Diffusion Feature Field for Text-based 3D Editing with Gaussian Splatting

Eunseo Koh **Sangeek Hyun** **MinKyu Lee**
Jiwoo Chung **Kangmin Seo** **Jae-Pil Heo**[†]

Sungkyunkwan University
{colorrain, hsi1032, bluelati98, wldn0202, skmskku, jaepilheo}@skku.edu

Abstract

Recent advances in text-based image editing have motivated the extension of these techniques into the 3D domain. However, existing methods typically apply 2D diffusion models independently to multiple viewpoints, resulting in significant artifacts, most notably the Janus problem, due to inconsistencies across edited views. To address this, we propose a novel approach termed DFFSplat, which integrates a 3D-consistent diffusion feature field into the editing pipeline. By rendering and injecting these 3D-consistent structural features into intermediate layers of a 2D diffusion model, our method effectively enforces geometric alignment and semantic coherence across views. However, averaging 3D features during the feature field learning process can lead to the loss of fine texture details. To overcome this, we introduce a dual-encoder architecture to disentangle view-independent structural information from view-dependent appearance details. By encoding only the disentangled structure into the 3D field and injecting it during 2D editing phase, our method produces semantically and multi-view coherent edited images while maintaining high quality editing. Additionally, we employ a time-invariance objective to ensure consistency across diffusion timesteps, enhancing the stability of learned representations. Experimental results demonstrate that our method achieves state-of-the-art performance in terms of CLIP similarity, and better preserves structural and semantic consistency compared to existing approaches.

1 Introduction

Recent advances in text-based image generation models [28, 29, 27, 22] have enabled their widespread application, particularly in text-based image editing tasks [35, 10, 1, 14]. Driven by increasing demand for intuitive and efficient editing of 3D content, recent efforts have focused on extending these text-based editing techniques from the 2D domain into 3D representations. Consequently, numerous text-based 3D editing [9, 13, 30, 43, 4, 37, 3] approaches have been proposed, significantly broadening their applicability across diverse fields, including film production, game development.

The primary objective of text-based 3D editing is to modify a given source 3D representation according to user-provided textual instructions. To edit a source 3D representation based on the given textual instruction, existing approaches typically follow a two-step process. First, 2D editing methods are applied to multi-view images rendered from the source 3D representation. Subsequently, the 3D representation is optimized to align with these edited 2D reference views.

[†]Corresponding author

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

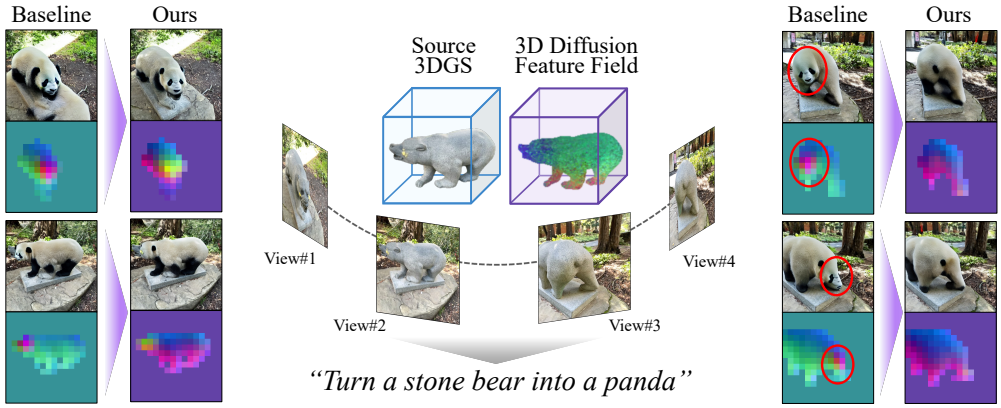


Figure 1: Diffusion-based 2D editing exhibits multi-view artifacts such as Janus problem, visible in the model's intermediate features, that lead to inconsistent edits across viewpoints (referred to as *Baseline*). For example, independent editing of 2D multi-view images causes the panda's face incorrectly appear on the bear's back. By enforcing feature-level multi-view and semantic consistency through our 3D diffusion feature field, these errors are resolved, yielding coherent edits across all views (referred to as *Ours*).

In 2D editing process, achieving *multi-view consistency* across edited images is essential. Multi-view consistency ensures that the edited scenes remain visually coherent across different viewpoints; inconsistent multi-view images inevitably introduce artifacts in the reconstructed 3D scene. However, existing text-based 3D editing methods utilize 2D diffusion models based editing approaches [1] that independently process each view, making it challenging to maintain multi-view consistency.

Here, we focus on the notable artifacts in 3D editing and generation known as the Janus problem [12, 24, 36]. Due to the inherent frontal-view bias in 2D diffusion models, frontal-view semantics attributes are often incorrectly synthesized onto multiple *non*-frontal viewpoints. This issue leads the semantic information to be misaligned in the multi-view images of source 3D during the 2D editing process. Interestingly, we observe that multi-view artifacts, like a panda's face on its back, already emerge in the diffusion model's intermediate features, highlighting that enforcing feature-level consistency is crucial to prevent such errors in the final 3D edits, as shown in Figure 1.

