Real-time Large-scale Deformation of Gaussian Splatting

LIN GAO*, Institute of Computing Technology, CAS and University of Chinese Academy of Sciences, China
JIE YANG, Institute of Computing Technology, CAS, China
BO-TAO ZHANG, Institute of Computing Technology, CAS and University of Chinese Academy of Sciences, China
JIA-MU SUN, Institute of Computing Technology, CAS and University of Chinese Academy of Sciences, China
YU-JIE YUAN, Institute of Computing Technology, CAS and University of Chinese Academy of Sciences, China
HONGBO FU, Hong Kong University of Science and Technology, China
YU-KUN LAI, Cardiff University, United Kingdom

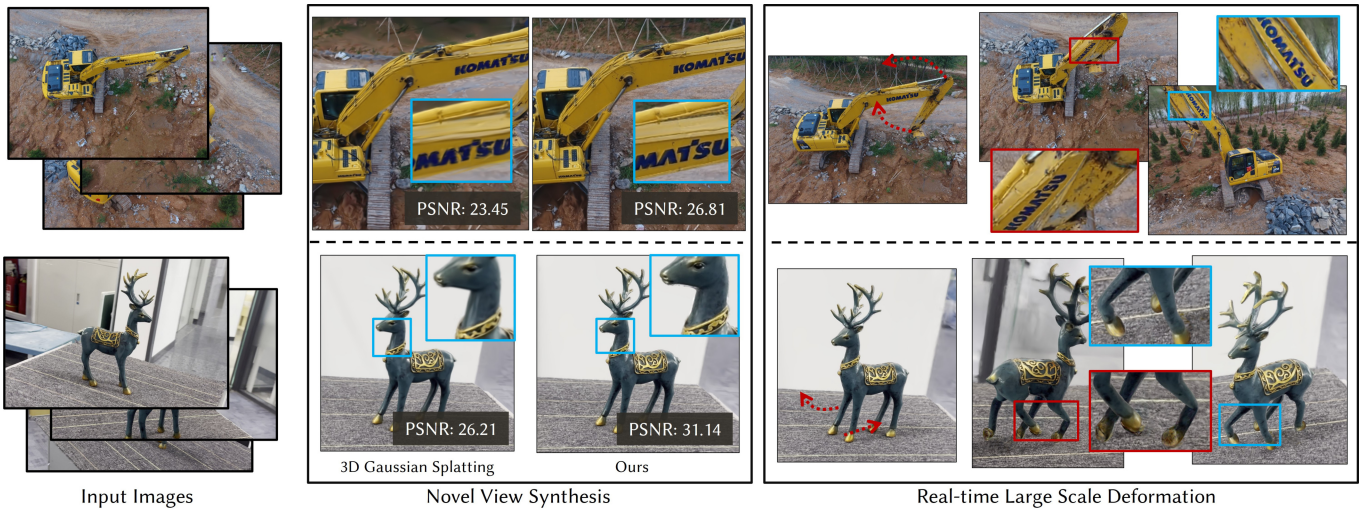

Fig. 1. Given a set of multi-view images of an object (left), we reconstruct the object with the proposed mesh-based Gaussian Splatting representation (GaussianMesh), involving both 3D Gaussians and an associated mesh. The mesh is adaptively refined along with Gaussian splitting, and also served as an effective regularization. As a result, our method achieves higher-quality novel view synthesis than 3D Gaussian splatting [Kerbl et al. 2023], evenly for large-scale deformation. Our 3D Gaussian deformation method produces high-quality deformation results in a real-time manner with large-scale deformations.

Neural implicit representations, including Neural Distance Fields and Neural Radiance Fields, have demonstrated significant capabilities for reconstructing surfaces with complicated geometry and topology, and generating novel views of a scene. Nevertheless, it is challenging for users to directly deform or manipulate these implicit representations with large deformations in a real-time fashion. Gaussian Splatting (GS) has recently become a promising method with explicit geometry for representing static scenes and facilitating high-quality and real-time synthesis of novel views. However, it cannot be easily deformed due to the use of discrete Gaussians and the lack of explicit topology. To address this, we develop a novel GS-based method (GaussianMesh) that enables interactive deformation. Our key idea is to design an innovative mesh-based GS representation, which is integrated into Gaussian learning and manipulation. 3D Gaussians are defined over an explicit mesh, and they are bound with each other: the rendering of 3D Gaussians guides the mesh face split for adaptive refinement, and the mesh face split directs the splitting of 3D Gaussians. Moreover, the explicit mesh constraints help regularize the Gaussian distribution, suppressing poor-quality Gaussians (*e.g.*, misaligned Gaussians, long-narrow shaped Gaussians), thus enhancing visual quality and reducing artifacts during deformation. Based on this representation, we further introduce a large-scale Gaussian deformation technique to enable deformable GS, which alters the parameters of 3D Gaussians according to the manipulation of the associated mesh. Our method benefits from existing mesh deformation datasets for more realistic data-driven Gaussian deformation. Extensive experiments show that our approach achieves high-quality reconstruction and effective deformation, while maintaining the promising rendering results at a high frame rate (65 FPS on average on a single commodity GPU).

*Corresponding author: Lin Gao (gaolin@ict.ac.cn).

Authors' addresses: Lin Gao, Jie Yang, Bo-Tao Zhang, Jia-Mu Sun and Yu-Jie Yuan are with Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences. Lin Gao, Bo-Tao Zhang, Jia-Mu Sun and Yu-Jie Yuan are also with University of Chinese Academy of Sciences. Hongbo Fu is with Division of Arts and Machine Creativity, Hong Kong University of Science and Technology. Yu-Kun Lai is with School of Computer Science and Informatics, Cardiff University.
Authors' e-mails: {gaolin, yangjie01, zhangbotao24s, sunjiamu21s, yuanyujie}@ict.ac.cn, fuplus@gmail.com, LaiY4@cardiff.ac.uk.

CCS Concepts: • **Computing methodologies** → **Image-based rendering**; *Mesh models.*

Additional Key Words and Phrases: 3D Gaussian Splatting, Deformation, Interactive, Data-Driven, Large-Scale

1 INTRODUCTION

Shape representations are fundamental in geometry processing. Traditionally, explicit representations like point clouds [Achlioptas et al. 2018; Fan et al. 2017; Qi et al. 2017a,b], voxels [Choy et al. 2016; Maturana and Scherer 2015; Wu et al. 2016], meshes [Groueix et al. 2018; Wang et al. 2018b,b] are commonly used in various contexts due to their intuitive and deformation-friendly property. In recent years, implicit representations (*e.g.*, Neural Radiance Fields (NeRFs) [Barron et al. 2022; Mildenhall et al. 2021], Signed Distance Fields (SDFs) [Chen and Zhang 2019; Chibane et al. 2020a; Mescheder et al. 2019]) have drawn much attention since they are able to reconstruct the highly realistic appearance and complicated geometry from only a few multi-view images. However, implicit representations also bear inherent drawbacks like slow rendering speed, limiting their applicability to practical applications. 3D Gaussian Splatting (3DGS) [Kerbl et al. 2023] is proposed to overcome these drawbacks while preserving the advantages. 3DGS learns the spatial distributions of Gaussian kernels from the initialization of SFM (Structure from Motion) [Ullman 1979] points, which naturally provide an explicit discrete 3D scene representation in contrast to the continuous representation used by NeRF. 3DGS has a small training cost and can achieve high-quality real-time rendering based on differentiable rasterization.

Since 3DGS is built upon discrete Gaussian kernels, it seems natural to directly perform deformation on them. However, simple deformation methods might produce suboptimal results. For example, SC-GS [Huang et al. 2024] learns sparse control points to model the dynamics of 3D scenes and enables deformation by manipulating the sparse points. Nevertheless, methods based on sparse control points struggle with complicated geometry or deformation due to the lack of topology prior. Simply deforming the Gaussians without any topology information (explicit mesh) generates strong misalignment artifacts when performing large-scale deformation, as shown in the 'Baseline' method in Figure 4. SuGaR [Guédon and Lepetit 2024] extracts an explicit mesh from a 3DGS representation by regularizing the Gaussians to distribute over the surface and thus enables the 3DGS editing by manipulating the extracted mesh. However, SuGaR only uses 3DGS to obtain the mesh and does not consider mesh properties like normals during Gaussian optimization. This might cause artifacts, especially when performing large-scale deformation. More recently, VR-GS [Jiang et al. 2024] learns a two-level embedding with an extracted tetrahedral mesh for a physical dynamic-aware interactive 3DGS system in Virtual Reality (VR), tackling the difficulty of editing high-fidelity virtual content in real-time. In contrast, our method allows the Gaussian kernels to offset from the explicit mesh to capture these features.

Motivated by the above observations, our proposed method enables high-quality real-time large-scale deformation on 3D Gaussian Splatting, as illustrated in Figure 1. Our core idea is to design an innovative mesh-based GS representation (GaussianMesh), which is integrated into Gaussian distribution learning and manipulation. In particular, given a mesh of a scene extracted by previous works[Schönberger et al. 2016; Wang et al. 2023a], we bind the mesh and a 3DGS representation with each other. We leverage this binding to provide guidance for both training and deformation of 3DGS in a novel manner. During the 3D-GS learning process, its Gaussian splitting follows two rules: 1. splitting along the face, which means the four split kernels remain on the surface according to one triangle is subdivided into four small faces, inspired by [Hu et al. 2022]; 2. splitting along the normal, which means the two split kernels are aligned along the surface normal. This establishes a one-to-one correspondence between triangular faces and Gaussians, allowing deformation gradients of the mesh to transfer effectively to rotation and scaling of the Gaussians during deformation. While some existing methods use explicit meshes as proxies to regularize Gaussians, they have cases where multiple faces correspond to a single Gaussian, leading to conflicts, especially during large-scale deformation and thereby limiting their ability to handle such deformations. This approach is more conducive to forming manipulations and enhancing the rendering effects (*e.g.*, high-frequency details) by eliminating irrational Gaussians (*e.g.*, misaligned Gaussians, long-narrow shaped Gaussians), which could cause artifacts.

Based on our proposed mesh-based GS, we introduce a large-scale Gaussian deformation technique to achieve deformable GS, which alters the parameters of 3D Gaussians according to the mesh deformation. In particular, we employ existing mesh deformation techniques [Gao et al. 2019a] on the mesh and apply the deformation gradients to the parameters of neighboring Gaussians. This process can be directly rendered by the splatting procedure in real-time. Our technique also supports intuitive data-driven deformation, which was only available in mesh-based methods before, thanks to the mesh-GS binding. Further, we utilize the segmentation to layer the Gaussians with foreground Gaussians and background Gaussians for in-the-wild scenes, thus enabling high-quality deformation of foreground objects in real scenes. Additionally, a regularization loss is introduced to enforce the spatial continuity and local rationality of Gaussian shapes, thus avoiding blurry visual artifacts due to the anisotropy of 3D Gaussian kernels for Gaussian deformation. Finally, we design an interactive tool, which supports real-time Gaussian manipulation and high-quality splatting while adhering to user-friendly constraints.

Extensive experiments and ablations on public datasets and self-captured scenes demonstrate that our mesh-based GS achieves better novel view synthesis compared with existing techniques while maintaining promising rendering speed (65 FPS on average on a single NVIDIA RTX 4090 GPU), and our method enables large-scale deformation of Gaussian splatting, outperforming existing methods.

Please refer to the accompanying video for real-time deformation results[1]. Our contributions can be summarized as follows:

- We propose a novel mesh-based GS (GaussianMesh) representation, which binds 3D Gaussians with the mesh representation and fully utilizes the mesh to guide the splitting of 3DGS, thus improving the quality of the learned GS.
- With the proposed GS representation, we introduce a large-scale Gaussian deformation method, which uses the vertex positions and deformation gradients to guide the GS. It takes advantage of mesh deformation methods while preserving real-time rendering and high-quality appearance robustly, even when deformed at a large scale.
- Extensive experiments demonstrate that our method achieves superior performance in terms of efficiency and quality of deformation compared to existing methods.

2 RELATED WORK

2.1 3D Shape Representations

Explicit Representations. Explicit representations dominated the industries and academic research for a long time. Classic representations, including point clouds [Fan et al. 2017; Qi et al. 2017a], voxels [Choy et al. 2016; Maturana and Scherer 2015; Wang et al. 2018a; Wu et al. 2016], and meshes [Gao et al. 2019b; Groueix et al. 2018; Hanocka et al. 2019; Wang et al. 2018b; Yang et al. 2022b], have been revisited for 3D deep learning. Although 3D explicit representations have a clear description of the geometry and appearance, they lack a flexible underlying topology representation and have limited capabilities of representing realistic appearance.

