Fillia Makedon. Spherical mapping for processing of 3d closed surfaces. *Image and vision computing*, 24 (7):743–761, 2006. 2, 12

[44] J Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3d neural field generation using triplane diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20875–20886, 2023. 2

[45] Jingxiang Sun, Bo Zhang, Ruizhi Shao, Lizhen Wang, Wen Liu, Zhenda Xie, and Yebin Liu. Dreamcraft3d: Hierarchical 3d generation with bootstrapped diffusion prior. *arXiv preprint arXiv:2310.16818*, 2023. 2

[46] Stanislaw Szymanowicz, Chrisitian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10208–10217, 2024. 2, 3, 7

[47] Fariborz Taherkhani, Aashish Rai, Quankai Gao, Shaunak Srivastava, Xuanbai Chen, Fernando De la Torre, Steven Song, Aayush Prakash, and Daeil Kim. Controllable 3d generative adversarial face model via disentangling shape and appearance. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 826–836, 2023. 2

[48] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*, 2023. 2, 3, 6, 8, 16

[49] Junshu Tang, Tengfei Wang, Bo Zhang, Ting Zhang, Ran Yi, Lizhuang Ma, and Dong Chen. Make-it-3d: High-fidelity 3d

creation from a single image with diffusion prior. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22819–22829, 2023. 2

[50] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European Conference on Computer Vision*, pages 1–18. Springer, 2025. 6, 7, 8, 16

[51] Zhicong Tang, Shuyang Gu, Chunyu Wang, Ting Zhang, Jianmin Bao, Dong Chen, and Baining Guo. Volumediffusion: Flexible text-to-3d generation with efficient volumetric encoder. *arXiv preprint arXiv:2312.11459*, 2023. 2

[52] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 7

[53] Tengfei Wang, Bo Zhang, Ting Zhang, Shuyang Gu, Jianmin Bao, Tadas Baltrusaitis, Jingjing Shen, Dong Chen, Fang Wen, Qifeng Chen, et al. Rodin: A generative model for sculpting 3d digital avatars using diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4563–4573, 2023. 2

[54] Yiming Wang, Qin Han, Marc Habermann, Kostas Daniilidis, Christian Theobalt, and Lingjie Liu. Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3295–3306, 2023. 2

[55] Yuxuan Wang, Xuanyu Yi, Zike Wu, Na Zhao, Long Chen, and Hanwang Zhang. View-consistent 3d editing with gaussian splatting. In *European Conference on Computer Vision*, pages 404–420. Springer, 2025. 2

[56] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[57] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20310–20320, 2024. 2

[58] Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. *arXiv preprint arXiv:2403.14621*, 2024. 2, 3

[59] Xingguang Yan, Han-Hung Lee, Ziyu Wan, and Angel X Chang. An object is worth 64x64 pixels: Generating 3d object via image diffusion. *arXiv preprint arXiv:2408.03178*, 2024. 2

[60] Taoran Yi, Jiemin Fang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. Gaussiandreamer: Fast generation from text to 3d gaussian splatting with point cloud priors. *arXiv preprint arXiv:2310.08529*, 2023. 2

[61] Bowen Zhang, Yiji Cheng, Jiaolong Yang, Chunyu Wang, Feng Zhao, Yansong Tang, Dong Chen, and Baining Guo. Gaussiancube: Structuring gaussian splatting using optimal transport for 3d generative modeling. *arXiv preprint arXiv:2403.19655*, 2024. 2, 3, 6, 7, 8, 16, 17

[62] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. In *European Conference on Computer Vision*, pages 1–19. Springer, 2025. 2, 3

[63] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023. 7

[64] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 6, 14

[65] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 5

[66] Junsheng Zhou, Yu-Shen Liu, and Zhizhong Han. Zero-shot scene reconstruction from single images with deep prior assembly. *arXiv preprint arXiv:2410.15971*, 2024. 2

[67] Junsheng Zhou, Weiqi Zhang, and Yu-Shen Liu. Diffgs: Functional gaussian splatting diffusion. *arXiv preprint arXiv:2410.19657*, 2024. 2, 3

[68] Junsheng Zhou, Weiqi Zhang, Baorui Ma, Kanle Shi, Yu-Shen Liu, and Zhizhong Han. Udiff: Generating conditional unsigned distance fields with optimal wavelet diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21496–21506, 2024. 2

[69] Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10324–10335, 2024. 2, 3, 7, 16

[70] Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10324–10335, 2024. 2

UVGS: Reimagining Unstructured 3D Gaussian Splatting using UV Mapping

Supplementary Material

Our supplementary material contains a wide range of information that cover implementation details for our networks and training procedures, as well as a large variety of qualitative results.

Supplementary Video: We refer the interested reader to the supplementary video where we provide an overview of how our proposed approach works as well as a plethora of qualitative results across different tasks.

