

## 1 Additional results

**Additional results.** We present additional qualitative results of our DFFSplat in Figure 1. DFFSplat leverages the 3D diffusion feature field to encode view-consistent semantic information from the source and utilizes this information during the editing process. As illustrated in Figure 1, our approach effectively preserves the original structure of the source 3DGS. However, using the 3D diffusion feature field alone leads to the loss of view-dependent information. To mitigate this issue, we employ a dual-encoder scheme that introduces an alternative pathway bypassing the 3D diffusion feature field, preserving view-dependent appearance details derived from the textual input prompt. This strategy allows our method to maintain appearance details specified by the editing instruction prompts, thereby enhancing editing fidelity. As shown in Figure 1, our method accurately reflects the provided editing instructions.

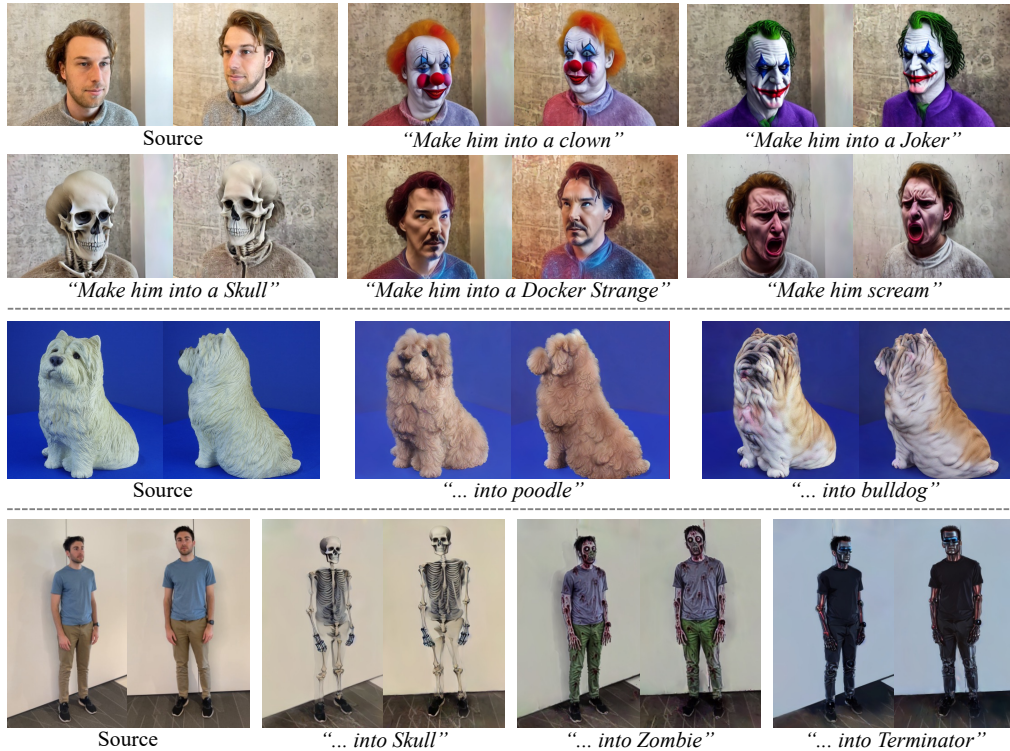


Figure 1: **Additional DFFSplat results.** We provide additional qualitative results obtained using various input prompts. These results demonstrate that our DFFSplat effectively preserves the original source structure while maintaining high fidelity to the input textual instructions.

**Effectiveness of our method.** To qualitatively evaluate the effectiveness of our DFFSplat, we conducted comparative experiments on editing results obtained with and without applying DFFSplat. As demonstrated in Figure 2, for the instruction *"Turn him into a Storm Trooper with mask"*, the edited images without DFFSplat exhibited a bias towards generating frontal-view faces. This causes unintended frontal faces to erroneously appear in the side-view of final edited 3DGS. In contrast, applying DFFSplat effectively preserved the original side-view facial structure. Similarly, given the instruction *"Make the bear look like a grizzly bear"*, the edited views without DFFSplat incorrectly introduced *"grizzly bear's face"* at the back-view, resulting in the so-called Janus artifacts. Conversely, using DFFSplat successfully eliminated these unintended back-view facial features, effectively preserving the original structure and significantly mitigating such artifacts.