Implicit Representations. Different from explicit representations, implicit representations, including signed distance fields (SDFs) [Chen and Zhang 2019; Chibane et al. 2020a; Mescheder et al. 2019] and unsigned distance fields (UDFs) [Chibane et al. 2020b; Guillard et al. 2022; Liu et al. 2023a], can accurately model arbitrary geometry and topology. Thanks to the continuous nature of implicit representations, they can be combined with neural networks to support data-driven geometry learning.

In recent years, Neural Radiance Fields (NeRFs) have become increasingly popular as they allow for 3D optimization with only 2D supervision via volumetric rendering [Kajiya and Von Herzen 1984]. It has become prevalent in numerous tasks, such as 3D reconstruction [Li et al. 2023; Wang et al. 2021; Zhu et al. 2017], 3D generation [Liu et al. 2023b; Poole et al. 2023; Sun et al. 2024], and editing [Haque et al. 2023; Liu et al. 2021; Wang et al. 2023b]. Nevertheless, implicit approaches suffer from extensive sampling to fit the implicit functions of 3D scenes. This leads to significant computational costs, particularly in high-resolution or interactive rendering scenarios, even with accelerated NeRF versions [Fridovich-Keil et al. 2022; Müller et al. 2022; Yu et al. 2021]. It is thus difficult for NeRFs to achieve real-time rendering and high-quality view synthesis at the same time.

Gaussian Splatting. Recently, 3DGS has emerged as an explicit 3D representation, demonstrating remarkable rendering quality and high efficiency [Chen and Wang 2024; Fei et al. 2024; Wu et al. 2024b]. It can be used for 3D or 4D reconstruction [Luiten et al. 2024; Yang et al. 2024], avatar modeling [Kocabas et al. 2024; Li et al. 2024], etc. The GS-based generation has also gained significant attention [Chen et al. 2024; Tang et al. 2024; Yi et al. 2024]. 3DGS facilitates numerous applications thanks to its efficient differentiable rendering and high-fidelity rendering. Our proposed method is built upon 3DGS and inherits the rendering speed and the power to express the detailed appearance of 3DGS while additionally providing deformation ability, *e.g.*, data-driven deformation.

2.2 3D Shape Deformation

Mesh Deformation. Deforming a 3D model involves altering its explicit shape according to user-defined boundary conditions, while preserving the local geometry features. There are various ways to deform a 3D model for directly manipulating the mesh representation or embedded proxy, such as Laplacian coordinates [Lipman et al. 2005; Sorkine 2005], Poisson equations [Yu et al. 2004], data-driven mesh deformation [Gao et al. 2019a; Sumner et al. 2005], and cage-based approaches [Yifan et al. 2020; Zhang et al. 2020]. These methods can be conducted in real-time while preserving geometry details. These mesh deformation methods provide the basic ideas for the following novel shape representations.

NeRF Deformation&Editing. As a pioneering work for NeRF deformation and editing, EditNeRF [Liu et al. 2021] effectively modifies the shape and color of neural fields by including latent codes as conditioning factors. Several subsequent works [Bao et al. 2023; Gao et al. 2023; Wang et al. 2022, 2024] utilize the CLIP model [Wang et al. 2022] to facilitate NeRF editing via text prompts or reference images. Another stream is to use some predefined explicit proxies to support the deformation, such as skeletons for humans [Jiang et al. 2023b; Peng et al. 2021] and explicit geometries [Jambon et al. 2023; Xu and Harada 2022; Yang et al. 2022a; Yuan et al. 2022], which all transfer the deformations on explicit proxies to NeRFs. Besides, some 2D image manipulation (*e.g.*, inpainting, strokes) is also adopted into the NeRF editing via optimization schemes [Kobayashi et al. 2022; Liu et al. 2024b; Wang et al. 2023b; Zhuang et al. 2023] or applied to dynamic NeRFs [Pumarola et al. 2021] for 4D- editing [Jiang et al. 2023a].

Although it is innovative to introduce explicit proxies to enhance editing, these works have limited applicability due to the high computational cost and slow rendering speed of NeRFs.

Deformation of Gaussian Splatting. In contrast, 3DGS enables a relatively low training cost and high-quality real-time rendering via splatting rendering. The deformation on 3DGS has also been explored in various fields, including deformable Gaussians [Duisterhof et al. 2023; Kocabas et al. 2024; Yang et al. 2024], Gaussian avatars [Hu et al. 2024; Li et al. 2024; Zheng et al. 2024], and text-driven Gaussian editing [Palandra et al. 2024; Wu et al. 2024a; Zhuang et al. 2024]. PhysGaussian [Xie et al. 2024] employs discrete particle clouds from 3DGS for physically-based dynamics and photo-realistic rendering through continuum deformation [Bonet and Wood 1997] of Gaussian kernels. SC-GS [Huang et al. 2024] learns sparse control points for 3D scene dynamics but faces challenges with large-scale and complex deformation. SuGaR [Guédon and

[1]The source code is released at the link.

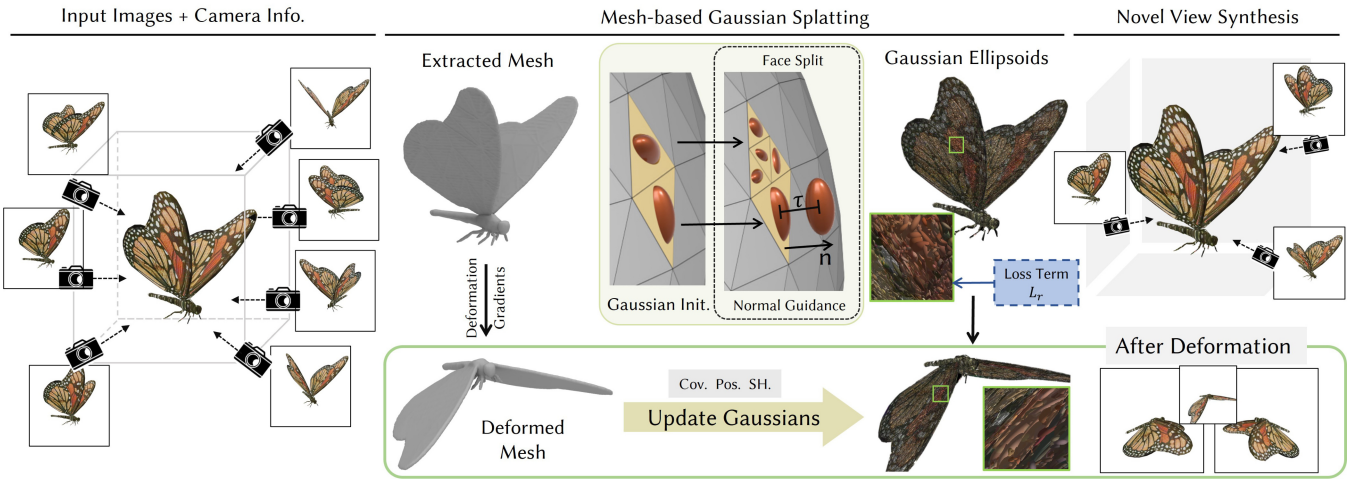

Fig. 2. **An overview of our pipeline.** Our mesh-based GS representation is specifically designed for Gaussian deformation. Given a set of calibrated images, we first reconstruct an explicit mesh and use this explicit geometry prior to initialize the Gaussians. During the learning phase, the explicit mesh guides the Gaussian learning through two strategies: a) Face Split; b) Normal Guidance. Additionally, we introduce a regularization loss L_r to constrain the scale of Gaussians, preventing the abnormally shaped Gaussians. During the deformation phase, user-controlled deformations on an explicit mesh (deformation gradients) drive the Gaussian parameters, producing deformed 3DGS for novel view rendering. Overall, our pipeline not only achieves accurate and realistic rendering from novel views but also supports effortless and real-time deformation of 3DGS.

Lepetit 2024] extracts explicit meshes from 3DGS representations but relies on simple adjustment of Gaussian parameters based on deformed meshes and thus struggles with large-scale deformation. Concurrent to our work, VR-GS [Jiang et al. 2024] introduces a two-level embedding with an extracted tetrahedral mesh for a physical dynamic-aware interactive 3DGS system in VR, tackling the difficulty of editing high-fidelity virtual content in real-time. The above approaches deform the original 3DGS by adjusting the parameters of Gaussian Splatting, and are limited with their effectiveness in large-scale deformation. Our observation is that 3DGS, which is based on discrete and unstructured Gaussian kernels, needs the thin shell geometric priors to organize the Gaussian kernels to perform large-scale deformation while preserving meaningful appearance.

Our work pioneers the adaptation of mesh-based deformation to 3DGS by harnessing the priors of explicit representations: the surface properties like normals of the mesh and the gradients generated by explicit deformation methods. The full utilization of the explicit mesh representation provides adequate geometric prior to 3DGS and improves both the reconstruction and deformation.

3 METHODOLOGY

Given a collection of multi-view images of a scene, we express its geometry and appearance as a Gaussian distribution bound with an explicit mesh extracted by existing approaches (*e.g.*, NeuS2 [Wang et al. 2023a] in our implementation). Our goal is to enable real-time and realistic deformation of 3DGS. To achieve this, we introduce an explicit mesh as the topological guidance and use mesh-based Gaussian distribution learning to constrain the parameters and the growth process of Gaussian functions, thus ensuring the correlation between the 3D Gaussians and the geometric shape. After Gaussian distribution learning, thanks to the binding of GS and the mesh, the deformation gradients from the user-controlled deformed mesh are applied to the Gaussians' parameters. In addition, a regularization of the Gaussian shape is designed to eliminate extreme anisotropy of the Gaussians in the deformation process. The integration of the above ideas leads to an interactive tool for real-time deformation and photorealistic rendering of novel views following the user's control. Figure 2 illustrates an overview of our pipeline. In the following subsections, we first introduce some preliminaries, including the 3DGS representation and mesh deformation in Sec. 3.1, and then present our mesh-based GS representation in Sec. 3.2, followed by the deformation technique in Sec. 3.3.

3.1 Preliminaries

3D Gaussian Splatting. 3DGS representation depicts 3D scene structures using distribution of 3D Gaussians. Each Gaussian element is defined by a center position $\mu \in \mathbb{R}^3$, a covariance matrix $\Sigma \in \mathbb{R}^7$, color $c \in \mathbb{R}^k$ (represented by spherical harmonic coefficients for view-dependent color, where k represents the degrees of freedom), and opacity $\alpha \in \mathbb{R}$. The Gaussian function $g(x)$ can be defined by the following formulation:

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

where the covariance matrix Σ can be factorized into a rotation matrix $\mathbf{R}$ expressed as a quaternion $q \in \mathbb{R}^4$ and a scaling matrix $\mathbf{S}$ represented by a 3D-vector $s \in \mathbb{R}^3$ for the differentiable optimization: $\Sigma = \mathbf{RSS}^T\mathbf{R}^T$.

The rendering technique of splatting, as initially introduced in [Zwicker et al. 2001], is to project the Gaussians onto the camera image planes, which are employed to generate novel view images. The location and covariance of the projected 2D Gaussians can be expressed as follows: $\Sigma' = JW\Sigma W^T J^T$ and $\mu' = JW\mu$,

involving a view transformation W and the Jacobian J of the affine approximation of the projective transformation. Specifically, the color of each Gaussian is assigned to every pixel based on the Gaussian representation described in Equ. 1. The opacity controls the influence of each Gaussian. The per-pixel color C is formulated as the weighted sum of N ordered Gaussians that are associated with the pixel: $C = \sum_{i \in N} SH(d_i, c_i)\sigma'_i \prod_{j=1}^{i-1}(1-\alpha_j)$, where $SH(\cdot,\cdot)$ is a spherical harmonic function with input direction d_i and SH coefficients c_i, and σ'_i represents the z-depth ordered effective opacity: $\sigma'_i = \sigma_i e^{-\frac{1}{2}(p-\mu'_i)^T \Sigma'_i (p-\mu'_i)}$.

Mesh Deformation. Given a triangular mesh $\mathcal{M}$, we can deform the mesh with respect to the user's control (*e.g.*, sparse control points) using mesh deformation methods [Gao et al. 2019a; Sorkine and Alexa 2007; Sumner et al. 2005]. For example, the approach by Gao et al. [2019a] can handle large-scale mesh deformations by optimizing the deformation gradients. They formulate the deformation by minimizing an energy function to ensure the stiffness of the mesh and prevent local distortions:

$$\mathrm{E}\left(\mathcal{M}'\right) = \sum_{i=1}^{N} \sum_{j \in \mathcal{N}(i)} w_{ij} \left\| \left(v'_i - v'_j\right) - \mathbf{T}_i \left(v_i - v_j\right) \right\|^2, \qquad (2)$$

where $\mathcal{N}(i)$ represents the set of 1-ring neighboring vertices of vertex i, v_i is the spatial coordinates of the i^{th} vertex on mesh $\mathcal{M}$, v'_i is the spatial coordinates of the corresponding vertex on the deformed mesh $\mathcal{M}'$, and w_{ij} is the cotangent weight [Sorkine and Alexa 2007], The transformation matrix $\mathbf{T}_i$ for each vertex i can be calculated by minimizing the above equation, and it can be decomposed into a rotation matrix $\bar{\mathbf{R}}_i$ and a shear matrix $\bar{\mathbf{S}}_i$ by polar decomposition. Both matrices will be applied to the corresponding Gaussians during the deformation procedure. Note that this deformation formulation also supports data-driven deformation if prior exemplar deformed meshes are available. The $\mathbf{T}_i$ could be optimized by blending the existing deformation gradients from the exemplar meshes.