1. Spherical Mapping

Spherical Mapping: Spherical mapping [43] is a fundamental technique in computer graphics that is used to project 3D meshes onto a 2D map generally for texture mapping, where a 2D image is wrapped around a 3D object, such as a cylinder or a sphere. However, cylindrical mapping fails to capture the top and bottom parts of the object in the same UV map, and can introduce distortions for objects that extend far in the Z-direction. Hence we opted for spherical mapping the process of which involves converting 3D Cartesian coordinates (x, y, z) into spherical coordinates (ρ, θ, ϕ) and then mapping these onto a 2D plane. Algorithm [1] explains spherical unwrapping in detail for a single layer(K=1). The same process can be repeated for multiple layers, by keeping a track of opacity values.

Thresholding Opacity 3DGS use multiple points with varying opacity values to represent an object from any specific viewpoint. However, it is oftentimes noticed that many of these points have very low opacity values and do not contribute to the object's overall representation or appearance. We filter these points using a threshold opacity value with no impact on the object's overall geometry and representation to reduce the number of tractable primitives.

Dynamic GS Selection and Multiple Layers When projecting 3DGS points to UV maps using spherical mapping, multiple points may map to the same pixel in UV space as shown in Fig. 3. The two 3DGS points (g_1) and (g_2) map to the same pixel on UV map (P_a) causing many-to-one mapping. However, the UV map can only hold a single 3DGS primitive at any given pixel. To address this, we propose a Dynamic Selection approach where each UV pixel retains the 3DGS attributes with the highest opacity intersecting the same ray from the centroid to the farthest 3DGS primitive along the ray. Using the same example in Fig. 3, if opacity o_1 of Gaussian g_1 is less than opacity o_2 of g_2. then only g_2 attributes will be stored in the UV map at pixel P_a. Through multiple testing, we observed that this method helps retain the overall geometry and appearance of the 3DGS object while resolving many-to-one mapping issues with minimal quality loss.

This method with single layer is applicable to most of the objects in our dataset. However, this might fail in the case of more complex objects or real-world scene representation. There could be multiple layers of Gaussians holding higher opacity and contributing to the overall scene's appearance or geometry, and even partial or full occlusions. To better represent such objects and scenes and to prove the effectiveness of UVGS, we stack multiple layers of UV maps, where each UVGS layer holds the 3DGS primitives of the top-K^{th} opacity value intersecting the same ray. This can be accomplished by inscribing the 3DGS object inside multiple spheres where each sphere maps the 3DGS attribute corresponding to the top-K^{th} opacity value along the same ray. To show the effectiveness of proposed UVGS maps in capturing the intricacies of a complex real-world scene, we use a 12 layer UVGS map to reconstruct the real-world 3D scenes. The results are presented in Fig. 8. We also com-

Algorithm 1 Spherical Unwrapping for UVGS map (K=1).

Require: $3DGS \in \mathbb{R}^{n\times 14},\ (M, N) \in \mathbb{Z},\ K = 1$
Ensure: $position(\sigma), color(c), scale(s) \in \mathbb{R}^{n\times 3}$
Ensure: $rotation(r) \in \mathbb{R}^{n\times 4},\ opacity(o) \in \mathbb{R}^{n\times 1}$
Extract $xyz(\sigma),\ opac(o)$ from $3DGS$
Spherical radius, $r \leftarrow \sqrt{x^2 + y^2 + z^2}$
Azimuthal Angle, $\theta \leftarrow tan^{-1}(y, x)$
Polar Angle, $\phi \leftarrow cos^{-1}(z, r)$
$(\theta,\ \phi) \leftarrow (\text{deg}(\theta) + 180,\ \text{deg}(\phi))$
$\theta_{UV} \leftarrow \text{round}((\theta/360) \times M)$
$\phi_{UV} \leftarrow \text{round}((\phi/180) \times N)$
Initialize $UV_{map} \leftarrow \text{zeros}(M, N, 14)$
Initialize $UV_{opac} \leftarrow \text{zeros}(height, width)$
for all (t, P, xyz, o) in $(\theta_{UV}, \phi_{UV}, 3DGS, opac)$ **do**
  if $0 \le P < height$ and $0 \le t < width$ **then**
    if $UV_{map}[P, t]$ is 0 **then**
      $UV_{map}[P, t] \leftarrow 3DGS[ind]$
      $UV_{opac}[P, t] \leftarrow o$
    else
      if $o > UV_{opac}[P, t]$ **then**
        $UV_{map}[P, t] \leftarrow 3DGS[ind]$
      end if
    end if
  end if
end for
return UV_{map}