To this end, we propose DFFSplat, which augments the 2D editing phase with a multi-view consistent diffusion feature. Specifically, we aim to build a unified 3D feature field of the source scene that is shared across multiple views, and inject the rendered feature during the 2D diffusion editing [1]. Our key insight is that guiding the diffusion process with the rendered features already coherent in 3D encourages both geometric alignment across views and semantic fidelity to the source content.

However, as 3D features are fused across multiple viewpoints, they converge to an averaged representation and may lose per-view texture details. Accordingly, we observe that directly injecting these averaged features results in blurred edits. From this design, we hypothesize that the information lost due to its view-dependent nature during 3D embedding primarily related to appearance, whereas the preserved view-independent information retained in the feature field relates to the semantic structure.

To address the issue of blurry artifacts, we propose isolating only the view-independent semantic information into the Gaussian Splatting [15] representation, while preserving the remaining view-dependent features directly from the editing process guided by the textual instruction. Given the strong correlation between structural layouts and semantic maps, we anticipate that such an isolation strategy will effectively maintain per-view semantic consistency and simultaneously enhance the visual quality. Finally, we fuse edited appearance features with rendered 3D-consistent semantic features and inject them into selected intermediate layers of the diffusion model.

Experimentally, our method achieves the state-of-the-art CLIP similarity score. More importantly, as shown in Figure 1, it effectively mitigates semantic distortions such as the Janus problem, preserving source identities across views. Additionally, we quantify semantic consistency by measuring the preservation of feature correspondences between image pairs before and after 3D editing, where our method achieves the highest correspondence score among existing approaches.

2 Related work

Diffusion-based text-based editing Recently, with the rapid progress of text-based image generation, text-based image editing has also advanced significantly. DiffusionCLIP [17] integrates CLIP-based optimization to refine edits while preserving identity. Prompt-to-Prompt (P2P) [10] introduces a method that leverages source attention maps to guide cross-attention manipulation, enabling precise and structure-preserving modifications through prompt changes. Plug-and-Play (PnP) [35] presents a framework that injects spatial features from a source image to achieve controlled image-to-image translation guided by text prompts while preserving the source layout. Recent approaches like InstructPix2Pix [1] fine-tune models on synthetic paired datasets generated using Prompt-to-Prompt [10], allowing diverse and complex edits through a single forward pass.

Building on text-guided 2D editing techniques, most existing 3D editing approaches operate by applying edits independently to per-view renderings of the source scene, which often leads to multi-view inconsistencies. In contrast, we learn a geometry-aligned, 3D consistent feature field and inject its rendered features during the editing process, thereby enforcing consistency across viewpoints.

Text-based 3D scene editing Many existing text-based 3D scene editing methods build upon text-based 2D image editing techniques. Instruct-NeRF2NeRF [9] applies iterative view-by-view editing using a 2D diffusion model, followed by NeRF [21] optimization. Subsequent studies [32, 5, 40] have proposed various approaches to improve multi-view consistency and enhance the quality of 3D scene editing.

More recently, with the emergence of 3D Gaussian Splatting (3DGS) as a new 3D representation characterized by its remarkably fast rendering speed, several methods have been introduced to enable text-based editing on 3DGS. GaussianEditor [4] proposes a 3D Gaussian splatting editing framework enabling fast and controllable object manipulation through semantic tracing and hierarchical representations. Subsequent works such as Direct Gaussian Editor (DGE) [3] further improved multi-view consistency through geometry-aware edits, however, it is limited to adjacent views. The works of GaussCtrl [38] and VcEdit [37] propose novel methods to enforce multi-view consistency, however not considering the semantic consistency across views. Our DFFSplat framework addresses these limitations by explicitly encoding structural consistency within a learned 3D diffusion feature field, significantly improving semantic and multi-view coherence.

3 Preliminary

3.1 Representation of feature field in 3DGS

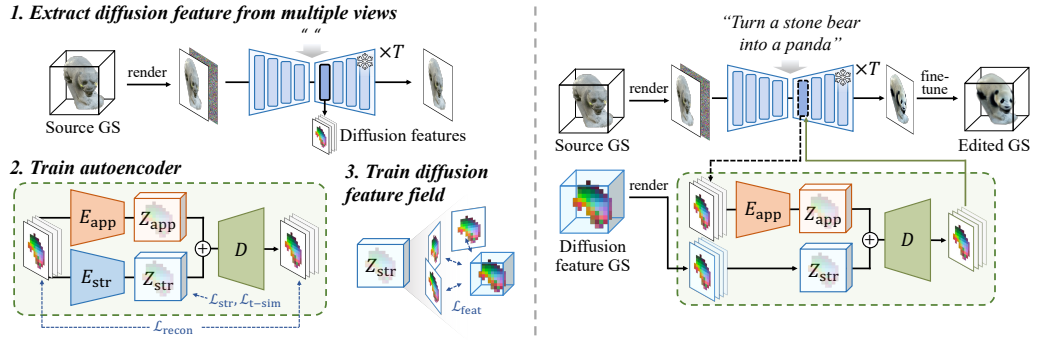
Recent active research on 3D scene understanding [18, 6, 16, 19, 20, 31, 39, 33, 8] aims to embed semantic features into 3DGS [25, 42, 26] by assigning learnable semantic vectors to each Gaussian and training them to match ground-truth semantic maps. The rendering process of semantic features can be formulated analogously to that of RGB images $\mathbf{I} \in \mathbb{R}^{3\times H\times W}$, where H, W represent the height and width of the image size, as follows:

$$C = \sum_{i\in\mathcal{G}} c_i \alpha_i \mathcal{T}_i, \quad F = \sum_{i\in\mathcal{G}} f_i \alpha_i \mathcal{T}_i, \tag{1}$$

where $\mathcal{G}$ is the ordered set of Gaussians contributing to a given pixel. C and F are the rendered RGB color and semantic feature vector, and c_i and f_i are color and semantic features of the i-th Gaussian, respectively. α_i is the product of a learnable opacity parameter and the Gaussian's projected density at the pixel, and $\mathcal{T}_i$ is the accumulated transmittance up to i-th Gaussian, defined as $\mathcal{T}_i = \prod_{j=1}^{i-1}(1-\alpha_j)$. Unlike RGB values c, which are view-dependent and thus require spherical harmonic (SH) coefficients conditioned on viewing direction, semantic features f are inherently view-independent. This eliminates the need for additional SH coefficients during the rendering.

In addition, recent practices [25] utilize a low-dimensional semantic latent compressed by an autoencoder as ground-truth semantic features for reducing memory consumption. This process can be formulated as below:

$$Z = E(F), \quad \hat{F} = D(Z), \tag{2}$$



(a) Training process of autoencoder and diffusion feature field (b) Editing process of GS

Figure 2: Overview of the proposed method. (a) To obtain 3D-consistent diffusion features, we train a 3D diffusion feature field. However, rendering these features for editing averages out per-view details and sacrifices fine textures. To prevent this, we introduce a dual-encoder scheme that decomposes structural and appearance features at the autoencoder level. Then, we train the feature field exclusively on these structural features, which have multi-view consistent geometry. (b) For 3D editing, we edit the multi-view source images by injecting the rendered structural features from our learned feature field. By keeping the appearance feature from images being edited, the results remain 3D consistent while faithfully reflecting the user-given edit text.

where $E(\cdot), D(\cdot)$ are encoder and decoder, and $Z, F, \hat{F}$ are their compressed semantic features, uncompressed and reconstructed counterparts, respectively. The training objective for the autoencoder minimizes the distance between reconstructed and ground-truth features, while the 3DGS objective minimizes the distance between rendered and compressed features.

4 3D Gaussian Splatting editing via diffusion feature field

In this paper, we focus on addressing the inconsistencies frequently encountered in existing text-based 3D editing approaches, such as the Janus problem, where independently edited multi-view images fail to maintain consistent geometry and semantics.

To overcome these issues, we propose a novel editing framework that constructs a 3D-consistent diffusion feature field, which is then rendered and injected into the editing process to ensure multi-view and semantic consistency in the edited images. Furthermore, we observe that directly using the learned 3D features in editing causes blurriness in the edited images, as its per-view appearance information are averaged while building 3D feature field. To resolve this, we introduce a dual-encoder architecture that separates structural features from appearance information, and only embeds multi-view consistent structural information within the 3D feature field. Finally, we elaborate how our 3D-consistent structural features are integrated into the 2D diffusion editing pipeline and the subsequent 3D reconstruction process.

4.1 3D-consistent diffusion feature field for semantic consistent editing

Diffusion features inherently encode structural semantics within their intermediate representations [35], enabling explicit control over the layout of generated images during editing. Therefore, by constructing diffusion features that are coherent across multiple views (i.e., 3D-consistent), directly injecting these coherent features into a per-view 2D diffusion editing model can effectively produce geometrically view-consistent edited images.

To achieve this, we first construct a diffusion feature field by aligning and fusing per-view diffusion features extracted from multi-view images of the source 3D scene. Once trained, this feature field provides 3D-consistent diffusion features, which we render and inject at key layers within the diffusion model. This explicitly encourages geometric consistency across multi-view images and semantic fidelity to the original content from source throughout the editing process. This strategy proves effective because diffusion features extracted from multi-view images inherently possess a degree of semantic coherence. Moreover, this 3D diffusion feature field also facilitates the correction of inconsistent or semantically biased features that may arise in individual views. Such inconsistencies

are typically (1) view-dependent and (2) relatively sparse across the entire set of viewpoints. Therefore, by explicitly averaging structural information across multiple views, the 3D feature field can correct biased views by utilizing structural information from unbiased views. This capability of cross-view averaging offered by the 3D feature field effectively resolves these biases.