3.2 Mesh-based Gaussian Splatting

While the 3DGS can produce realistic rendered images in real-time, it struggles to accurately represent the details and topological structure of a 3D scene. This limitation arises from its reliance on discrete Gaussian kernels, particularly when it comes to deformation. Since the discrete Gaussian kernels do not employ the connections between the Gaussians, it is hard to achieve feasible deformation for the vanilla 3DGS. In order to tackle this problem, we introduce our mesh-based GS (GaussianMesh). Our method focuses on integrating 3D Gaussian kernels with specified mesh surfaces, enhancing the Gaussian deformation process.

We use a reconstructed mesh $\mathcal{M}$, obtained using an existing efficient method [Wang et al. 2023a] as an explicit prior constraint. A naïve approach involves deforming this explicit prior and driving the Gaussians learned by vanilla 3DGS using the projection on the explicit meshes. However, this method often results in artifacts (*e.g.*, blurry and misalignment of Gaussians with explicit priors in Figure 4 Baseline), especially for large-scale deformation. To address this issue, we propose two strategies to regulate the Gaussian parameters and the growth of Gaussian kernels, as illustrated in Figure 2. These strategies ensure the correlation between 3D Gaussians and the explicit prior. They aim to regularize the 3D Gaussian kernels while maintaining their ability to accurately represent geometric and textural features. Specifically, we initialize a Gaussian by anchoring it precisely at the centroid of every triangular face on the mesh surface. During the training of mesh-based GS, different from the vanilla 3DGS, we allow the division of Gaussians by utilizing the following strategies:

- Face Split: A single triangle is subdivided into four smaller triangles over the surface by inserting a new vertex at the midpoint of each edge. The Gaussian kernels are also split correspondingly, and the position of each Gaussian is initialized with barycentric coordinates $w = (w_a, w_b, w_c)$ of new subdivided faces.
- Normal Guidance: Each Gaussian has a perpendicular movement to the surface under normal guidance. The distance of this movement τ is learnable.

The distribution of 3D Gaussians is determined by the two strategies mentioned above, which incorporate explicit mesh priors. The first strategy aims to guarantee a sufficient number of Gaussian kernels to accurately represent the visual appearance of the 3D scene, following the guidance of the mesh surface. The second strategy helps represent the fine-grained texture details of a 3D scene for Novel View Synthesis (NVS). Both mandate the distribution of Gaussian kernels near the explicit surface beforehand. Note that the Gaussian removal operation follows the original 3DGS, but we ensure at least one Gaussian for each mesh face during the training process (although it can be transparent with zero opacity)

During training,the barycentric coordinates $w = (w_a, w_b, w_c)$ and the offset distance τ are parameterized into additional attributes for 3DGS learning. The barycentric coordinates (w_a, w_b, w_c) represent the weights assigned to three vertices $(\mathbf{v}_a, \mathbf{v}_b, \mathbf{v}_c)$ belonging to the attached triangle,and $\tau \in [-0.5, 0.5]$ is a ratio that is multiplied with the radius R of the circumcircle of the nearby triangle to control the displacement along the surface normal $\mathbf{n}$. To summarize, the spatial position μ of the Gaussian kernel is formulated as:

$$\mu = (w_a \mathbf{v}_a + w_b \mathbf{v}_b + w_c \mathbf{v}_c) + \tau R \mathbf{n}, \qquad (3)$$

By utilizing the prior of explicit meshes, we employ the aforementioned strategies to regularize the density of Gaussians according to the explicit surface and generate new Gaussian kernels that are initialized to (1/3,1/3,1/3) for barycentric coordinates w and inherit offsets τ from the Gaussian before splitting, followed by Equ. 3. The new Gaussians continue participating in the optimization. Currently, each new Gaussian is attached to a subdivided face and split according to its attached face split for the next iteration.

Regularization L_r. Under the guidance of the above-proposed strategies for 3DGS training, there might exist long-narrow-shaped Gaussians to capture the high-frequency details. However, the long-narrow-shaped Gaussians will not align to the surface, thus resulting in the artifacts (refer to Figure 3) when performing the large-scale deformation. For better visual quality of the deformed 3DGS, we introduce regularization to ensure the Gaussians' spatial coherence and local consistency (see the illustration in Figure 3). Additionally, to support arbitrary deformation, local meshes inevitably undergo

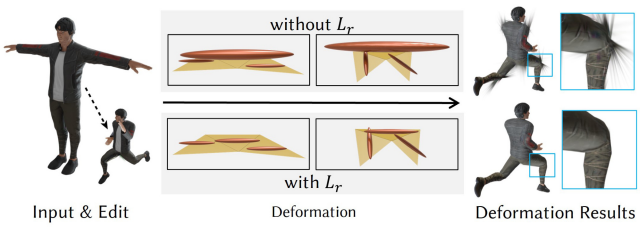

Fig. 3. **Regularization L_r.** To reduce the artifacts during large-scale deformation, we propose regularization during 3D Gaussian training. The regularization aims to limit the scale of each Gaussian kernel within its attached triangle. If we disable the regularization, it leads to some artifact (first row) when performing the deformation on 3D Gaussian.

drastic changes with large-scale deformation. It will lead to visual artifacts in Figure 3 when the learned Gaussian shape is large enough to cover multiple triangles on the surface. To ensure plausible deformation results and capture high-frequency textures, we employ a regularization term L_r into the training process. This term adjusts the Gaussian shape based on the size of neighboring triangles, ensuring the appropriate Gaussian is learned and the local continuity is preserved during deformation. Specifically, the regularization L_r is formulated as:

$$L_r = \sum_{g \in \mathcal{G}} \max\left(\max\left(s_i\right) - \gamma R_i, 0\right), \tag{4}$$

where g is a member of Gaussian kernel set $\mathcal{G}$, s_i is the 3D scaling vector of each Gaussian, R_i is the radius of the circumcircle of the triangle where the Gaussian is attached and $\gamma = 3$ is a hyper-parameter to control the influence on the size of the Gaussian from its attached triangles. All these techniques are applied during Gaussian optimization. After training, the Gaussians can be driven directly by the mesh deformation without any optimization, and rendered from novel views.

3.3 Editable Gaussian with Mesh Deformation

By utilizing existing mesh deformation methods and the efficient differentiable rasterization of GS, it is possible to achieve real-time deformation of Gaussians based on the GS representation proposed in Sec. 3.2. To illustrate this idea, we employ the mesh deformation techniques and formulations discussed in Sec. 3.1. The user can manipulate 3D Gaussians using various controls, such as non-rigid deformation, translation, rotation, etc. Equ. 2 states that each vertex v_i in the deformed mesh $\mathcal{M}$ is linked to a transformation matrix T_i, which represents local changes around vertex v_i between the deformed mesh $\mathcal{M}'$ and the original mesh $\mathcal{M}$. The matrix T_i can be decomposed into a rotation matrix $\bar{\mathbf{R}}_i$ and a shear matrix $\bar{\mathbf{S}}_i$ using the polar decomposition. It is easy to prove that the Gaussian distribution remains unchanged following an affine transformation. Thus, we can easily apply the rotation matrix $\bar{\mathbf{R}}_i$ and shear matrix $\bar{\mathbf{S}}_i$ from the deformed mesh to its associated Gaussian kernels, as well as the displacement of the deformed mesh faces.

Thanks to the explicit meshes, our mesh-based GS deformation can achieve complex and reasonable deformation with data priors, such as the deformation of cloth and articulated shapes. To achieve the complex data-driven deformations, we begin by extracting principal deformation bases from external data sources, such as physical simulations or pre-existing animations. These bases capture the essential patterns of shape variation and movement. Next, we apply these deformation bases to the explicit mesh representation, allowing us to perform data-driven deformations that are both realistic and complex. This step ensures that our deformations are grounded in physically plausible or artistically pleasing motion patterns. Following this, we establish a shape-to-shape correspondence between the deformed explicit mesh and the shape reconstructed from multi-view images. This correspondence mapping is crucial, as it allows us to transfer the vertex-wise transformations from the deformed mesh to the reconstructed shape. This transfer process ensures that the deformations learned from our data priors can be accurately applied to the specific geometry captured in the multi-view images. Finally, we use the deformed meshes, now informed by both the data priors and the specific geometry of our reconstructed shape, to drive the deformation of the Gaussians. This approach allows the Gaussian representation to inherit the complex, data-driven deformations while maintaining the benefits of its implicit nature. Hence, we can easily apply the desired deformation to the 3DGS via the user's controls. This pipeline effectively bridges the gap between explicit mesh deformations derived from rich data priors and the flexible, efficient Gaussian representation. It enables us to achieve highly detailed and realistic deformations of 3DGS via the user's control, respecting both the learned patterns and the specific geometry and appearance of the captured object.

Each deformed Gaussian g' is bound with a triangle $f' = (\mathbf{v}'_a, \mathbf{v}'_b, \mathbf{v}'_c)$ with three deformed vertices. The relative displacement ΔP and deformation gradient T_i for the deformed face can be expressed using the barycentric coordinates (w_a, w_b, w_c):

$$\begin{aligned} \Delta P &= w_a(\mathbf{v}'_a - \mathbf{v}_a) + w_b(\mathbf{v}'_b - \mathbf{v}_b) + w_c(\mathbf{v}'_c + \mathbf{v}_c), \\ \bar{\mathbf{R}}_i &= w_a log(\bar{\mathbf{R}}_{\mathbf{v}'_a}) + w_b log(\bar{\mathbf{R}}_{\mathbf{v}'_b}) + w_c log(\bar{\mathbf{R}}_{\mathbf{v}'_c}), \\ \bar{\mathbf{S}}_i &= w_a \bar{\mathbf{S}}_{\mathbf{v}'_a} + w_b \bar{\mathbf{S}}_{\mathbf{v}'_b} + w_c \bar{\mathbf{S}}_{\mathbf{v}'_c}, T_i = exp(\bar{\mathbf{R}}_i)\bar{\mathbf{S}}_i. \end{aligned} \tag{5}$$

Following the above equations, we can get the transformed Gaussian kernels with position $\mu' = \mu + \Delta P$ and covariance matrix $\Sigma' = T_i \Sigma T_i^\top$. The deformed Gaussian kernel is

$$g'(x) = e^{-\frac{1}{2}(x-(\mu+\Delta P))^T (T_i \Sigma T_i^\top)^{-1}(x-(\mu+\Delta P))}. \tag{6}$$

In addition, 3DGS employs spherical harmonics to express color, wherein a given Gaussian exhibits varying colors when viewed from different angles, enabling the modeling of view-dependent appearance. Hence, for a deformed Gaussian kernel g', it is necessary to adjust the orientation of spherical harmonics [Guédon and Lepetit 2024] by applying the inverse of the local rotation matrix $exp(\bar{\mathbf{R}}_i)$ from the deformed mesh to the view direction d_i, *i.e.*, $SH(exp(\bar{\mathbf{R}}_i)^T d_i, c_i)$. In conclusion, our mesh-based GS representation allows flexible manipulation of Gaussians through mesh deformation and high-fidelity rendering of the results in novel views.

Real-Time Interactive Tool. We integrate the above techniques into an interactive deformation tool that allows for the real-time deformation of 3DGS with respect to the user's controls. Users can

utilize the triangle mesh as a proxy to accomplish real-time large-scale deformation of 3DGS and produce high-fidelity rendering results.

4 EXPERIMENTS & EVALUATIONS

In this section, we conduct a series of qualitative and quantitative experiments to evaluate the effectiveness of our approach, including comparison with existing techniques, deformation results (with or without data priors) on both synthetic and real-captured datasets, and ablation studies to analyze the impact of our main design choices.

4.1 Datasets and Metrics

To validate the effectiveness of our method, we performed comprehensive experiments on the widely used NeRF-Synthetic [Pumarola et al. 2021] dataset, synthetic data from SketchFab [Pinson 2011] (butterfly, Dress, Jeans, Sofa, Digital Human, Banana, Giraffe), in-the-wild scenes from BlendedMVS [Yao et al. 2020] and Tanks and Temples [Knapitsch et al. 2017], as well as the real-world scenes captured by ourselves.