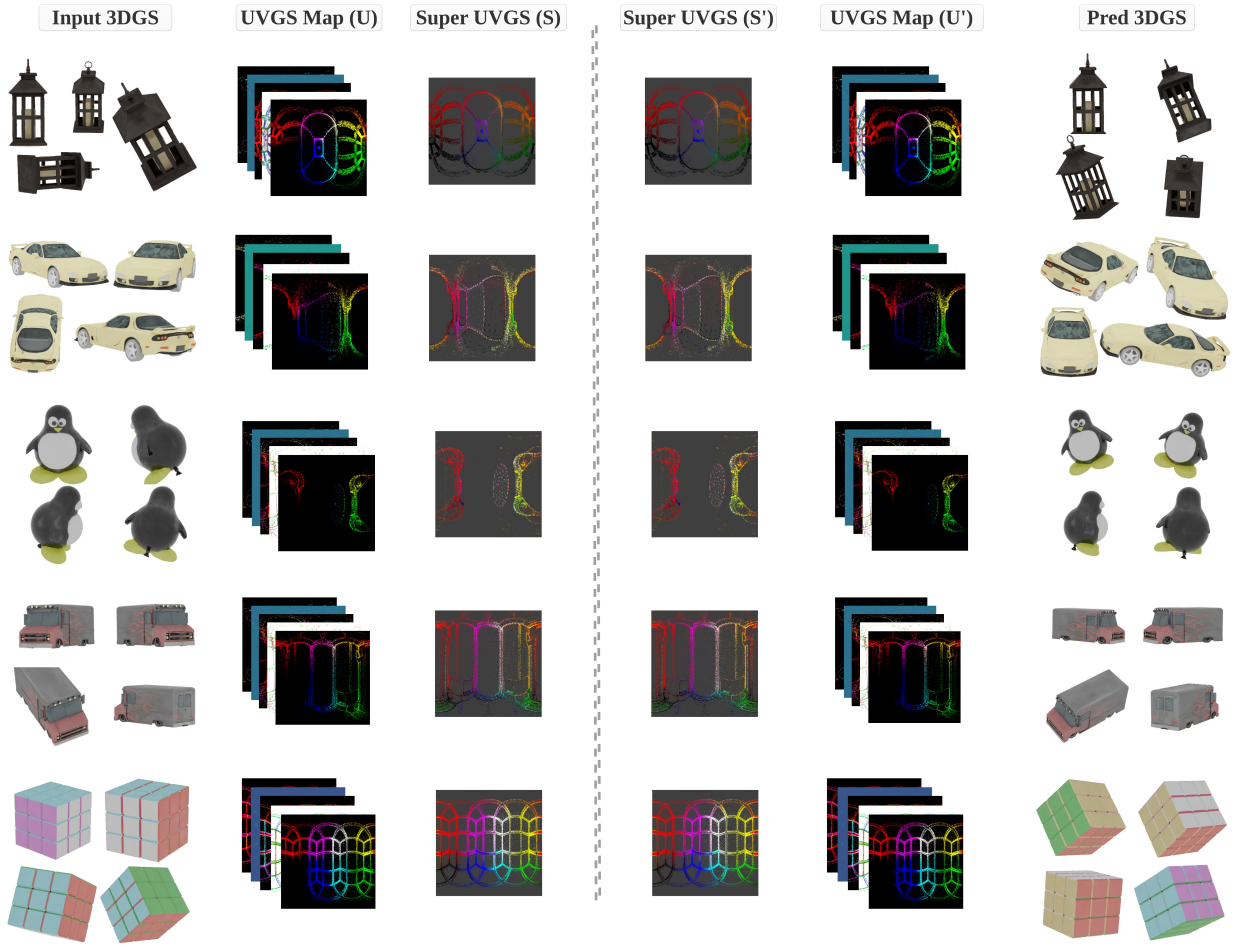

Figure 1. In this figure, we show the qualitative results of reconstructing 3DGS object using pretrained Image Autoencoder (A) via Super UVGS. We obtain UVGS maps (U) through spherical projection of 3DGS objects, followed by using forward mapping network to get Super UVGS (S). A pretrained AE is used to reconstruct Super UVGS (S'), which can be converted to UVGS maps (U') through inverse mapping network. At last, through inverse spherical mapping, we can get predicted 3DGS object which has the same appearance and geometry as the input object with minimal loss.