Following prior work [25, 42, 26], we first train the 3D diffusion feature field on the source 3DGS, as detailed in Sec. 3.1. However, diffusion features differ notably from typical vision model features due to the presence of a timestep t. Thus, we construct a timestep-conditioned encoder E_{base} which has inputs as a diffusion feature of specific timestep F^t and its corresponding timestep t. This process can be formulated as below:

$$Z^t_{\text{base}} = E_{\text{base}}(F^t, t) \tag{3}$$

where Z^t_{base} is compressed latent of F^t by the encoder E_{base}. However, training 3D feature field based on this naïve encoder has several issues as noted in below section, so we introduce a dual-encoder scheme.

4.2 Dual-encoder scheme for decomposing structure-appearance features

By rendering features from our 3D diffusion feature field, we inherently obtain 3D-consistent features. However, aggregating information across multiple viewpoints causes per-view appearance details to become averaged out. Consequently, we observe that directly injecting these averaged features during editing results in blurred outputs with insufficient fine textures. This averaging issue is further exacerbated by the inherently time-variant nature of diffusion features during the denoising process, making it challenging to accurately represent these features within conventional 3D feature field. Consequently, directly injecting these averaged features during editing results in blurred outputs lacking sufficient fine textures. Although such representations might be acceptable in 3D recognition tasks [25, 42], which only need view-independent semantic information, they significantly limit editing applications, where preserving visual detail and fidelity to user-provided instruction is crucial.

To address this limitation, we explicitly separate appearance information (fine, per-view textures, and time-variant) from structural information (geometric, viewpoint-consistent, and time-invariant). We first train the diffusion feature field of 3DGS exclusively on structural information to ensure robust geometric consistency across views. During editing, we fuse the rendered features from this field with appearance information specifically generated to align with user-provided prompts. This combined representation enables our method to produce semantic-consistent edits, accurately reflecting user inputs while effectively preserving viewpoint-specific appearance details.

To this end, we propose a dual-encoder scheme that separates appearance and structural features at the encoder level. Specifically, the encoder is divided into two distinct branches: a structure encoder E_{str} and an appearance encoder E_{app}. Following conventional autoencoder setups, the outputs from these two encoders are combined and subsequently processed through a shared decoder for reconstruction. This process can be formally described as follows:

$$Z^t_{\text{app}} = E_{\text{app}}(F^t, t), \quad Z^t_{\text{str}} = E_{\text{str}}(F^t, t), \quad \hat{F}^t = D(Z^t_{\text{app}}, Z^t_{\text{str}}), \tag{4}$$

where Z^t_{str} and Z^t_{app} are structural and appearance features encoded by corresponding encoders, respectively.

Then, in training phase, we explicitly differentiate the roles of the two encoders by imposing an additional constraint features to learn on the latent representation of the structure encoder. Specifically, we leverage the self-similarity maps derived from DINOv2 [2, 23] features, which have demonstrated effectiveness in capturing structural information [34]. These self-similarity map serve as an auxiliary objective by minimizing their difference from the self-similarity maps generated from the structure encoder's latent output. Due to the inherent information-bottleneck characteristic of autoencoders, enforcing structural constraints within the structure encoder naturally encourages the complementary appearance encoder to capture per-view, appearance-specific details. This structure similarity loss is formulated as follows:

$$\mathcal{L}_{\text{str}} = \left\| S(F_{\text{dino}}(\mathbf{I})) - S(Z^t_{\text{str}}) \right\|_2, \text{where } S(K)_{ij} = \text{cos-sim}(K_i, K_j), \tag{5}$$

where t is sampling from $\mathcal{I} \sim (T/n, 2T/n, \ldots, T)$ where T is the number of diffusion timesteps and n is the number of timestep interval, cos-sim$(\cdot, \cdot)$ is the cosine similarity between two feature maps and $S(K) \in \mathbb{R}^{hw \times hw}$ is a self-similarity map of a given feature map $K \in \mathbb{R}^{c \times hw}$, where

$i, j \in \{1, ..., hw\}$. By leveraging structure information extracted from the rendered image $\mathbf{I}$ of the source 3D scene using a DINOv2 extractor F_{dino}, E_{str} effectively captures semantic information. To match the spatial resolution with DINOv2 features, we interpolate the DINOv2 features to the same resolution as the structure encoder's latent output.

We further enforce that the structure encoder's latent outputs at different timesteps remain consistent, ensuring they can be represented by a single, static 3D scene. These constraints can be formulated as below:

$$\mathcal{L}_{\text{t-sim}} = \frac{1}{n(n-1)} \sum_{t,t' \in \mathcal{I}} \mathbf{1}_{[t' \neq t]} \left\| Z^t_{\text{str}} - Z^{t'}_{\text{str}} \right\|_2. \tag{6}$$

$\mathcal{L}_{\text{str}}$ is a structure similarity loss that imposes the latent of structure encoder Z_{str} to have structural information mostly, and $\mathcal{L}_{\text{t-sim}}$ is a time-similarity loss that encourages the structure encoder's latent representations to be invariant across diffusion timesteps.