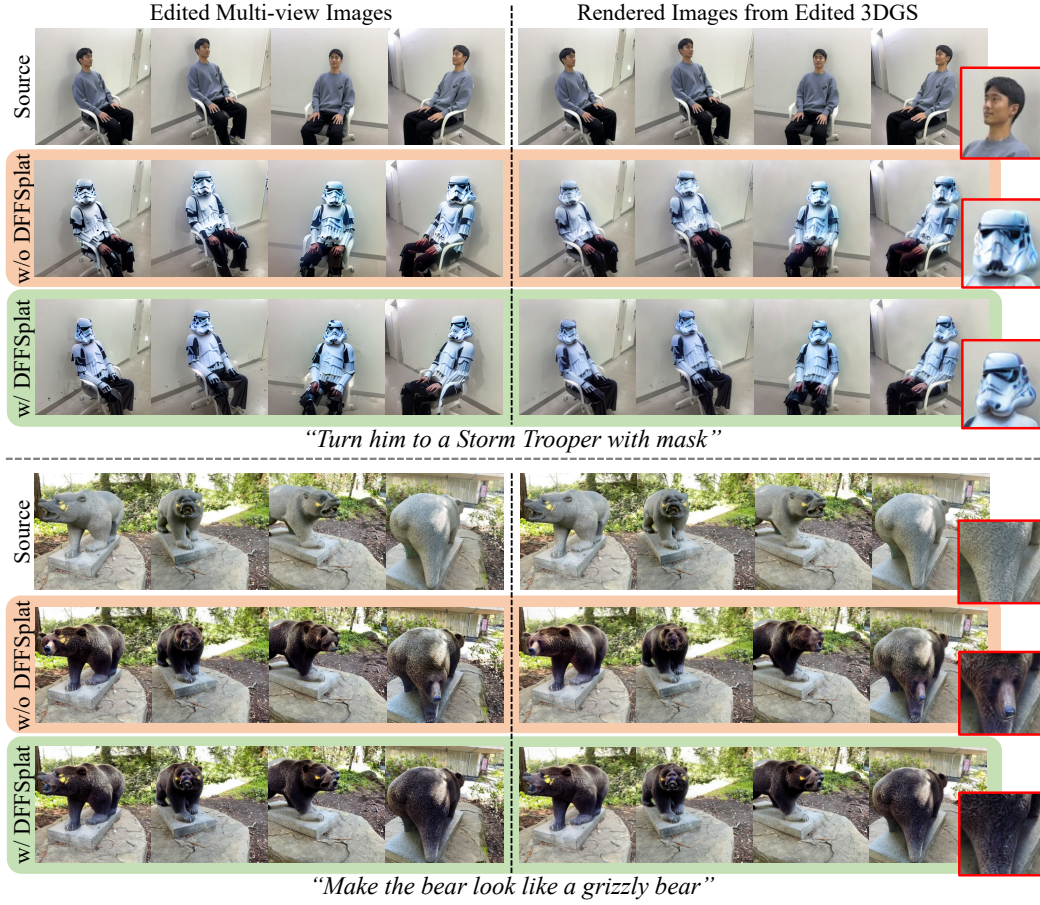


Figure 2: **Effectiveness of our method.** (Left) Editing results of multi-view source images, and (Right) rendered results of the edited 3DGS trained with these edited images. The orange regions indicate results obtained without using DFFSplat, while the green regions correspond to results obtained with DFFSplat. When applying DFFSplat, the edited images aligns with the original source structure, effectively resolving the Janus artifacts in edited 3DGS.

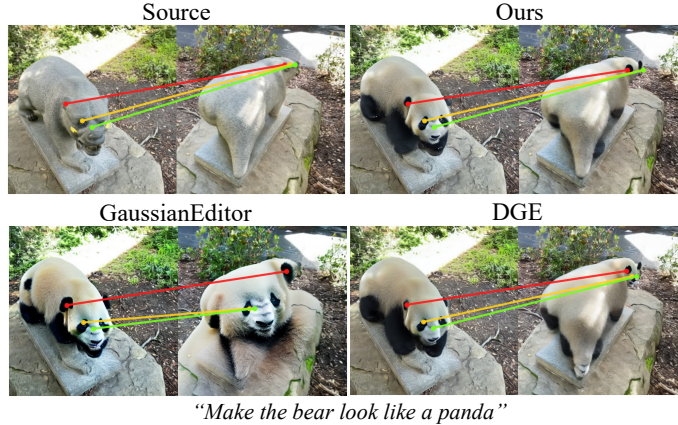


Figure 3: **Correspondence matching for the edited bear scene.** We compute correspondence maps between pairs of different views for both the source images and their corresponding edited results. Compared to other baseline methods, our approach demonstrates correspondence maps that are significantly closer to those of the source, confirming that our method more effectively maintains structure of source and per-view semantic consistency.