The NeRF-Synthetic dataset contains eight static scenes with 360° random viewpoint settings with the known camera poses. For synthetic data from SketchFab, we use Blender to generate training data with the identical configuration as NeRF-Synthetic. The BlendedMVS dataset contains a lot of in-the-wild scenes with camera poses. Tanks and Temples contain both training data and testing data for indoor scenes and large outdoor scenes with complex geometric layouts and camera trajectories. The self-captured data is captured by a mobile phone (an iPhone 11) with its camera poses calibrated by COLMAP [Schonberger and Frahm 2016; Schönberger et al. 2016]. All datasets involve a diverse range of visual appearances and geometric properties of various deformable objects. For the public datasets, we use the default training and testing sets splits in all our experiments. In order to evaluate the effectiveness of our approach, we use three metrics to measure the quality of novel view synthesis, including Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM) [Wang et al. 2004], and Learned Perceptual Image Patch Similarity (LPIPS) [Zhang et al. 2018].

Implementation Details. The pipeline comprises two main components: Gaussian distribution learning and real-time 3DGS deformation via an intuitive and friendly 'drag' way. Our code is implemented based on the 3D-GS [Kerbl et al. 2023] and Drag Your GAN [Pan et al. 2023], incorporating effective training and rendering capabilities. The explicit mesh can be easily extracted by NeuS2 [Wang et al. 2023a] or created by artists (*i.e.*, synthetic data). It takes around 20 minutes to train the Gaussian distribution under our proposed mesh-based GS (please see more details in the supplemental materials). Similarly to the explicit mesh, the mesh sequences for the data-driven deformation of 3DGS are obtained from physical simulation or designed by artists. All experiments are performed on a PC equipped with an i9-12900K CPU and an RTX 4090 graphics card. It is important to mention that our pipeline solely focuses on optimizing the Gaussian parameters without any involvement of network parameters.

Table 1. **Comparisons of novel view synthesis on NeRF-Synthetic dataset.** We performed the quantitative comparison with four alternative methods on the novel view synthesis task. Four baselines (NeRF-Editing, 3D-GS, SuGaR, and GaMeS) are evaluated in three metrics, *i.e.*, PSNR, SSIM, and LPIPS. Our method achieves the best in PSNR and SSIM, and reaches comparable performance on the LPIPS with SuGaR. It demonstrates that our method can synthesize high-fidelity renderings on 3DGS and supports arbitrary deformation edits on 3DGS.

Methods	NeRF-Editing	3D-GS	SuGaR	GaMeS	Ours
PSNR↑	30.58	33.31	32.39	33.01	33.84
SSIM↑	0.960	0.967	0.958	0.966	0.974
LPIPS↓	0.058	0.023	0.037	0.030	0.030

4.2 Comparisons & Evaluations

We evaluate the performance of our approach and the comparison methods on the datasets discussed in Sec. 4.1, both qualitatively and quantitatively. We focus on two tasks: novel view synthesis and 3DGS deformation. We consider six state-of-the-art (SoTA) approaches: NeRF-Editing [Yuan et al. 2022], SuGaR [Guédon and Lepetit 2024], 3D-GS [Kerbl et al. 2023], GaMeS [Waczyńska et al. 2024], VR-GS [Jiang et al. 2024], and SC-GS [Huang et al. 2024]. 3D-GS is a baseline for novel view synthesis ,and NeRF-Editing is a classical method for NeRF-based editing. Other comparison works are 3DGS-based methods that support deformation or editing.

First, we evaluate their performance of the novel view synthesis task on the NeRF-Synthetic dataset. We follow the same evaluation setting in Sec. 4.1 to measure the performance on the test dataset. Table 1 reports the results on the novel view synthesis task and clearly shows that our technique outperforms the four baselines in PSNR and SSIM with similar Gaussian kernels (please refer to the supplementary materials for details), while reaching comparable performance in LPIPS. The results reveal that our approach has the ability to generate realistic results from novel views.

Additionally, we evaluate the performance of 3DGS deformation. Since there is no ground-truth deformation for real objects, we show the visual results of novel view qualitatively. We perform the experiments on a total of four cases, comprising both synthetic data and real-captured data. In the supplementary materials, we additionally evaluate the deformation quality on synthetic data quantitatively under the aforementioned three metrics. Four alternative methods are evaluated, including NeRF-Editing, SuGaR, GaMeS, and a baseline based on 3D-GS. The baseline is an extension of the 3D-GS without incorporating our essential designs (Face Split, Normal Guidance, and regularization L_r), only constraining the Gaussians to align with the reconstructed surface (extracted by NeuS2) since there is an explicit surface as guidance.

Figure 4 presents four deformation results by our method through the modification of the explicit mesh. NeRF-Editing deforms NeRF by utilizing a derived tetrahedron mesh, which is time-consuming and cannot handle large deformations. This often results in blurry rendering outcomes, particularly for high-frequency details. The baseline is only to attach the Gaussians to the NeRF-based surface without incorporating any special designs. Consequently, this approach leads to certain irregularly shaped Gaussian elements, resulting in artifacts when significant deformation and high-frequency

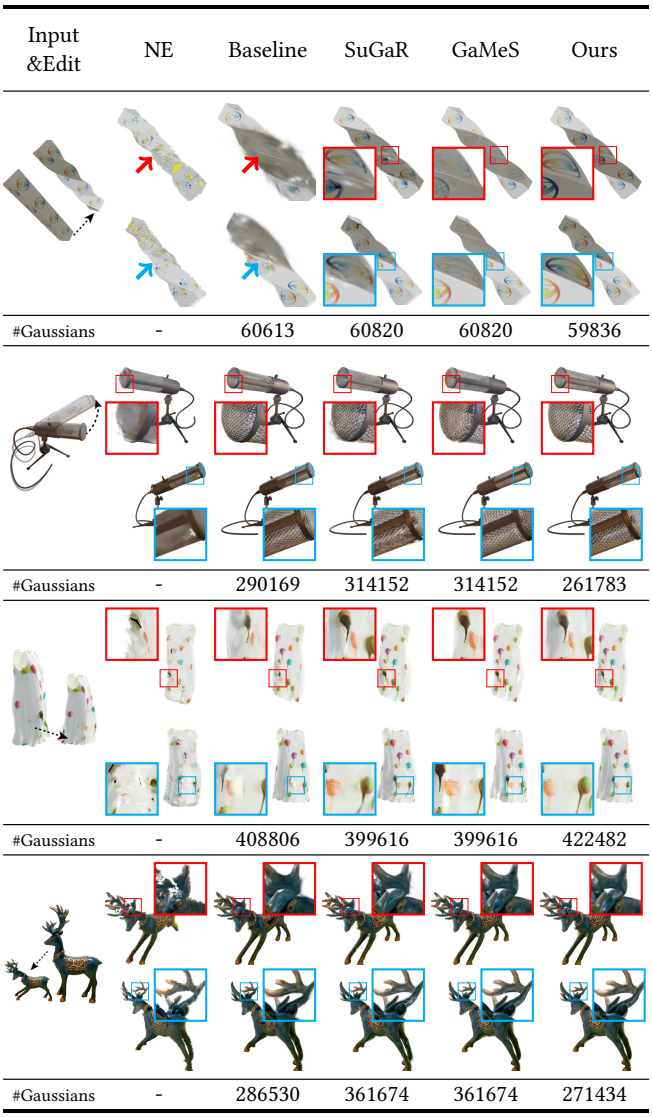

Fig. 4. **Deformation comparison between our method and the alternative methods.** There are Mic from NeRF-Synthetic dataset, and Cubiod, Dress, and Deer captured by ourselves. We have highlighted the difference with different color boxes for different views and listed the numbers of Gaussians for all 3D-GS-based methods. The results show that our method successfully preserves the high-frequency details after large-scale deformation under similar numbers of Gaussians.

features appear. Although SuGaR successfully reconstructs the mesh from the 3D GS and realizes the deformation by adjusting the parameters of Gaussians, it fails to capture fine features at high frequencies under significant deformations (*e.g.* , first row in Fig. 4). GaMeS also utilizes the explicit mesh and binds some Gaussians to each triangle, not taking the face slice and split operations during training, still resulting in some artifacts with large-scale deformation. In contrast, our approach effectively models a better Gaussian distribution using explicit mesh guidance, achieving enhanced

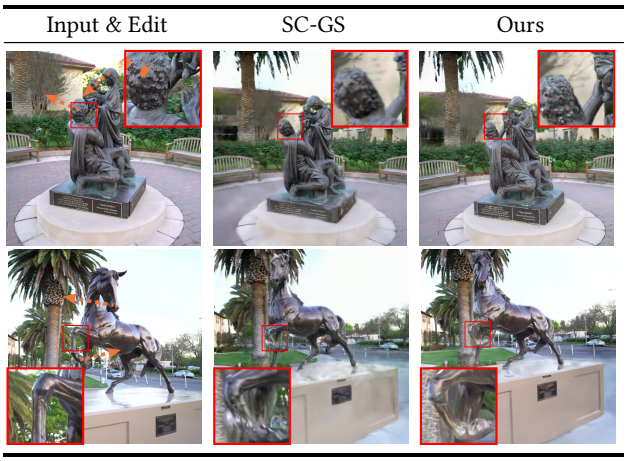

Fig. 5. **Comparison with SC-GS [Huang et al. 2024].** For two real examples from the Tanks and Temples [Knapitsch et al. 2017], we perform similar deformations to SC-GS (displayed in the 1st column) and render them from the same view. From the results, it is clear that our method can better preserve the high-frequency details (highlighted with colored box) under the same deformation.

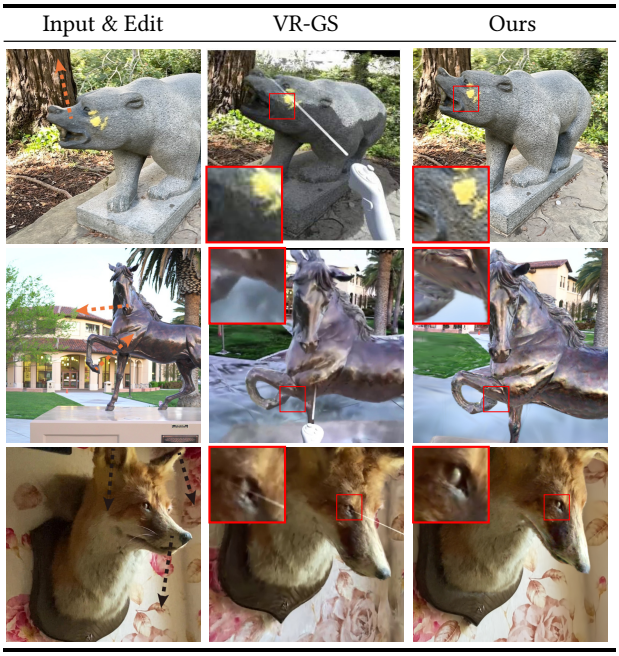

Fig. 6. **Comparison with VR-GS [Jiang et al. 2024].** There are three real examples (Fox, Bear, and Horse) from the Instant-NGP dataset [Müller et al. 2022], the Instruct-NeRF2NeRF dataset [Haque et al. 2023], the Tanks and Temples [Knapitsch et al. 2017]. We only perform similar deformations to VR-GS in the first column and render them according to the presented figures in their ArXiv paper since their code is not available yet. The results show that our method can render the higher quality of the high-frequency details, especially for large-scale deformations.

rendering quality and real-time deforming performance at a frame rate of 65 FPS. Please refer to the accompanying video for more deformation results.

Further, we evaluate the performance on the large-scale in-the-wild scenes and compare our method with VR-GS [Jiang et al. 2024] and SC-GS [Huang et al. 2024]. For such scenes, we segment the foreground and background via the tool [Cheng et al. 2024] and learn separate Gaussians for them: foreground Gaussian learning via our mesh-based GS representation and background Gaussian learning via vanilla 3DGS. Hence, we can deform only the foreground object and keep the background unchanged, with high visual quality. To compare with VR-GS, we perform a similar deformation and render using similar views according to the presented figures in their ArXiv paper since their code is not available yet. Figure 5 and Figure 6 show visual comparison results on four real examples (Family, Fox, Bear, and Horse) from Instant-NGP dataset [Müller et al. 2022], the Instruct-NeRF2NeRF dataset [Haque et al. 2023] the Tanks and Temples [Knapitsch et al. 2017]. From the results, we can see that our method can capture high-frequency details, such as the eye on the fox. Further, our method can easily achieve complex deformation thanks to explicit mesh when we introduce the mesh-based sequences as the data priors.