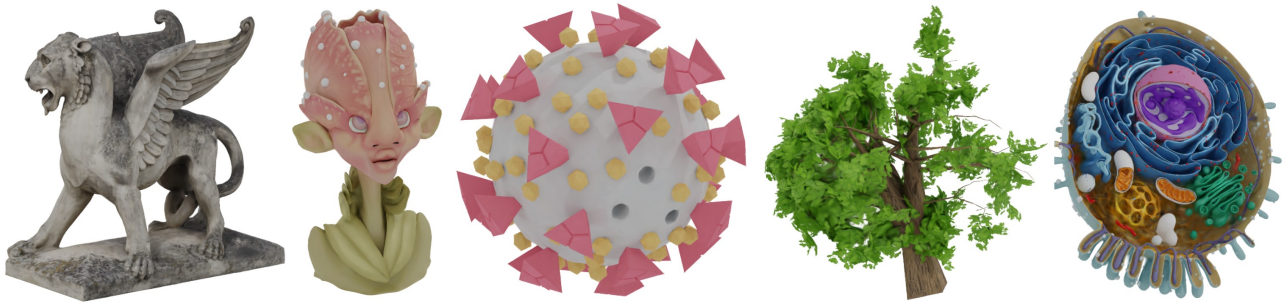

Figure 2. Complex object reconstructions (K=4) using pretrained image-based autoencoder.

pare the effect of increasing the number of UVGS layers in representing a real-world 3D scene in Fig. 3 In future work, we want to extend this ability for potentially many applications in 3D dynamic scene reconstructions using video diffusion models, and the segmentation or tracking of objects in 3DGS scenes as the features in the UVGS maps can be easily processed with the neural networks and tracked over time.

2. Mapping Networks

Forward Mapping Details: This process is defined as:

$$f^f_{map} = [\,[\phi^f_P(\sigma)]\,[\phi^f_T([r,s])]\,[\phi^f_A[o,c]]\,] \tag{3}$$

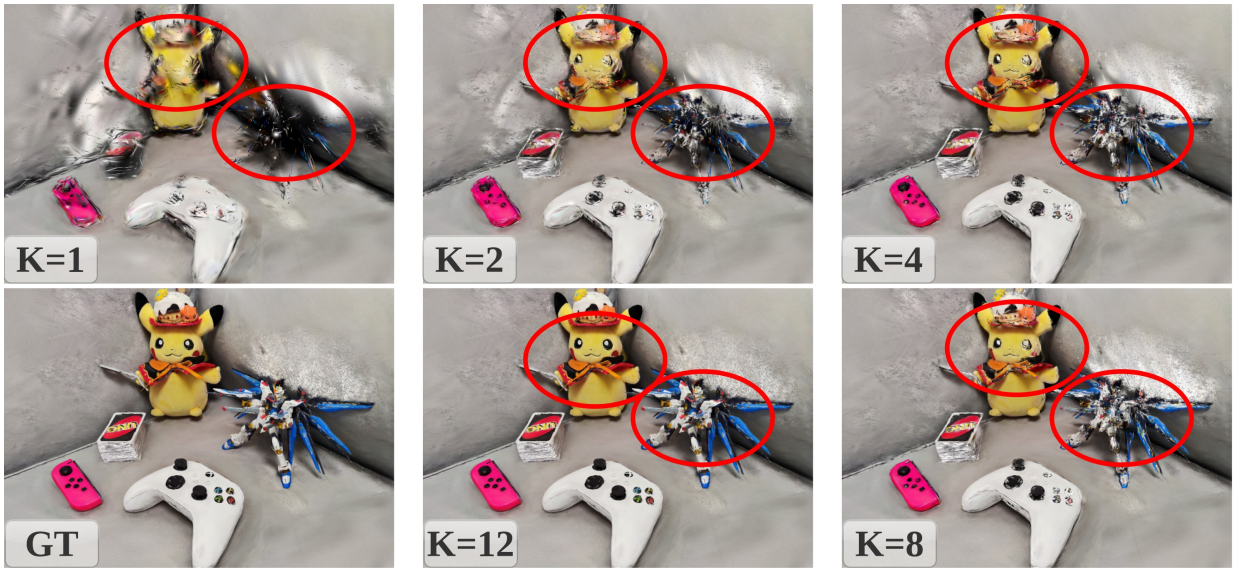

Figure 3. Reconstruction of a real-world scene for different K values. Smaller K results in many-to-one issue, hence lacking details.

The central branch (ϕ_C^f) is composed of $2L$ hidden Convolution layers. The first L hidden convolution layers increase the feature dimension at each step, while the last L layers does the inverse and squeezes the high-dimensional feature maps to 3 channels to output Super UVGS image $S \in \mathbb{R}^{M \times N \times 3}$.

$$S = tanh(\ \phi_C^f[f_{map}^f]\ ) \ \in \ \mathbb{R}^{(H,W,3)} \tag{4}$$

Each CNN layer is followed by a batch normalization layer and ReLU activation both in multi-branch and central branch modules. The last layer of central branch is activated using $tanh$ to ensure the Super UVGS doesn't take any ambiguous value resulting in gradient explosion or undesired artifacts. The obtained Super UVGS S representation squeezes all the 3DGS attributes to a 3 dimensional image while also maintaining local and global structural correspondence among them.