**Additional correspondence matching results.** To verify semantic consistency after editing, we compute correspondence maps between pairs of different views for both the source images and their corresponding edited results. As shown in Figure 3, GaussianEditor incorrectly generates a face in the back-view, causing erroneous face correspondences toward the back. Similarly, DGE incorrectly generates a face oriented toward the side-view, resulting in inaccurate face correspondences to the side. In contrast, our method accurately maintains the original structure, correctly matching the face to the frontal region. This confirms our method’s effectiveness in preserving the source structure and maintaining per-view semantic consistency.

## 2 Implementation details

**Configure.** We propose a novel diffusion feature field that captures semantic information for 3D Gaussian Splatting, enabling per-view semantically consistent editing from user-provided textual prompts. Our approach edits 3DGS models within approximately 4 minutes on average across our experimental scenes. Training the dual-encoder, which separately encodes view-independent semantic features and view-dependent information, takes about 30 seconds, while constructing the 3D-consistent diffusion feature field requires about 4 minutes. All experiments are conducted on a Nvidia Quadro RTX A6000 GPU with 48G memory.

**Text-based 3DGS editing.** We train our dual-encoder scheme, consisting of  $E_{str}$ ,  $E_{app}$ , and  $D$ , for 2500 iterations using the Adam optimizer with a learning rate of 0.002. To train the diffusion feature field of Gaussian Splatting, we optimize only the features associated with each Gaussian for 5000 iterations using the Adam optimizer with a learning rate of 0.01, while keeping the Gaussian positions and shapes fixed. Since our diffusion feature field exclusively encodes view-independent semantic information, we do not apply spherical harmonics (SH) calculations to these features.

During editing, we utilize the Instruct-Pix2Pix editor model, which is based on Stable Diffusion v1.5, to edit the rendered images. Following the experimental setups of previous methods (GaussianEditor, DGE), we use the DDIM scheduler with 20 sampling steps. Additionally, we adopt the noise sampling strategy from DGE, which samples noise from  $t \in [0.02, 0.98]$ . During image editing performed every 500 iterations, we apply additional noise corresponding to  $t = 0.3$  from the second iteration onward to achieve more stable editing results. Also, when optimizing the Gaussian Splatting model using multi-view edited images, we employ the Adam optimizer with the same hyperparameter settings as previous works (GaussianEditor, DGE), using a total of 1000 update iterations.

**Ablation study.** We conduct an ablation study on our key proposed components, specifically focusing on the dual-encoder scheme (involving  $E_{\text{app}}$  and structural loss  $\mathcal{L}_{\text{str}}$ ) and the time-invariance objective ( $\mathcal{L}_{\text{t-sim}}$ ). For the scenario without the dual-encoder scheme, we employ only a single encoder trained solely with reconstruction loss ( $\mathcal{L}_{\text{recon}}$ ), and this single encoder’s features are utilized to construct the diffusion feature field. For the scenario without the time-invariance objective, we retain the dual-encoder scheme but exclude the time-invariance term  $\mathcal{L}_{\text{t-sim}}$ . Additionally, to verify the effectiveness of the 3D diffusion feature field, we directly inject 2D features, obtained after encoding and decoding with our dual-encoder scheme, into the editing pipeline, without rendering process from 3D diffusion feature field. The results of this ablation study are presented in Table 2 and the left side of Figure 4 in the main paper.

Furthermore, we quantitatively measure the distortion introduced into the reconstructed diffusion features by feature injection. Specifically, we plot the feature distortion differences across diffusion timesteps in two conditions: (1) dual-encoder scheme with and without the 3D feature field, and (2) single encoder scheme (trained only with reconstruction loss) with and without the 3D feature field. The results of this analysis are presented on the right side of Figure 4 in the main paper. These analyses quantitatively confirm that employing the 3D feature field can lead to loss of view-dependent information, and that our proposed dual-encoder scheme effectively mitigates this issue.