4.3 More Deformation Results

Our technique enables an interactive tool that allows for real-time deformation by interactively manipulating the sparse control points. The control points are specified by users via the click operation in our GUI interface. The learned Gaussian distribution is effectively guided by our mesh-based GS and allows for excellent generalization, even in the face of challenging deformations. In order to verify the generalizability of our proposed deformation pipeline, we present more deformation examples in Figure 7, involving various deformations of synthetic and real-captured examples. Please refer to the supplementary video for real-time deformation results.

Thanks to the explicit meshes, our technique can use data priors to achieve complex deformations of objects, such as cloth and articulated shapes. Figure 8 demonstrates that our method can easily deform the Gaussians that satisfy the data distributions. We introduce the data prior from the physical simulation for the deforming dress and animated shapes for the running deer to drive the reconstructed shapes from multi-view images according to the manually specified shape-to-shape correspondence. Finally, the desired deformation can be easily applied to the 3DGS via user's controls.

4.4 Ablation Studies

In this subsection, we evaluate the crucial designs in our representation and deforming pipeline by conducting the ablations. The crucial designs influence the performance of Gaussian optimization. Specifically, we will evaluate the influence on 3D Gaussian deformation with different qualities of the explicit mesh, the Face Split operation, the utilization of Normal Guidance for Gaussian distribution learning, and the regularization L_r for 3D GS deformation. Further, we also validate the effectiveness of our representation, compared to the vanilla Mesh+TextureMap representation.

Table 2. **Quantitative evaluation of our key designs on the NeRF-Synthetic dataset.** Three commonly used metrics (*i.e.*, PSNR, SSIM, and LPIPS) are reported. The results clearly demonstrate that our full method reaches the best performance, indicating that our key designs significantly contribute to our overall pipeline. Note that *F.S.* means Face Split, and *N.G.* means Normal Guidance.

Methods	PSNR↑			SSIM↑			LPIPS↓		
	w/o *F.S.*	w/o *N.G.*	Ours	w/o *F.S.*	w/o *N.G.*	Ours	w/o *F.S.*	w/o *N.G.*	Ours
Lego	35.61	33.48	**36.20**	0.977	0.978	**0.985**	0.020	0.020	**0.013**
Ficus	34.98	33.74	**36.05**	0.991	0.990	**0.993**	0.014	0.016	**0.014**
Drums	27.27	27.32	**27.85**	0.962	0.959	**0.963**	0.046	0.048	**0.044**
Chair	34.76	33.53	**35.76**	0.981	0.975	**0.986**	0.017	0.020	**0.014**
Hotdog	38.41	36.42	**38.45**	0.963	0.962	**0.988**	0.018	0.021	**0.017**
Mic	33.59	33.02	**34.76**	0.990	0.988	**0.991**	0.008	0.009	**0.008**
Material	30.95	30.50	**31.21**	0.965	0.963	**0.978**	0.027	0.029	**0.023**
Ship	29.47	29.36	**30.47**	0.894	0.894	**0.910**	0.118	0.123	**0.110**
Average	33.13	32.17	**33.84**	0.965	0.964	**0.974**	0.033	0.036	**0.030**

Explicit Mesh with Different Qualities. The explicit mesh plays an important role in our deformable mesh-based GS representation. To verify the robustness and reliance of different qualities of the mesh, we have conducted two kinds of experiments, including the different mesh extraction approaches and different mesh qualities (*i.e.*, adding noise, remeshing, smoothing, simplification). Two examples are validated for each evaluation. We reconstruct the explicit meshes under the same setting for different ablated versions. Then, we train our network on the different meshes and deform them with the same control handles. Finally, we compare the visual quality of deformation results from novel views.

For the different mesh extraction approaches, we have evaluated commonly used approaches for mesh extraction from multi-view images, including the traditional multi-view stereo reconstruction (COLMAP [Schonberger and Frahm 2016; Schönberger et al. 2016]), NeRF-based reconstruction method (NeuS2 [Wang et al. 2023a]), and 3DGS-based reconstruction method (SuGaR [Guédon and Lepetit 2024]). For a fair evaluation, we ensure that the number of vertices is the same and enough for all extracted meshes via mesh simplification [Garland and Heckbert 1997]. Two outdoor scenes from the BlendedMVS are evaluated to demonstrate the robustness on the different mesh extractions. The results in Figure 9 show our method can successfully achieve high-fidelity deformation results in large-scale deformation and keep similar visual rendering quality for different mesh extraction approaches.

Due to COLMAP's longer running time compared to NeuS2 and SuGaR's difficulty in reconstructing too thin geometries, we use NeuS2 for mesh extraction by default. We utilized mesh operations to change the mesh quality, including remeshing, smoothing, simplification, and adding noise. For adding noise, we respectively add different scales of Gaussian noise in the reconstructed meshes, ranging from 0.01, 0.02, 0.05, and 0.1 times along the vertex normal. For smoothing, we apply mesh smoothing operations (Laplacian mesh smoothing [Sorkine 2005]) at different scales separately and maintain the number of vertices the same. For the remeshing, we perform remeshing operations [Alliez et al. 2003]on the meshes separately to ensure that the topology of the meshes is changed but the numbers of vertices are approximately the same. For the different numbers of faces, we have simplified the reconstructed

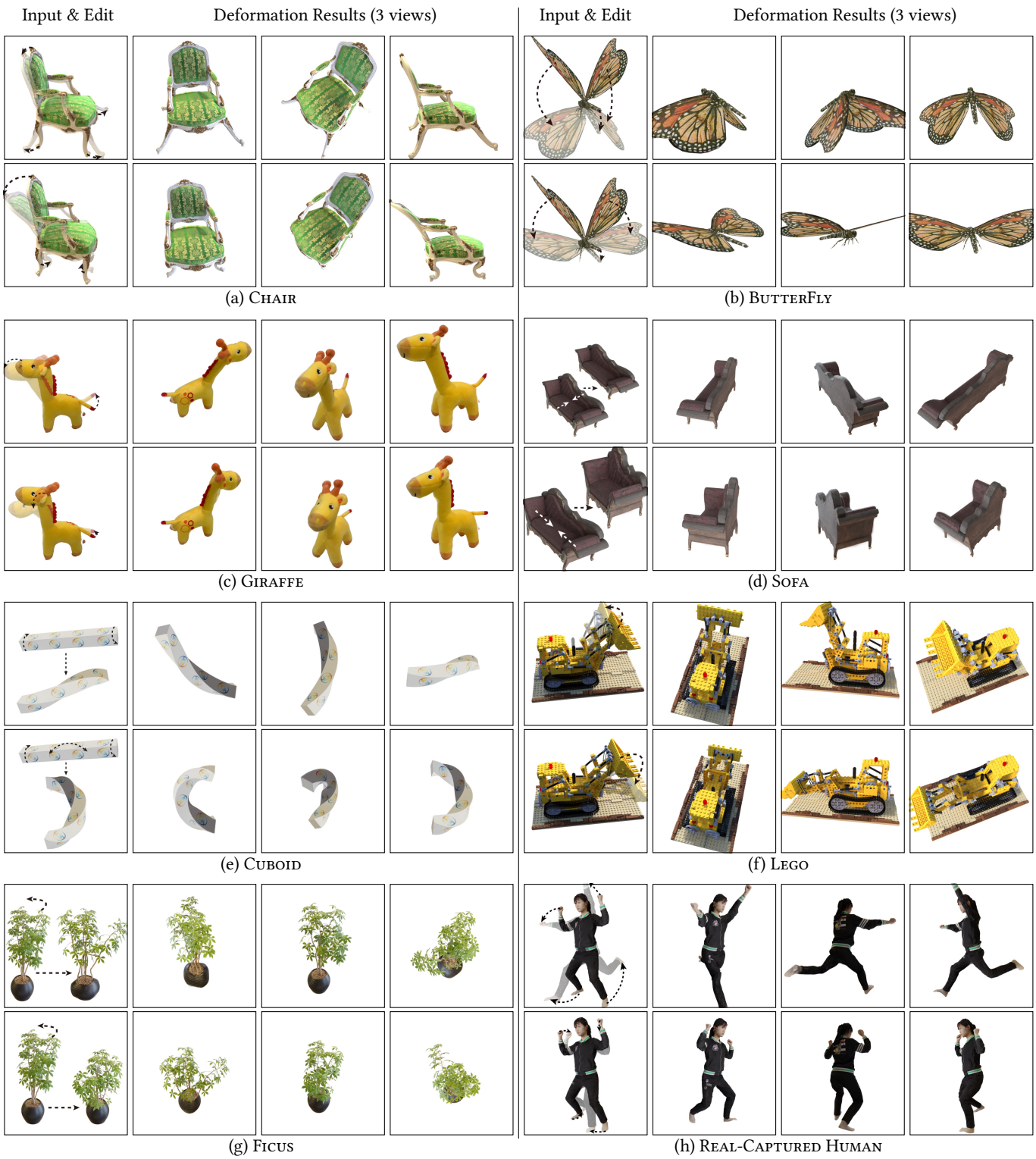

Fig. 7. **Various 3D GS deformation results by our method.** The examples include ButterFly, Sofa, Cuboid from SketchFab [Pinson 2011], Giraffe captured by ourselves, Lego, Ficus, Chair from the NeRF-Synthetic dataset [Pumarola et al. 2021], and Real-Captured Human from the THuman3.0 Dataset [Su et al. 2023]. Each example consists of 2 edits. It is clearly shown that our results are realistic and high-fidelity from novel view rendering.

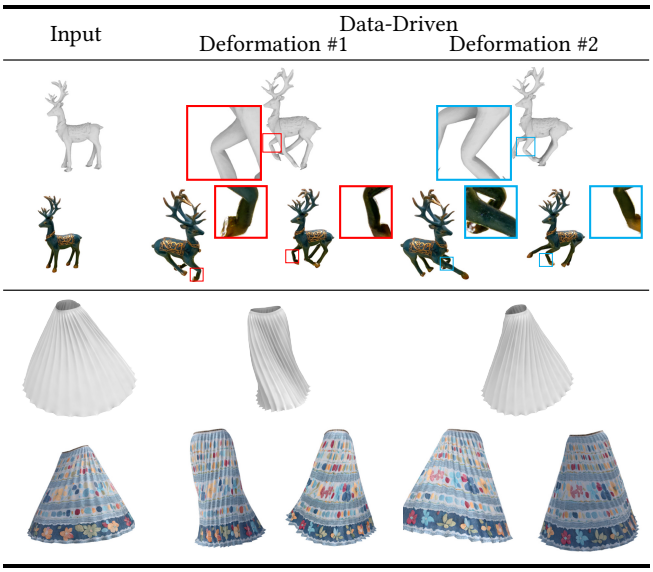

Fig. 8. **Data-driven deformation results.** We perform the data-driven deformation of 3DGS on two examples, including a synthetic data DRESS and a real-captured data DEER. Here, it presents two data-driven deformations and renders the deformation results from two views. The results show that our method can achieve the complex deformation of 3DGS under the guidance of data priors from physical simulation or other animated sequences.

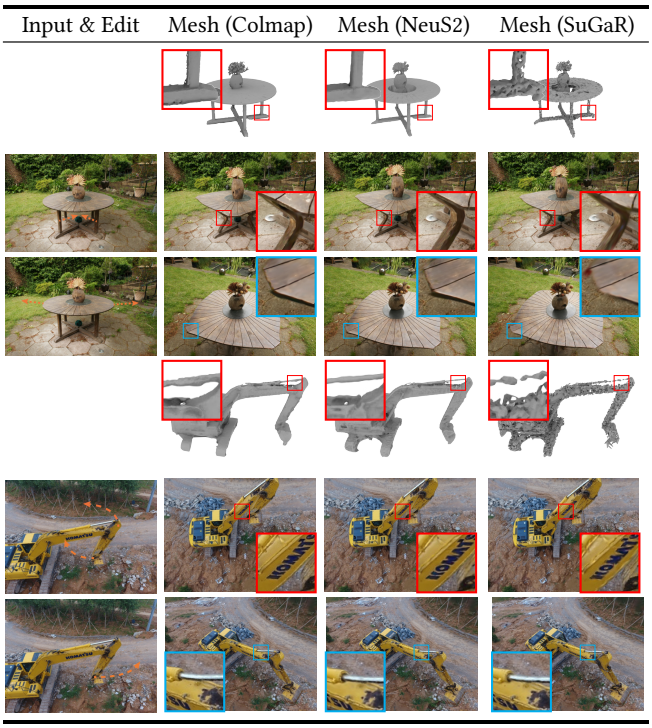

Fig. 9. **Robustness validation on different mesh extraction approaches.** We evalute the commonly used mesh extractions from multiview images on two outdoor scenes form BlendedMVS, including traditional Multiview steoro reconstruction (Colmap [Schonberger and Frahm 2016; Schönberger et al. 2016]), NeRF-based reconstruction method (NeuS2 [Wang et al. 2023a]), and 3DGS-based reconstruction method (SuGaR [Guédon and Lepetit 2024]). For the different extracted meshes, we ensure the same number of vertex via the mesh simplications [Garland and Heckbert 1997]. From the results, it demonstrated that our method is very robust to different mesh extraction approaches.

mesh by NeuS2 with different resolutions (#Faces 100K, 50K, 20K, 10K). The meshes with different qualities are fed into our proposed representation to guide the Gaussian learning. Then, we deform the explicit mesh with the same control handles to drive the Gaussians and render them in desired views.