Inverse Mapping: The first L layers in the Central branch increases the feature dimension and the last L layers reduces them to obtain a combined feature map.

$$f_{map}^i = \phi_C^i(S)$$

The final layer is a set of 3 branches projecting the features to position, translation, and appearance attributes, respectively.

$$f_\sigma^i = [\phi_P^i(f_{map}^i)]$$
$$f_{r,s}^i = [\phi_T^i(f_{map}^i)]$$
$$f_{o,c}^i = [\phi_A^i(f_{map}^i)]$$

Similar to the forward mapping network, each layer in the central branch and attribute specific branches is followed by batch normalization and $Relu(.)$ activation. The last set of branch layers are activated using $tanh(.)$ to prevent ambiguous values resulting in gradient explosion or reconstruction artifacts.

$$\hat{U} = tanh(\ [[f_\sigma^i]\ [f_{r,s}^i]\ [f_{o,c}^i]]\ )$$

Losses Details: We used MSE to focus on pixel-wise difference during the training. We solely used MSE for a few iterations to make the mapping networks learn the overall structural representation of the UVGS map using:

$$\mathcal{L}_{mse} = \frac{1}{n}\sum_{i=1}^{n}(U_i - \hat{U}_i)^2. \tag{5}$$

After training the model for few iterations using MSE, we introduce the LPIPS loss giving same weight to both MSE and LPIPS over a few iterations. We observed that increasing the weight value of LPIPS over the iterations resulted in better and faster convergence results.

$$\mathcal{L}_{lpips} = \sum_l w_l \left\| \phi_l(x) - \phi_l(y) \right\|^2, \tag{6}$$

where $\phi_l(x)$ and $\phi_l(y)$ are feature maps extracted from pretrained layers of AlexNet[64].

Mapping Training Details: Before training the models, we normalized the different attributes in UVGS to $[-1, 1]$ using the same normalization functions as used in 3DGS paper[19]. The normalized UVGS maps are used to train the multi-branch forward and reverse mapping networks using MSE and LPIPS loss. We trained the mapping networks on $8 \times A100\ (80GB)$ GPUs with a Batch Size of 96 for 120 hours using Adam optimizer with a learning rate of $6e-5$ and set $\beta_1 = 0.5$ and $\beta_2 = 0.9$ with weight decay of 0.01.