Training 3D diffusion feature field As shown in Figure 2-(a), we train our 3D diffusion feature field using the dual-encoder scheme described in Sec. 3.1. Concretely, we optimize an autoencoder with three loss terms: (1) a standard reconstruction loss to recover per-view diffusion features, (2) a structural similarity loss $\mathcal{L}_{\text{str}}$ based on DINOv2 self-similarity maps, and (3) a time-similarity loss $\mathcal{L}_{\text{t-sim}}$ to enforce invariance across diffusion timesteps. In addition, we assemble the structure encoder's latent outputs Z^t_{str} over all timesteps $t \in [0, T]$ for each source view, average them to form a static 3D target feature, and render this target via our 3DGS parameterization. We then train the 3DGS parameters f by minimizing the feature-map discrepancy between the rendered feature map $\tilde{F}$ and the time-averaged target $\bar{Z}_{\text{str}}$, thereby grounding our 3D field in multi-view, time-invariant structural information. This process can be formulated as below:

$$\mathcal{L}_{\text{ae}} = \mathcal{L}_{\text{recon}} + \lambda_1 \mathcal{L}_{\text{str}} + \lambda_2 \mathcal{L}_{\text{t-sim}}, \quad \mathcal{L}_{\text{recon}} = \tfrac{1}{T} \textstyle\sum_{t \in [0,T]} d(\hat{F}^t, F^t), \tag{7}$$

$$\mathcal{L}_{\text{feat}} = d(\tilde{F}, \bar{Z}_{\text{str}}), \quad \text{where} \quad \bar{Z}_{\text{str}} = \tfrac{1}{T} \textstyle\sum_{t \in [0,T]} Z^t_{\text{str}}, \tag{8}$$

where $d(\cdot, \cdot)$ is a distance function (e.g., L2 distance), and $\mathcal{L}_{\text{ae}}$ and $\mathcal{L}_{\text{feat}}$ are objectives of autoencoder and feature of 3DGS, respectively, and λ_1, λ_2 are the strengths of regularization for structure and time-invariance.

4.3 3D Gaussian Splatting editing via view-consistency guidance

In this section, we describe how the trained diffusion feature field of 3DGS and the dual-encoder mechanism are leveraged to effectively obtain multi-view edited images. In particular, given an instruction prompt from the user, our primary goal during editing is to preserve the appearance information corresponding to the edit, while ensuring semantic consistency with the source scene.

To achieve this, we first encode the features obtained during the editing process using the appearance encoder E_{app}. Then, to enforce semantic consistency with the source, we simultaneously render view-consistent semantic features from the 3DGS semantic feature field, thereby providing essential source semantic information. These rendered 3D-consistent semantic features are subsequently fused with the encoded appearance features and then passed through the decoder to obtain reconstructed diffusion features. Finally, by injecting these diffusion features into the original editing pipeline (i.e., by replacing the model's native features with our diffusion features at selected intermediate layers), we produce multi-view edited images that faithfully reflect the user's prompt while accurately preserving the source semantics (Figure 2-(b)). This process is formulated as below:

$$\hat{F}^t_{\text{injected}} = D(E_{\text{app}}(F^t, t), \tilde{F}), \tag{9}$$

where $\hat{F}^t_{\text{injected}}$ is modified diffusion feature during editing, which contains 3D-consistent semantic information and appearance of editing condition.

Because this injection effectively supplies a structure-guided conditioning, we augment the original guidance [11] in InstructPi2Pix [1] with an additional "view-consistency guidance" term. This combined guidance ensures that edits not only follow the user's text instructions but also remain geometrically coherent across all views. This process is defined as below:

$$\begin{aligned} &\hat{e}_\theta(z_t, c_I, c_S, c_T) = e_\theta(z_t, \phi, \phi, \phi) + s_I \cdot (e_\theta(z_t, c_I, \phi, \phi) - e_\theta(z_t, \phi, \phi, \phi)) \\ &+ \underline{s_S \cdot (e_\theta(z_t, c_I, c_S, \phi) - e_\theta(z_t, c_I, \phi, \phi))} + s_T \cdot (e_\theta(z_t, c_I, \underline{c_S}, c_T) - e_\theta(z_t, c_I, \underline{c_S}, \phi)) \end{aligned} \tag{10}$$