Figure 10 and Figure 11 demonstrate that our 3D Gaussian deformation pipeline is robust to different mesh extraction approaches and different mesh qualities and thus achieves similar performance on Gaussian deformation from novel views.

As a commonly used 3D geometry representation, the mesh is favored in many tasks of rendering and CAD design, especially when combined with texture maps as the appearance descriptor. However, in most cases, this representation is hard to represent detailed geometry and geometric textures, such as the fur and hair on the surface (*e.g.*, flower/grass-shaped geometry, stuffed toys) [Lengyel et al. 2001]. Given multi-view images as input, we use the NeRF-based reconstructed mesh and texture map as the baseline (denoted as 'Mesh + TextureMap') to compare with our proposed representation. The explicit geometry and texture map are respectively recovered by the NeRF-based method (NeuS2) and differential rendering (nvdiffrast [Laine et al. 2020]). In detail, the approach jointly optimizes geometry and texture map with its resolution set to 4096×4096. Then, we render it via PyTorch3D [Ravi et al. 2020]with the same camera pose for comparison. For our method, we only use the same geometry mesh with 'Mesh + TextureMap' as the explicit proxy to optimize the Gaussian kernels. Here, we show three examples, two synthetic ones and one in-the-wild example in Figure 12. It is obvious from the results that our method can successfully represent high-quality details (*e.g.*, leaves of flowers and hair from stuffed toys) when using the same explicit geometry compared to the 'Mesh + TextureMap' representation. The texture map in the 'Mesh + TextureMap' method is blurry due to the instability of the geometric optimization process. It fails to fully eliminate the influence of misregistration, resulting in blurriness.

Face Split Operation (F.S.). To facilitate the 3DGS deformation, we introduce the explicit mesh as a constraint to bind the Gaussians onto the surface as much as possible and design the face split strategy to divide the Gaussian. To validate the effectiveness of the Face Split operation, we remove the operation and instead employ a straightforward division strategy: In order to ensure that the final number of Gaussians is as consistent as possible, we copy one Gaussian into four copies that are connected to the current face and then let the network optimize the attributes of these Gaussians. For the ablated version, the Gaussian distribution cannot be regularized evenly due to lack of the Face Split operation (the Face

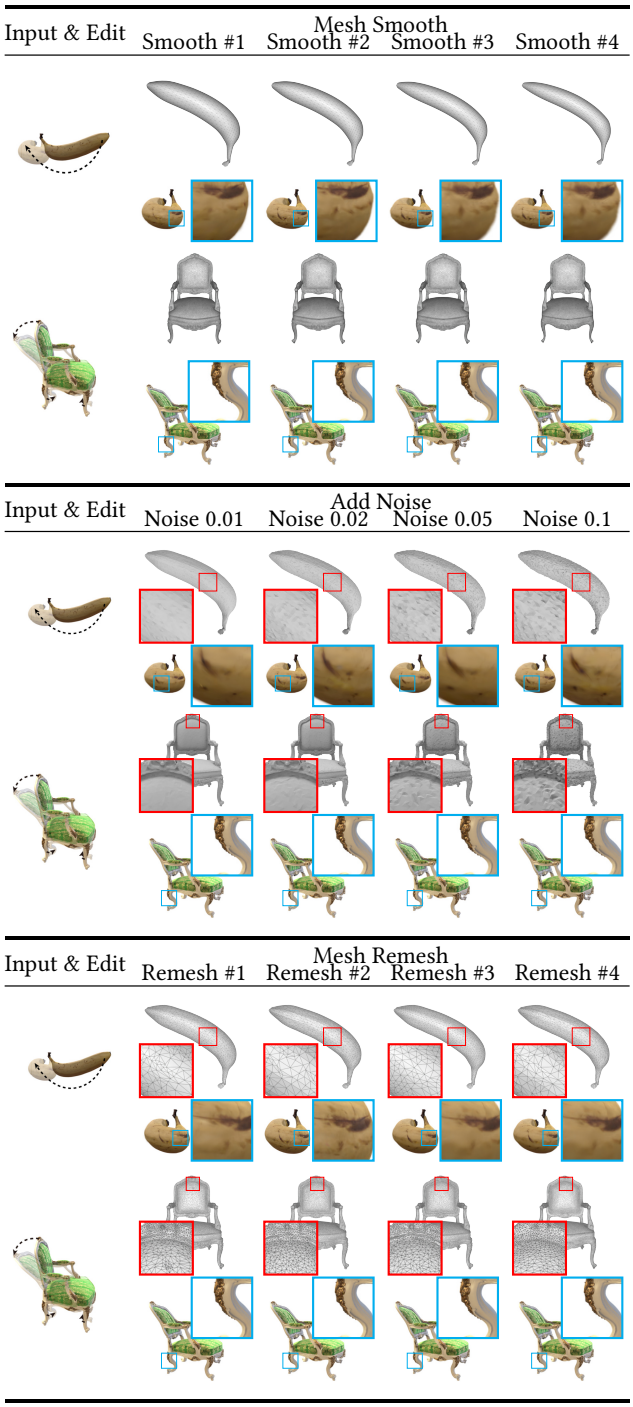

Fig. 10. **Robustness validation on different mesh qualities.** We perform the evaluations on two synthetic examples, Chair and Banana. Three modifications (*i.e.*, add noise, remeshing, and smoothing) are employed to alter the quality of the mesh obtained from NeuS2 at four different scales. These meshes are utilized to train the 3D GS and deformation, then render the deformation results visually. Based on the results, our method demonstrates a high level of resilience to various mesh modifications.

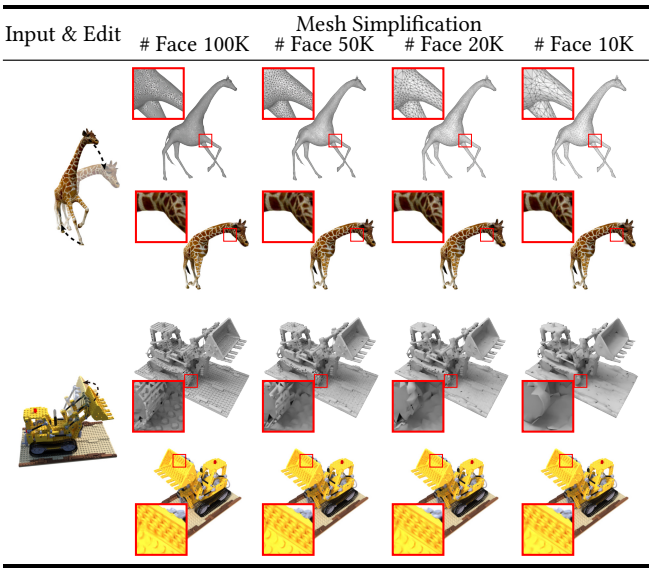

Fig. 11. **Robustness validation on meshes with different face numbers.** For robustness on different face numbers, we evaluate the impact of varying mesh resolutions on the deformation results for novel views. Hence, we perform the evaluations on two synthetic examples, Lego and Giraffe. The extracted mesh by NeuS2 are simplified by [Garland and Heckbert 1997] with different face numbers, ranging from 100K to 10K. The results indicate that our methods is not sensitive to the mesh resolution when generating novel views of deformed 3D Gaussians.

Split operation can regularize the Gaussian positions evenly near the surface), and only overfits the input images with the known views. Although the final rendering results (before edit) are visually good, the Gaussian distribution (w.o. Face Split) is not well suitable for deformation, especially for large-scale deformation. We perform the quantitative and qualitative evaluations. Table 2 reports the scores on the novel view synthesis under three commonly used metrics (*i.e.*, PSNR, SSIM, LPIPS). Figure 13 presents the qualitative results of the 3DGS deformation. The results of the quantitative and qualitative evaluations together demonstrate that the Face Split operation can improve the rendering results from novel views and prevent blurry rendering artifacts after deformation for high-frequency textures. (*e.g.*, the 2nd and 3rd columns in Figure 13).

Normal Guidance (N.G.). To accurately capture high-frequency details (such as the hair on the surface and stuffed toys), we propose normal guidance, ensuring that the Gaussian function is positioned close to the surface rather than just on the surface. To validate the effectiveness of normal guidance, we perform the evaluations on novel view synthesis task. Table 2 and Figure 14 together illustrate that normal guidance can successfully improve the rendering quality of novel views both quantitatively and qualitatively (*e.g.*, the highlighted detailed structures of the Horse and Hat examples). Note that the normal guidance improves the rendering quality significantly; if removing it, it also affects the deformation results, so we only validate it on the novel view synthesis task.

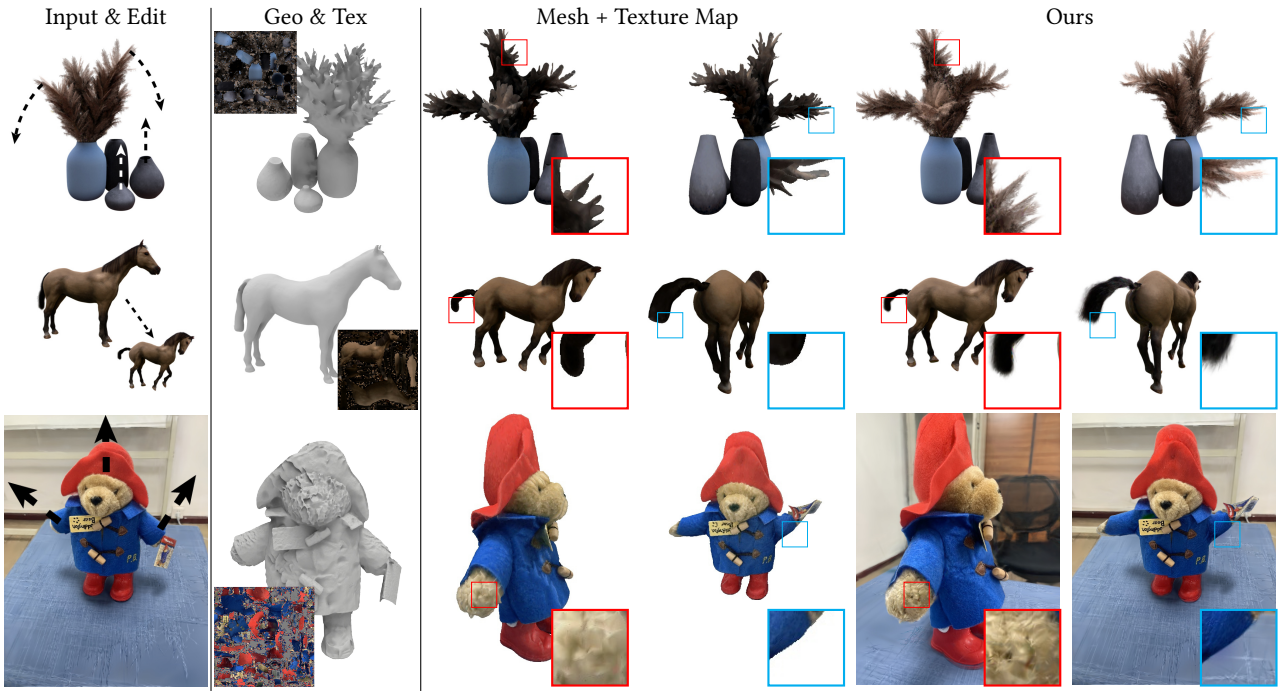

Fig. 12. **Comparison with the Texture + Mesh representation.** Our mesh-base GS representation utilized the mesh representation to boost the performance of vanilla 3DGS, but mesh representation still plays significant role in most pipeline of graphics. To validate the effectiveness of our proposed representation, we perform the comparison with classical Mesh+TextureMap representation on three examples (Teddy, Horse, Flowers) with detailed hair or grass-shaped geometry, since it is very expensive modeling for classical mesh representation. For the texture map and geometry, we use the NeuS2 [Wang et al. 2023a] to reconstruct the geometry and nvdiffrast [Laine et al. 2020] to recover the texture. Then we render it via Pytorch3D [Ravi et al. 2020] with the same camera. From the visual results, it is obviously clear that our representation achieve the better results on detailed appearance, even with the large-scale deformation.