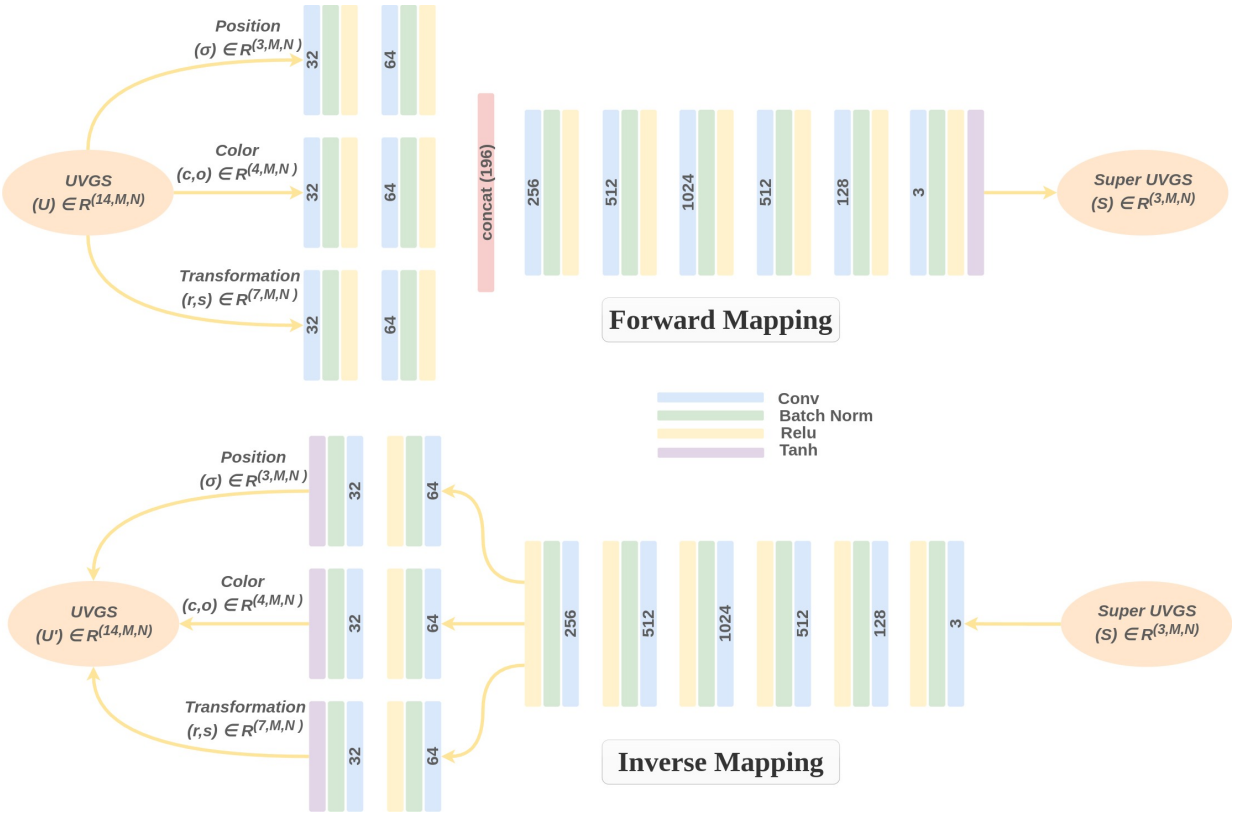

Figure 4. Forward Mapping Network for UVGS to Super UVGS mapping. The inverse mapping network follows just the inverse of this architecture with each attribute-specific branch now followed by $tanh()$ at the end.

We set the λ for LPIPS loss to be 0 for the first 24 hours of training and gradually increased it from 1 to 10 for the remaining training in a step of 1.

2.1. Interpolation with UVGS

We show that the proposed SuperUVGS representation can be used to perform local editing and interpolation directly in the UV domain. We can perform edits like swapping the parts of one object from the other, cropping the 3D object, or merging two objects together simply with the SuperUVGS images without any learning based method. The results are demonstrated in Fig 5.

3. LDM - Unconditional and Conditional Generation

Caption Generation To generate the relevant text captions for the objects in our dataset for conditional generation, we leverage CLIP [37], BLIP2 [21], and GPT4 [1] very similar to [25]. Specifically, we use BLIP2 to generate N different captions for randomly selected 20 views from the 88 rendered views for each object in the dataset. CLIP encoders are used to encode and calculate the cosine similarity between the N generated caption per view and the corresponding 20 views. The caption with max similarity is assigned to that particular view, resulting in 20 different captions for the same object. We now use GPT4 to extract a single caption distilling all the given 20 descriptions. We found that the resulting captions were very appropriate to the input objects, and thus we directly used them for conditional generation.

Figure 5. Linear interpolation between two 3DGS objects using SuperUVGS representation.

LDMs [17, 41] use pretrained VAEs [12] to convert the original image $x \in R^{H\times W\times 3}$ into a compact latent representation $z \in R^{h\times w\times c}$, where the forward and reverse diffusion processes are applied [41]. The VAE decoder then converts the compact latent representation back to pixels.

Table 4. We compare the FID and KID of unconditional generation using the current SOTA methods on 20K randomly generated samples from each method and ours. We also compare our method against SOTA text-conditioned generation frameworks on CLIP Score for 10K generated objects from each method.

Unconditional Generation			**Text-Conditioned Generation**	
Method	**FID ↓**	**KID ↓**	**Method**	**CLIP Score ↑**
Get3D [14]	53.17	4.19	DreamGaussian [48]	28.51
DiffTF [3]	84.57	8.73	Shap. E [7]	30.53
EG3D [4]	74.51	6.62	LGM [50]	30.74
GaussianCube	34.67	3.72	GaussianCube [61]	30.34
UVGS (Ours)	**26.20**	**3.24**	**UVGS (Ours)**	**32.62**

The objective function in latent diffusion model can be written as:

$$\mathbb{L}_{LDM} := \mathbb{E}_{\epsilon(x),\epsilon\sim\mathbb{N}(0,1),t}[||\epsilon - \epsilon_\theta(z_t, t)||_2^2] \quad (7)$$

where, $\mathbb{N}(0,1)$ is the Normal distribution, and t is the number of time steps and z_t is the noisy sample after t time steps.

Training was done using AdamW optimizer with a learning rate of $1e-4$ for 75 epochs on $8 \times A100\ (80GB)$ GPUs.

Once trained, we can randomly sample new high-quality 3DGS assets from the learned generative model.

To allow generation of objects from text, we also trained a conditional LDM, where we use Stable Diffusion (SD) [41] pipeline as it can use text prompt conditioning to guide the image generation through cross-attention. Similar to unconditional LDM, we use pretrained SD's VAE for mapping the Super UVGS image to a latent space and back to the reconstructed Super UVGS. The text prompts are given to a pretrained CLIP [37] text encoder to generate a text embedding $c_t \in \mathbb{R}^{77\times768}$, which is then passed to the UNet encoder of SD for cross-attention. We used a set of CLIP encoder and BLIP2 [21], and GPT4 [1] to generate captions for our dataset. The overall objective function for conditional LDM now becomes:

$$\mathbb{L}^C_{LDM} := \mathbb{E}_{\epsilon(x),\epsilon\sim\mathbb{N}(0,1),t,c_t}[||\epsilon - \epsilon_\theta(z_t, t, c_t)||_2^2] \quad (8)$$

where, $\epsilon_\theta(\cdot, t)$ is a time-conditional U-Net [42] model, $\mathbb{N}(0,1)$ is the Normal distribution, z_t is the latent code, and c_t is the text embedding. Training was done using AdamW optimizer with a learning rate of $1e-4$ for 50 epochs on $8 \times A100\ (80GB)$ GPUs. Once trained, this conditional LDM can use used to generate text-conditioned Super UVGS images, which can later be mapped to high-quality 3DGS objects.