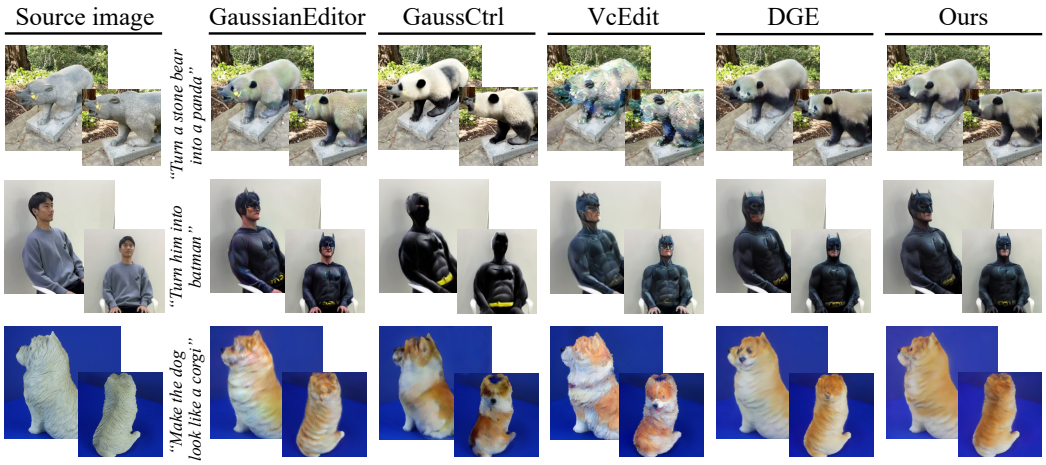


Figure 3: Qualitative comparison of edited 3D scene. While existing methods often suffer from multi-view inconsistencies due to the inherent frontal-view bias of diffusion models, causing severe artifacts in the edited 3DGS outputs, our approach effectively preserves the original source structure and achieves faithful edits aligned with the input text prompt.

where e_θ and $\hat{e}_\theta$ is a diffusion model and its guided prediction, and z_t is an input noised latent, and c_I and c_S and c_T are the conditions about image and structure and text, and $s_{(\cdot)}$s are a strength of guidance. Underlined notations denote our modification, the "view-consistency guidance" we additionally use.

5 Experiments

5.1 Implementation details

We conduct the experiments on 5 scenes from IN2N [9] and DreamEditor [43], including complex 360° scenes. To validate the effectiveness of the proposed methods, we compare ours with previous 3DGS editing methods [4, 38, 37, 3]. For a fair comparison, we use InstructPix2Pix as an image editor, which is the most popular editor used by prior methods [4, 3]. During editing, we utilize classifier-free guidance with scales of 7.5 for textual conditions and 1.5 for image conditions, unless specified otherwise. For each scene, we edit 20 views of the source scene and use the edited images as the reconstruction target to update 3DGS. For training details of 3DGS, we iteratively update the parameters of 3DGS for a total of 1000 iterations, while image editing is performed every 500 iterations. We use Adam optimizer to update the 3DGS parameters and LPIPS [41] and L1 loss as reconstruction objectives, following IN2N [9]. For training our dual-encoder, we set $\lambda_1 = 1.0$ and $\lambda_2 = 0.05$. We further elaborate on the additional implementation and details in supplementary.

Evaluation metric Following the previous baselines [9, 37, 3], we employ CLIP similarity and CLIP directional similarity [17, 7] as evaluation metrics for quantitative comparison. CLIP similarity is computed as the cosine similarity between text and image embeddings, and clip directional similarity measures the difference in embedding directions from the target embedding to the source embedding between the image and text modalities. Both metrics evaluate how well the edited 3DGS reflects the target prompt by comparing image and text embeddings, but clip directional similarity is known for its robustness to mode collapse. We use every image available in the training dataset for evaluation, although only use 20 views of images for editing.

5.2 Comparisons with previous works

Quantitative comparison with previous works We first compare CLIP and its directional similarity to measure the text fidelity of the edited scene. As reported in Table 1, the proposed method achieves significantly higher scores in both CLIP and clip directional similarity. Furthermore, ours reports the lowest editing time, similar to prior arts [3].

Table 1: Quantitative comparison. CLIP and clip dir. denotes CLIP and clip directional similarities, respectively.

Method	CLIP	clip Dir.	Avg. Time
GaussianEditor [4]	24.40	16.67	∼7min
GaussCtrl [38]	22.87	13.17	∼8min
VcEdit [37]	23.18	14.03	∼20min
DGE [3]	24.66	17.21	∼4min
Ours	**25.11**	**18.15**	∼4min

Table 2: Ablation study of proposed components and 3D feature field, showing complementary gains in text–image alignment.

Method	CLIP	CLIP Dir.
Ours (Full method)	**25.11**	**18.15**
- $\mathcal{L}_{\text{t-sim}}$	25.09	18.09
- E_{app}	24.80	17.21
Ours (Full method)	**25.11**	**18.15**
- 3D feature field	25.00	17.82

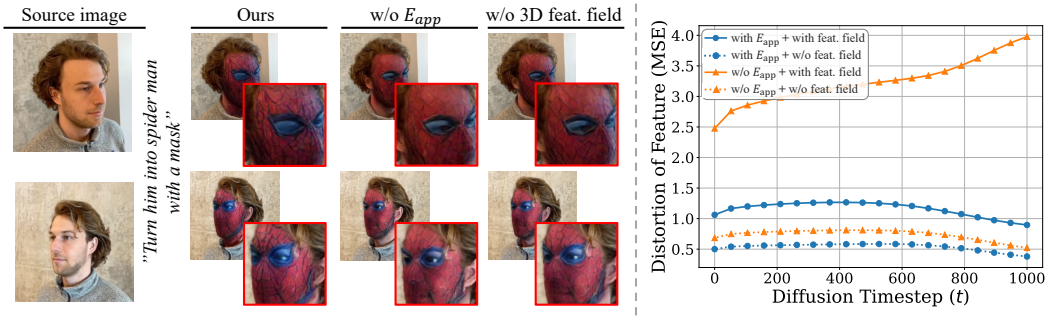


Figure 4: Ablation study. (Left) Qualitative comparison of ablated models. (Right) Measuring distortion between the reconstructed diffusion feature ($\hat{F}^t$) and the modified diffusion feature ($\hat{F}^t_{\text{inject}}$) across ablated models. Lower distortion indicates better preservation of source representation. The presence of an appearance encoder (E_{app}) reduces the distortion introduced by the 3D diffusion feature field ("feat. field" in the legend), demonstrating our dual-encoder effectively preserves information.