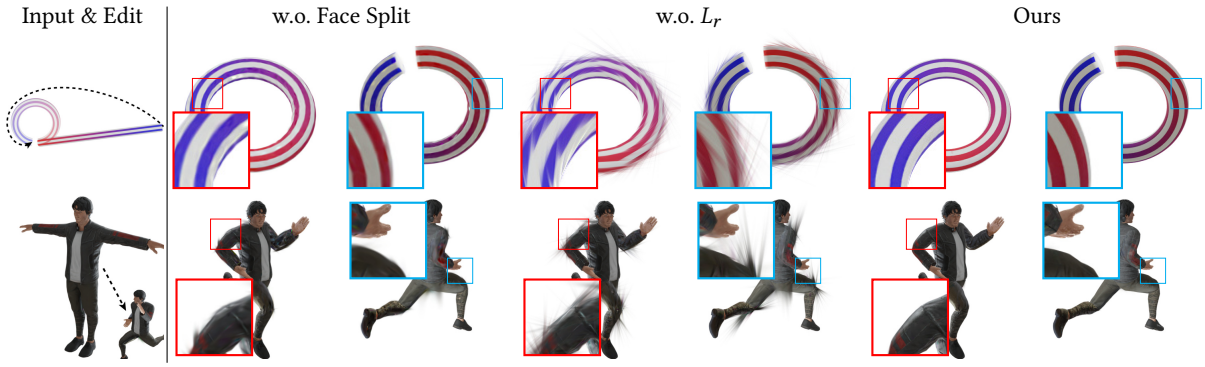

Fig. 13. **Ablations on Face Split operation and regularization L_r.** We perform a qualitative comparison on our two ablated versions: w/o Face Split and w/o L_r. The two cases illustrate that our full method can achieve the best results under large-scale deformation. It becomes more evident that there are a greater number of Gaussians with irrational shapes when L_r is not present. Disabling Face Split may result in blurry artifacts.

Regularization L_r. During Gaussian distribution training, many long-narrow shapes are used to represent high-frequency appearances but are poorly suited for large-scale deformations. To address this, we introduce the regularization term L_r, which enforces Gaussian sizes to be smaller than the attached faces, improving the results of large-scale deformation. In Figure 13, it is obvious that the ablated version (w/o L_r) leads to more artifacts when large-scale deformations appear. Since we apply the additional constraints to the vanilla 3DGS to make the Gaussians more suitable for deformation, this could affect the performance of vanilla 3D-GS for the novel view rendering. So, an appropriate balancing weight should be set to effectively enhance the deformation quality for large-scale deformations while ensuring the novel view synthesis quality. Users can further manually select the different weights for novel view synthesis and 3DGS deformation as needed.

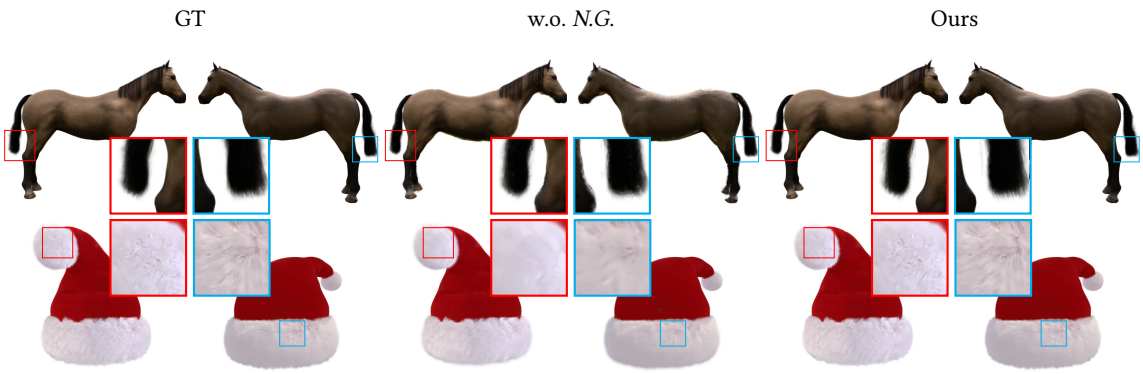

Fig. 14. **Ablation of Normal Guidance on novel view synthesis.** It is clear that Normal Guidance can enhance the high-frequency details and complex structures significantly from novel view renderings. The differences are highlighted with different colored boxes for different views (*e.g.*, the highlighted detailed structures of the Horse and Hat examples).

5 CONCLUSIONS & FUTURE WORK

Conclusions. In this paper, we have proposed a novel large-scale deformation method for 3D Gaussian Splatting, based on a mesh-based representation. The proposed 3DGS deformation method enables the manipulation of the 3DGS in an intuitive, interactive manner. To well facilitate the 3DGS deformation, we incorporate an explicit mesh that can be easily extracted by existing methods and is bound with Gaussian ellipsoids, thus enablingeffective large-scale deformation of 3D GS. In addition, we employ a Gaussian division model that operates on the explicit mesh through face split and normal guidance, which can improve the visual quality and prevent artifacts that might occur during large-scale deformations. Based on the mesh-based GS representation, we introduce a large-scale mesh deformation method to enable deformable Gaussian Splatting by altering the parameters of the 3D Gaussians according to the user's intuitive manipulations.

Limitations & Future Work. Despite our approach successfully achieving real-time large-scale deformation with the Gaussian Splatting representation, it still faces the following obstacles: 1) The visual appearance and shadow are still baked on the Gaussians and cannot be further edited. Hence, it is worth developing new methods that can not only deform the geometry of Gaussians, but also support editing of the appearance of Gaussians for future work. 2) This method relies on the extracted mesh as the proxy, it would fail if the mesh could not be extracted, for example, for complex transparent objects. The mesh extractor is an off-the-shell module in our pipeline and is not integrated into our mesh-based GS representation. Since mesh extraction is a very fundamental and challenging task for 3DGS, there are some research works [Hou et al. 2023; Liu et al. 2023a, 2024a; Long et al. 2023; Meng et al. 2023; Son et al. 2024] to explore better geometry reconstruction from 3DGS, especially for thin structures, non-watertight shapes, or large scenes. Further, the joint optimization of high-quality geometry and our mesh-based GS is more challenging. We are interested in exploring geometry extraction and editable properties together, by designing an end-to-end framework to optimize the mesh and Gaussian distribution jointly as part of future work. Moreover, inspired by the mesh pooling and unpooling operations in SubdivNet [Hu et al. 2022], how to optimize the Gaussian deletion operation is also a future research area. Face merging or mesh simplification [Garland and Heckbert 1997] are potential approach to substitute the inherited deletion operation in vanilla 3DGS.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (No. 62322210 and No. 62302484), the Innovation Funding of ICT, CAS (No. E461020), the Beijing Municipal Natural Science Foundation for Distinguished Young Scholars (No. JQ21013), the Beijing Municipal Science and Technology Commission (No. Z231100005923031), and the Engineering and Physical Sciences Research Council (grant number EP/Y028805/1).

REFERENCES

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. 2018. Learning Representations and Generative Models for 3D Point Clouds. In *International conference on machine learning*. PMLR, 40–49.

Pierre Alliez, Eric Colin De Verdire, Olivier Devillers, and Martin Isenburg. 2003. Isotropic Surface Remeshing. In *2003 Shape Modeling International.* IEEE, 49–58.

Chong Bao, Yinda Zhang, and Bangbang et al. Yang. 2023. SINE: Semantic-driven Image-based NeRF Editing with Prior-guided Editing Field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 20919–20929.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. 2022. Mip-NeRF 360: Unbounded Anti-aliased Neural Radiance Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 5470–5479.

Javier Bonet and Richard D Wood. 1997. *Nonlinear Continuum Mechanics for Finite Element Analysis.* Cambridge university press.

Guikun Chen and Wenguan Wang. 2024. A Survey on 3D Gaussian Splatting. *arXiv preprint arXiv:2401.03890* (2024).

Zilong Chen, Feng Wang, Yikai Wang, and Huaping Liu. 2024. Text-to-3D using Gaussian Splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 21401–21412.

Zhiqin Chen and Hao Zhang. 2019. Learning Implicit Fields for Generative Shape Modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 5939–5948.

Ho Kei Cheng, Seoung Wug Oh, Brian Price, Joon-Young Lee, and Alexander Schwing. 2024. Putting the Object Back Into Video Object Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 3151–3161.

Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. 2020a. Implicit Functions in Feature Space for 3D Shape Reconstruction and Completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 6970–6981.

Julian Chibane, Gerard Pons-Moll, et al. 2020b. Neural unsigned distance fields for implicit function learning. *Advances in Neural Information Processing Systems* 33 (2020), 21638–21652.

Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 2016. 3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*. Springer, 628–644.

Bardienus P Duisterhof, Zhao Mandi, Yunchao Yao, Jia-Wei Liu, Mike Zheng Shou, Shuran Song, and Jeffrey Ichnowski. 2023. MD-Splatting: Learning Metric Deformation from 4D Gaussians in Highly Deformable Scenes. *arXiv preprint arXiv:2312.00583* (2023).

Haoqiang Fan, Hao Su, and Leonidas J Guibas. 2017. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*. 605–613.

Ben Fei, Jingyi Xu, Rui Zhang, Qingyuan Zhou, Weidong Yang, and Ying He. 2024. 3D Gaussian as a New Vision Era: A Survey. *arXiv preprint arXiv:2402.07181* (2024).

Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. 2022. Plenoxels: Radiance Fields without Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5501–5510.

Lin Gao, Yu-Kun Lai, Jie Yang, Ling-Xiao Zhang, Shihong Xia, and Leif Kobbelt. 2019a. Sparse Data Driven Mesh Deformation. *IEEE transactions on visualization and computer graphics* 27, 3 (2019), 2085–2100.

Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao(Richard) Zhang. 2019b. SDM-NET: Deep Generative Network for Structured Deformable Mesh. *ACM Transactions on Graphics* 38, 6 (2019), 243:1–243:15.

William Gao, Noam Aigerman, Thibault Groueix, Vova Kim, and Rana Hanocka. 2023. TextDeformer: Geometry Manipulation using Text Guidance. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–11.

Michael Garland and Paul S Heckbert. 1997. Surface Simplification using Quadric Error Metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 209–216.

Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 2018. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 216–224.

Antoine Guédon and Vincent Lepetit. 2024. SuGaR: Surface-aligned Gaussian Splatting for Efficient 3D Mesh Reconstruction and High-quality Mesh Rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5354–5363.

Benoit Guillard, Federico Stella, and Pascal Fua. 2022. MeshUDF: Fast and Differentiable Meshing of Unsigned Distance Field Networks. In *European Conference on Computer Vision*. Springer, 576–592.

Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. 2019. MeshCNN: A Network with An Edge. *ACM Transactions on Graphics* 38, 4 (2019), 1–12.

Ayaan Haque, Matthew Tancik, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. 2023. Instruct-NeRF2NeRF: Editing 3D Scenes with Instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 19683–19693.

Fei Hou, Xuhui Chen, Wencheng Wang, Hong Qin, and Ying He. 2023. Robust Zero Level-Set Extraction from Unsigned Distance Fields Based on Double Covering. *ACM Transactions on Graphics* 42, 6 (2023), 1–15.

Liangxiao Hu, Hongwen Zhang, Yuxiang Zhang, Boyao Zhou, Boning Liu, Shengping Zhang, and Liqiang Nie. 2024. GaussianAvatar: Towards Realistic Human Avatar Modeling from a Single Video via Animatable 3D Gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 634–644.

Shi-Min Hu, Zheng-Ning Liu, Meng-Hao Guo, Jun-Xiong Cai, Jiahui Huang, Tai-Jiang Mu, and Ralph R Martin. 2022. Subdivision-based Mesh Convolution Networks. *ACM Transactions on Graphics* 41, 3 (2022), 1–16.

Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. 2024. SC-GS: Sparse-controlled Gaussian Splatting for Editable Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4220–4230.

Clément Jambon, Bernhard Kerbl, Georgios Kopanas, Stavros Diolatzis, Thomas Leimkühler, and George Drettakis. 2023. NeRFshop: Interactive Editing of Neural Radiance Fields. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 6, 1, Article 1 (may 2023), 21 pages. https://doi.org/10.1145/3585499

Dadong Jiang, Zhihui Ke, Xiaobo Zhou, and Xidong Shi. 2023a. 4D-Editor: Interactive Object-level Editing in Dynamic Neural Radiance Fields via Semantic Distillation. *arXiv e-prints* (2023), arXiv–2310.

Ruixiang Jiang, Can Wang, Jingbo Zhang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. 2023b. AvatarCraft: Transforming Text into Neural Human Avatars with Parameterized Shape and Pose Control. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 14371–14382.