4. Comparison with Baselines

We compare the generational capabilities of our method against various conditional and unconditional SOTA 3D object generation method on ShapeNet-cars dataset. Specifically, we used the methods using multiview rendering for optimization, like DiffTF [3] and Get3D [14]. We also compared our approach again the current SOTA methods trying to give structural representation to Gaussians, including GaussianCube [61] and TriplaneGaussian [69]. We also compared against general purpose SOTA large 3D content generation models like DreamGaussian [48], LGM [50], and EG3D [4].

To compare the quality of our generation results, as a standard practice, we use FID and KID for unconditional generation, and Clip Score for text-conditioned generation. Table 2 quantitatively compares the unconditional and conditional generation results of our method again various SOTA methods. From this table, it can be seen that our method performs a good job in unconditional generation of good quality 3D assets. The main reason behind this is the learned Super UVGS representation which not only maintains the appearance of the 3DGS object, but also serves as a proxy for geometrical shape by encoding all the 3DGS attributes into the same coherent feature space. Table 4 compares the CLIP Score of our text-conditioned generation results and the current SOTA methods. The unconditional and conditional qualitative comparison results are presented in Fig. 7 and Fig. 6, respectively.

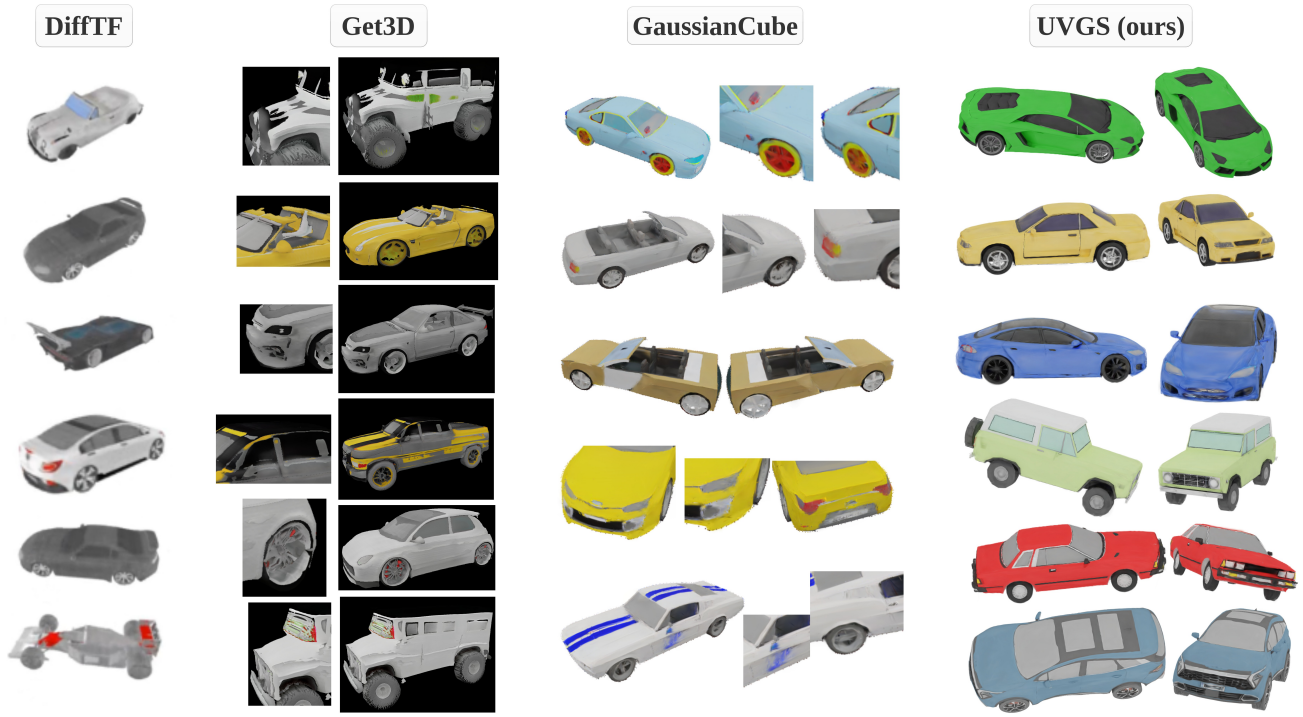

Figure 6. Here we show more comparison of unconditional 3D asset generation on the cars category with SOTA methods. Figure shows that DiffTF [3] produces low-quality, low-resolution cars lacking detail. While Get3D [14] achieve higher resolution, it suffers from 3D inconsistency, numerous artifacts, and lack richness in 3D detail. Similar issues are found in GaussianCube [61] along with symmetric inconsistency in the results. In contrast, our method generates high-quality, high-resolution objects that are 3D consistent with sharp, well-defined edges. The top three rows show the unconditional generation results of our method using ShapeNet dataset, while the bottom 3 show from Objaverse dataset.