Qualitative comparison with previous works Qualitative results are shown in Figure 3. The proposed method produces much fewer artifacts in the rendered images from edited scene. In particular, we observe the edited results from ours achieve a prominent semantic consistency, preserving the composition and semantics of the source object. It proves the effectiveness of learning the underlying structure of the source 3D scene via our 3D diffusion feature field.

For example, in the "Turn a stone bear into a panda" instruction (1st row), competing methods (e.g., VcEdit, DGE) fail to correct the diffusion model's frontal-view bias, resulting in multi-view inconsistencies that synthesize visible artifacts in the edited 3DGS. In contrast, ours successfully edit the 3D scene without such artifacts, demonstrating the coherence of edited images across all views. Moreover, in the "Turn him into Batman" prompt (2nd row), our approach better preserves the source geometry than DGE, demonstrating stronger semantic fidelity to the original 3D scene. Finally, as shown in the last row, unlike other methods that exhibit strong frontal bias, our approach does not display such bias even on the backside of the object.

5.3 Analysis on the proposed components

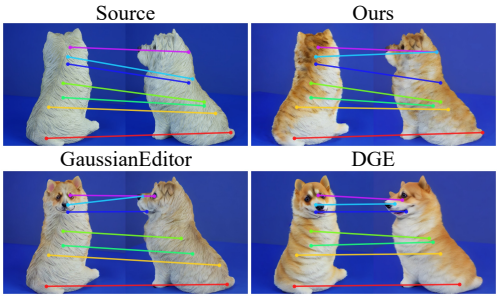
Ablation study We perform an ablation study on our key proposed components, specifically the dual-encoder scheme with structural loss (E_{app} and $\mathcal{L}_{\text{str}}$) and the time-invariance objective ($\mathcal{L}_{\text{t-sim}}$). Note that a naïve decomposition of encoders without the structural loss does not effectively separate learned representations; thus, we ablate these components together.

As reported in Table 2, the dual-encoder scheme substantially impacts performance, improving the clip directional similarity. This emphasizes the importance of explicitly decomposing structural and appearance information to achieve detailed, text-consistent 3D edits. Visual examples comparing our final model to its ablations are provided on the left side of Figure 4. Without the structure decomposition, the edited results lose per-view details, becoming overly blurred due to averaged features during editing.

We also ablate the time-invariance objective $\mathcal{L}_{\text{t-sim}}$ and quantitatively confirm that it boosts both the CLIP similarity and directional scores. Additionally, we evaluate the significance of learning the 3D

Table 3: Difference in correspondence matching map between multi-view source and edited images from complex 360° scenes. Our method demonstrates superior editing performance under sparse-view conditions compared to others.

Method	Bear scene	Dog scene
GaussianEditor [4]	5.08	8.01
GaussCtrl [38]	10.91	8.15
VcEdit [37]	6.02	6.59
DGE [3]	4.14	7.36
Ours	**3.93**	**5.53**



"Make the dog look like a corgi"

Figure 5: Correspondence matching comparison between two edited views of the dog scene.

diffusion feature field itself. That is, we directly inject the original 2D feature into the editing process, instead of rendering the 3D feature field. As shown in Table 2, omitting this field ("-3D feature field" row) significantly decreases text fidelity, as the edited images lose multi-view consistency and appear more blurred during 3D reconstruction process. Qualitative comparisons in Figure 4 further illustrate the necessity of the learned 3D field for achieving detailed textures.

Finally, we measure the distortion caused by feature injection in the reconstructed diffusion features ($\hat{F}^t_{\text{inject}}$) across our ablated models. Note that, we give null text as an editing condition for only validating the distortion caused by feature injection, not the editing capability. The lower distortion indicates the injected feature $\tilde{F}$ does not lose its fine-details, as it can recover these details at the reconstructed feature $\hat{F}^t_{\text{inject}}$. As illustrated in Figure 4, we conducted an ablation study on the presence or absence of the dual-encoder scheme (specifically, the appearance encoder, E_{app}) and the 3D diffusion feature field. The presence of E_{app} reduces the distortion caused by the 3D diffusion feature field, demonstrating that our dual-encoder effectively preserves information. Additionally, in the case without E_{app} but with the 3D diffusion feature field, the distortion increases as the timestep (noise-level) increases, indicating that the 3D diffusion feature field struggles to accurately retain time-invariant noise information.

Evaluating multi-view and semantic consistency via correspondence matching Previous experiments mainly evaluated fidelity to the input text prompts. Here, we further assess how well the edited results preserve the original structure and resolve known editing artifacts, such as the Janus problem. To achieve this, we compute pixel-level correspondence matching maps among multi-view images rendered from both the original and edited 3D scenes, and then measure the difference between these maps. A smaller difference indicates better preservation of the source scene's geometry and improved multi-view consistency due to accurate pixel-level matching across views.