Ying Jiang, Chang Yu, Tianyi Xie, Xuan Li, Yutao Feng, Huamin Wang, Minchen Li, Henry Lau, Feng Gao, Yin Yang, and Chenfanfu Jiang. 2024. VR-GS: A Physical Dynamics-Aware Interactive Gaussian Splatting System in Virtual Reality. In *ACM SIGGRAPH Conference Papers*. Article 78, 1-12 pages. https://doi.org/10.1145/3641519.3657448

James T Kajiya and Brian P Von Herzen. 1984. Ray Tracing Volume Densities. *ACM SIGGRAPH computer graphics* 18, 3 (1984), 165–174.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-time Radiance Field Rendering. *ACM Transactions on Graphics* 42, 4 (2023), 1–14.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. 2017. Tanks and Temples: Benchmarking Large-scale Scene Reconstruction. *ACM Transactions on Graphics* 36, 4 (2017), 1–13.

Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. 2022. Decomposing NeRF for Editing via Feature Field Distillation. *Advances in Neural Information Processing Systems* 35 (2022), 23311–23330.

Muhammed Kocabas, Jen-Hao Rick Chang, James Gabriel, Oncel Tuzel, and Anurag Ranjan. 2024. HUGS: Human Gaussian Splats. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 505–515.

Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. 2020. Modular Primitives for High-performance Differentiable Rendering. *ACM Transactions on Graphics* 39, 6 (2020), 1–14.

Jerome Edward Lengyel, Emil Praun, Adam Finkelstein, and Hugues Hoppe. 2001. Real-time Fur Over Arbitrary Surfaces. In *Proceedings of the 2001 Symposium on Interactive 3D Graphics*. 227–232.

Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. 2023. Neuralangelo: High-fidelity Neural Surface Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8456–8465.

Zhe Li, Zerong Zheng, Lizhen Wang, and Yebin Liu. 2024. Animatable Gaussians: Learning Pose-dependent Gaussian Maps for High-fidelity Human Avatar Modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 19711–19722.

Yaron Lipman, Olga Sorkine-Hornung, Marc Alexa, Daniel Cohen-Or, David Levin, Christian Rössl, and Hans-Peter Seidel. 2005. Laplacian Framework for Interactive Mesh Editing. *Int. J. Shape Model.* 11 (2005), 43–62.

Hao-Kang Liu, I Shen, Bing-Yu Chen, et al. 2024b. NeRF-In: Free-form NeRF Inpainting with RGB-D Priors. *Computer Graphics and Applications* 44, 2 (2024), 100–109.

Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. 2023b. Zero-1-to-3: Zero-shot one Image to 3D Object. In *Proceedings of the IEEE/CVF international conference on computer vision*. 9298–9309.

Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. 2021. Editing Conditional Radiance Fields. In *Proceedings of the IEEE/CVF international conference on computer vision*. 5773–5783.

Yu-Tao Liu, Li Wang, Jie Yang, Weikai Chen, Xiaoxu Meng, Bo Yang, and Lin Gao. 2023a. NeUDF: Leaning Neural Unsigned Distance Fields with Volume Rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 237–247.

Zhen Liu, Yao Feng, Yuliang Xiu, Weiyang Liu, Liam Paull, Michael J. Black, and Bernhard Schölkopf. 2024a. Ghost-on-the-Shell: An Expressive Representation of General 3D Shapes. In *International Conference on Learning Representations*. 1:14.

Xiaoxiao Long, Cheng Lin, Lingjie Liu, Yuan Liu, Peng Wang, Christian Theobalt, Taku Komura, and Wenping Wang. 2023. NeuralUDF: Learning Unsigned Distance Fields for Multi-view Reconstruction of Surfaces with Arbitrary Topologies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 20834–20843.

Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. 2024. Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. In *International Conference on 3D Vision*. 1–11.

Daniel Maturana and Sebastian Scherer. 2015. VoxNet: A 3D Convolutional Neural Network for Real-time Object Recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 922–928.

Xiaoxu Meng, Weikai Chen, and Bo Yang. 2023. NeAT: Learning Neural Implicit Surfaces with Arbitrary Topologies from Multi-view Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 248–258.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. 2019. Occupancy Networks: Learning 3D Reconstruction in Function Space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4460–4470.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Transactions on Graphics* 41, 4 (2022), 1–15.

Francesco Palandra, Andrea Sanchietti, Daniele Baieri, and Emanuele Rodolà. 2024. GSEdit: Efficient Text-Guided Editing of 3D Objects via Gaussian Splatting. *arXiv preprint arXiv:2403.05154* (2024).

Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. 2023. Drag Your GAN: Interactive Point-based Manipulation on

the Generative Image Manifold. In *ACM SIGGRAPH Conference Proceedings*. 1–11.

Sida Peng, Yuanqing Zhang, Yinghao Xu, and et al. 2021. Neural Body: Implicit Neural Representations with Structured Latent Codes for Novel View Synthesis of Dynamic Humans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9054–9063.

Cédric Pinson. 2011. Sketchfab - The Best 3D Viewer on the Web. https://sketchfab.com/.

Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. 2023. DreamFusion: Text-to-3D using 2D Diffusion. In *The Eleventh International Conference on Learning Representations*. 1–14.

Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. 2021. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10318–10327.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017a. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*. 652–660.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in neural information processing systems*. 5099–5108.

Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. 2020. Accelerating 3D Deep Learning with PyTorch3D. *arXiv:2007.08501* (2020).

Johannes L Schonberger and Jan-Michael Frahm. 2016. Structure-from-motion Revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4104–4113.

Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. 2016. Pixelwise View Selection for Unstructured Multi-view Stereo. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*. Springer, 501–518.

Sanghyun Son, Matheus Gadelha, Yang Zhou, Zexiang Xu, Ming C Lin, and Yi Zhou. 2024. DMesh: A Differentiable Representation for General Meshes. *arXiv preprint arXiv:2404.13445* (2024).

Olga Sorkine. 2005. Laplacian Mesh Processing. *Eurographics (State of the Art Reports)* 4, 4 (2005), 1.

Olga Sorkine and Marc Alexa. 2007. As-rigid-as-possible Surface Modeling. In *Symposium on Geometry processing*, Vol. 4. Citeseer, 109–116.

Zhaoqi Su, Tao Yu, Yangang Wang, and Yebin Liu. 2023. DeepCloth: Neural Garment Representation for Shape and Style Editing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 2 (2023), 1581–1593. https://doi.org/10.1109/TPAMI.2022.3168569

Robert W Sumner, Matthias Zwicker, Craig Gotsman, and Jovan Popović. 2005. Mesh-based Inverse Kinematics. *ACM transactions on graphics* 24, 3 (2005), 488–495.

Jia-Mu Sun, Tong Wu, and Lin Gao. 2024. Recent Advances in Implicit Representation-based 3D Shape Generation. *Visual Intelligence* 2, 1 (2024), 9.

Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. 2024. DreamGaussian: Generative Gaussian Splatting for Efficient 3D Content Creation. In *International Conference on Learning Representations*. 1–14.

Shimon Ullman. 1979. The Interpretation of Structure from Motion. *Proceedings of the Royal Society of London. Series B. Biological Sciences* 203, 1153 (1979), 405–426.

Joanna Waczyńska, Piotr Borycki, Sławomir Tadeja, Jacek Tabor, and Przemysław Spurek. 2024. GaMeS: Mesh-Based Adapting and Modification of Gaussian Splatting. *arXiv preprint arXiv:2402.01459* (2024).

Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. 2022. CLIP-NeRF: Text-and-image Driven Manipulation of Neural Radiance Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3835–3844.

Can Wang, Ruixiang Jiang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. 2024. NeRF-Art: Text-Driven Neural Radiance Fields Stylization. *IEEE Transactions on Visualization and Computer Graphics* 30, 8 (2024), 4983–4996. https://doi.org/10.1109/TVCG.2023.3283400

Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. 2018b. Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images. In *Proceedings of the European conference on computer vision (ECCV)*. 52–67.

Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. 2021. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. In *Advances in Neural Information Processing Systems*, Vol. 34. Curran Associates, Inc., 27171–27183.

Peng-Shuai Wang, Chun-Yu Sun, Yang Liu, and Xin Tong. 2018a. Adaptive O-CNN: A Patch-based Deep Representation of 3D Shapes. *ACM Transactions on Graphics* 37, 6 (2018), 1–11.

Xiangyu Wang, Jingsen Zhu, Qi Ye, Yuchi Huo, Yunlong Ran, Zhihua Zhong, and Jiming Chen. 2023b. Seal-3D: Interactive Pixel-level Editing for Neural Radiance Fields. In *Proceedings of the IEEE/CVF Conference on International Conference on Computer Vision*. 17683–17693.

Yiming Wang, Qin Han, Marc Habermann, Kostas Daniilidis, Christian Theobalt, and Lingjie Liu. 2023a. NeuS2: Fast Learning of Neural Implicit Surfaces for Multi-view Reconstruction. In *Proceedings of the IEEE/CVF Conference on International Conference on Computer Vision*. 3295–3306.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612.

Jing Wu, Jia-Wang Bian, Xinghui Li, Guangrun Wang, Ian Reid, Philip Torr, and Victor Adrian Prisacariu. 2024a. GaussCtrl: Multi-View Consistent Text-Driven 3D Gaussian Splatting Editing. In *Proceedings of the European Conference on Computer Vision*. 1–17.

Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. 2016. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. In *Advances in Neural Information Processing Systems*. 82–90.

Tong Wu, Yu-Jie Yuan, Ling-Xiao Zhang, Jie Yang, Yan-Pei Cao, Ling-Qi Yan, and Lin Gao. 2024b. Recent Advances in 3D Gaussian Splatting. *Computational Visual Media* (2024), 1–30.

Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. 2024. PhysGaussian: Physics-integrated 3D Gaussians for Generative Dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4389–4398.

Tianhan Xu and Tatsuya Harada. 2022. Deforming Radiance Fields with Cages. In *European Conference on Computer Vision*. Springer, 159–175.

Bangbang Yang, Chong Bao, Junyi Zeng, Hujun Bao, Yinda Zhang, Zhaopeng Cui, and Guofeng Zhang. 2022a. NeuMesh: Learning Disentangled Neural Mesh-based Implicit Field for Geometry and Texture Editing. In *European Conference on Computer Vision*. Springer, 597–614.

Jie Yang, Kaichun Mo, Yu-Kun Lai, Leonidas J Guibas, and Lin Gao. 2022b. DSG-Net: Learning Disentangled Structure and Geometry for 3D Shape Generation. *ACM Transactions on Graphics* 42, 1 (2022), 1–17.

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. 2024. Deformable 3D Gaussians for High-Fidelity Monocular Dynamic Scene Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 20331–20341.

Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. 2020. BlendedMVS: A Large-scale Dataset for Generalized Multi-view Stereo Networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 1790–1799.

Taoran Yi, Jiemin Fang, Junjie Wang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. 2024. GaussianDreamer: Fast Generation from Text to 3D Gaussians by Bridging 2D and 3D Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 6796–6807.

Wang Yifan, Noam Aigerman, Vladimir G. Kim, Siddhartha Chaudhuri, and Olga Sorkine-Hornung. 2020. Neural Cages for Detail-Preserving 3D Deformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 75–83.

Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. 2021. Plenoctrees for Real-time Rendering of Neural Radiance Fields. In *Proceedings of the IEEE/CVF Conference on International Conference on Computer Vision*. 5752–5761.

Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. 2004. Mesh Editing with Poisson-based Gradient Field Manipulation. In *ACM SIGGRAPH 2004 Papers*. 644–651.

Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. 2022. NeRF-Editing: Geometry Editing of Neural Radiance Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 18332–18343.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *Proceedings of the IEEE conference Conference on Computer Vision and Pattern Recognition*. 586–595.

Yuzhe Zhang, Jianmin Zheng, and Yiyu Cai. 2020. Proxy-driven Free-form Deformation by Topology-adjustable Control Lattice. *Computers & Graphics* 89 (2020), 167–177.

Shunyuan Zheng, Boyao Zhou, Ruizhi Shao, Boning Liu, Shengping Zhang, Liqiang Nie, and Yebin Liu. 2024. GPS-Gaussian: Generalizable Pixel-wise 3D Gaussian Splatting for Real-time Human Novel View Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 19680–19690.

Hao Zhu, Yongming Nie, Tao Yue, and Xun Cao. 2017. The Role of Prior in Image based 3D Modeling: A Survey. *Frontiers of Computer Science* 11 (2017), 175–191.

Jingyu Zhuang, Di Kang, Yan-Pei Cao, Guanbin Li, Liang Lin, and Ying Shan. 2024. TIP-Editor: An Accurate 3D Editor following Both Text-prompts and Image-prompts. *ACM Transactions on Graphics* 43, 4 (2024), 1–12.

Jingyu Zhuang, Chen Wang, Liang Lin, Lingjie Liu, and Guanbin Li. 2023. DreamEditor: Text-driven 3D Scene Editing with Neural Fields. In *SIGGRAPH Asia 2023 Conference Papers*. 1–10.

Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. 2001. Surface Splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. 371–378.