Figure 7. Text-conditioned generation results on various baselines and the proposed method. Our method not only generates high-quality assets for simpler objects, but also for complicated objects with intricate geometries like *the wheel* or *the airplane*.

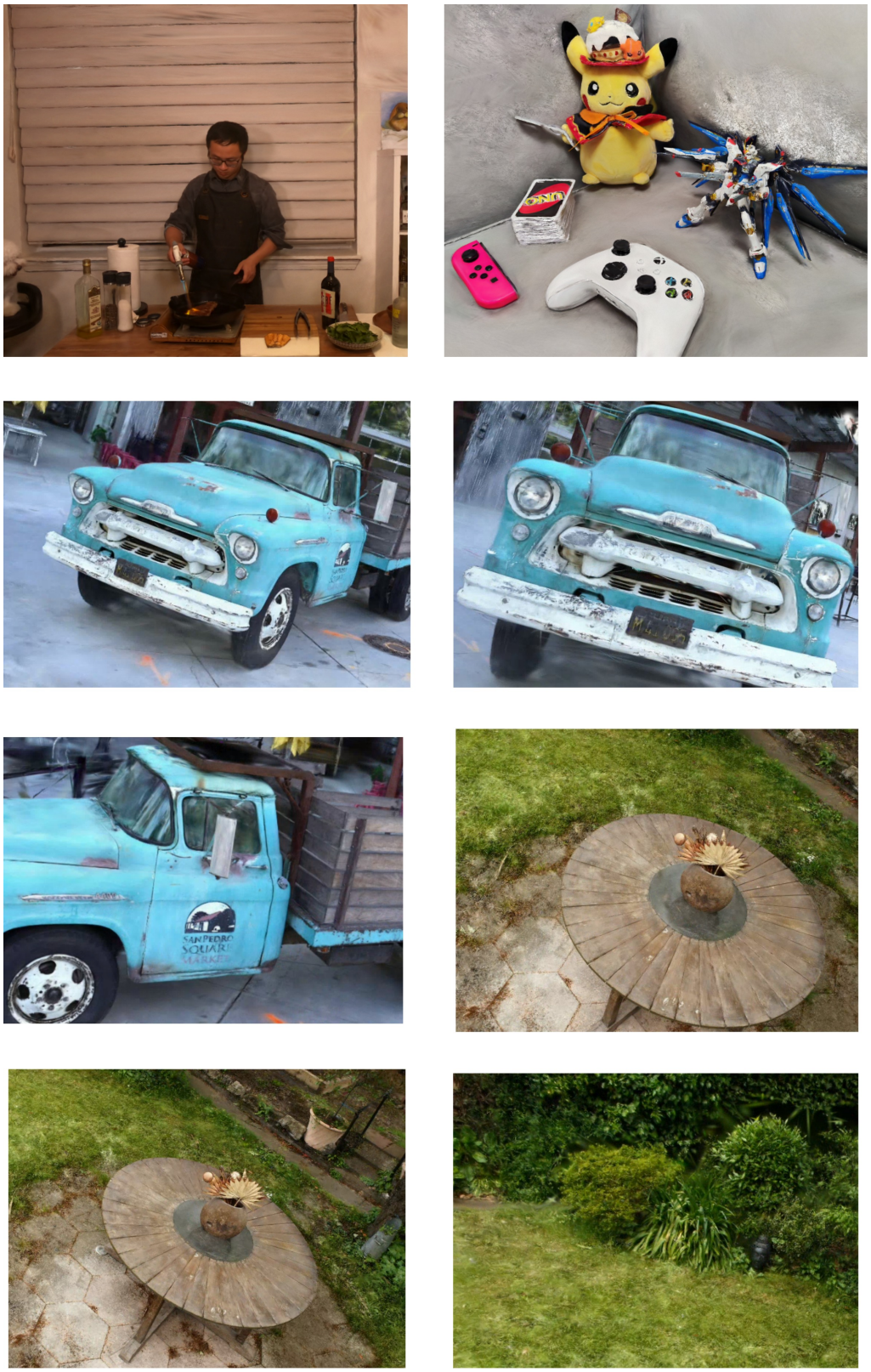

Figure 8. To show the effectiveness of proposed UVGS maps in capturing the intricacies of a complex real-world scene, we used a 12 layer UV map to reconstruct the 3D scenes.