**Evaluation via correspondence matching.** To verify that the edited results effectively preserve the source structure and mitigate editing artifacts such as the Janus problem, we compute differences of correspondence matching maps between rendered source and edited images. Specifically, correspondence matching maps are obtained by measuring similarities between semantic features extracted from pairs of view images. We extract these semantic features from the 11th layer of DINOv2-base, which captures rich semantic information, and match each pixel to its most similar counterpart. We uniformly sample  $N$  views from the training set at regular camera intervals, and calculate the difference of correspondence maps between source and edited images as follows:

$$\text{Diff}_{\text{corr}} = \frac{1}{N(N-1)} \sum_{i=1}^N \sum_{\substack{j=1 \\ j \neq i}}^N \|M_{v_i} \odot (C_{i,j}^{\text{src}} - C_{i,j}^{\text{edit}})\|_2, \quad (1)$$

where the correspondence maps  $C_{i,j} \in \mathbb{R}^{h \times w \times 2}$  between view  $v_i$  and another view  $v_j$  ( $j \neq i$ ) are defined as:

$$C_{i,j}(p) = \arg \min_q \|F_D(\mathbf{I}_{v_i})(p) - F_D(\mathbf{I}_{v_j})(q)\|_2. \quad (2)$$

Here,  $F_D(\mathbf{I}_{v_i}) \in \mathbb{R}^{h \times w \times d}$  represents the feature map extracted from image  $\mathbf{I}_{v_i}$  (from viewpoint  $v_i$ ) by the DINOv2-large model, and  $p, q$  represent pixel indices. The mask  $M_{v_i} \in \mathbb{R}^{h \times w}$  is the foreground mask indicating the object region for view  $v_i$ , and the symbol  $\odot$  denotes element-wise multiplication. A smaller value of  $\text{Diff}_{\text{corr}}$  indicates better preservation of the source geometry and improved multi-view consistency.

**Evaluation prompts.** In addition to the 10 scene-prompt pairs from DGE, we include 3 additional scene-prompt pairs. Detailed descriptions of the instruction prompts used for editing, as well as the source and target prompts employed for measuring CLIP and CLIP directional similarity, are reported in Table 1.

### 3 Social impact

Our proposed text-based 3DGS editing framework significantly simplifies content creation and editing, lowering the technical barrier for generating high-quality 3D assets. However, this ease of content manipulation also raises concerns regarding the potential generation and dissemination of misleading or harmful content. Moreover, biases or harmful outputs inherent in the underlying Stable Diffusion model could exacerbate these risks. Therefore, we urge users to adhere strictly to responsible usage guidelines and all relevant laws and regulations when employing our methodology.

Table 1: **Detailed evaluation dataset.** The source and target prompts are used for computing CLIP and CLIP directional scores. The edit instructions are textual inputs of InstructPix2Pix.

Scene	Source Prompt	Target Prompt	Edit Instruction
Face	“A man with curly hair in a grey jacket”	“A man with curly hair in a grey jacket with a Venetian mask”	“Give him a Venetian mask”
Face	“A man with curly hair in a grey jacket”	“A man with curly hair in a checkered cloth”	“Give him a checkered jacket”
Face	“A man with curly hair in a grey jacket”	“A spider man with a mask and curly hair”	“Turn him into spider man with a mask”
Bear	“A stone bear in a garden”	“A robotic bear in the garden”	“Make the bear look like a robot”
Bear	“A stone bear in a garden”	“A panda in the garden”	“Make the bear look like a panda”
Bear	“A stone bear in a garden”	“A bear with rainbow color”	“Make the color of the bear look like rainbow”
Person	“A man standing next to a wall wearing a blue T-shirt and brown pants”	“A man looks like a mosaic sculpture standing next to a wall”	“Make the man look like a mosaic sculpture”
Person	“A man standing next to a wall wearing a blue T-shirt and brown pants”	“A man wearing a shirt with a pineapple pattern”	“Make the person wear a shirt with a pineapple pattern”
Person	“A man standing next to a wall wearing a blue T-shirt and brown pants”	“An Iron Man stands to a wall”	“Turn him into Iron Man”
Person	“A man standing next to a wall wearing a blue T-shirt and brown pants”	“A robot stands to a wall”	“Turn the man into a robot”
Dog	“A photo of a dog”	“A photo of a corgi”	“Make the dog look like a corgi”
Yuseung	“A photo of a man”	“A photo of a clown”	“Turn him to a clown”
Yuseung	“A photo of a man”	“A photo of a Batman”	“Turn him to a Batman”