Quantitative results for correspondence matching differences are presented in Table 3. As shown, our proposed method achieves significantly lower distortion in correspondence matching compared to baselines. Qualitative visualizations of the correspondence maps are provided in Figure 5, where baseline methods show noticeable artifacts (e.g., generating faces on the back of an animal), indicating failures in preserving source structure. In contrast, our approach consistently preserves the original structure, resulting in highly accurate correspondence maps.

Analyze the impacts of 3D feature field via correspondence matching To further analyze the individual impacts of DINO regularization ($\mathcal{L}_{\text{str}}$) and the 3D feature field, we conducted the correspondence matching experiment by selectively removing each component. Specifically, we compared scenarios without DINO regularization and without the 3D feature field, respectively. Since the appearance encoder cannot be trained effectively without DINO regularization, we performed the comparative experiments excluding the appearance encoder. In Table 4, we observe that excluding the 3D feature field results in a significantly larger difference in the correspondence matching map compared to excluding DINO regularization. This clearly indicates that the 3D feature field substantially contributes to improving multi-view consistency and semantic consistency, rather than DINO regularization.

Table 4: Correspondence matching score without DINO regularization or 3D feature field.

Method	Bear scene	Dog scene
w/o E_{app}	2.96	5.28
w/o E_{app} + w/o DINO regularization	3.02	5.21
w/o E_{app} + w/o 3D feature field	3.15	6.10

Figure 6: Edited 3DGS results across iterations. While our baseline, GaussianEditor, suffers from the Janus problem, our method effectively mitigates it.

Visualization of the edited 3DGS over optimization iterations We additionally visualize the evolution of the edited 3DGS over the course of optimization iterations. As shown in Figure 6, our baseline method (GaussianEditor [4]) suffers from the Janus problem, where undesired facial attributes emerge on the backside of the bear at very early iterations and gradually intensify as optimization progresses. In contrast, our proposed method effectively suppresses such artifacts throughout the optimization process, maintaining consistent and faithful editing results across all viewpoints.

6 Limitation

Our proposed editing framework is inherently constrained by the editing capability of InstructPix2Pix; therefore, it cannot handle 3DGS editing cases that go beyond the editing scope defined by InstructPix2Pix. This dependency fundamentally limits the diversity and complexity of the edits that can be achieved within our framework. Furthermore, since our method places a strong emphasis on preserving the structural integrity of the source 3DGS, it is less suitable for tasks that require substantial geometric transformations. As a result, editing scenarios involving large pose changes, object repositioning, or major structural alterations remain challenging. Addressing these limitations, either by incorporating more flexible editing modules or by enabling geometry-aware transformations, will be an important direction for our future work.

7 Conclusion

We introduced DFFSplat, a novel framework for text-based 3D editing that significantly improves multi-view and semantic consistency by incorporating a 3D-consistent diffusion feature field. By leveraging a dual-encoder architecture to disentangle structural and appearance information, along with enforcing time-invariance across diffusion steps, our method effectively addresses prevalent artifacts such as the Janus problem. Experimental results confirm that our approach outperforms prior methods in terms of CLIP similarity and semantic coherence, highlighting its potential for high-quality 3D content editing.

Acknowledgements

This work was supported in part by MSIT/IITP (No. RS-2022-II220680, RS-2020-II201821, RS-2019-II190421, RS-2024-00459618, RS-2024-00360227, RS-2024-00437633, RS-2024-00437102, RS-2025-25442569), MSIT/NRF (No. RS-2024-00357729), and KNPA/KIPoT (No. RS-2025-25393280).

References

[1] T. Brooks, A. Holynski, and A. A. Efros. Instructpix2pix: Learning to follow image editing instructions. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 18392–18402, 2023.

[2] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In Proceedings of the IEEE/CVF international conference on computer vision, pages 9650–9660, 2021.

[3] M. Chen, I. Laina, and A. Vedaldi. Dge: Direct gaussian 3d editing by consistent multi-view editing. In European Conference on Computer Vision, pages 74–92. Springer, 2024.

[4] Y. Chen, Z. Chen, C. Zhang, F. Wang, X. Yang, Y. Wang, Z. Cai, L. Yang, H. Liu, and G. Lin. Gaussianeditor: Swift and controllable 3d editing with gaussian splatting. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 21476–21485, 2024.

[5] J. Dong and Y.-X. Wang. Vica-nerf: View-consistency-aware 3d editing of neural radiance fields. Advances in Neural Information Processing Systems, 36:61466–61477, 2023.

[6] Z. Fan, P. Wang, Y. Jiang, X. Gong, D. Xu, and Z. Wang. Nerf-sos: Any-view self-supervised object segmentation on complex scenes. arXiv preprint arXiv:2209.08776, 2022.

[7] R. Gal, O. Patashnik, H. Maron, A. H. Bermano, G. Chechik, and D. Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. ACM Transactions on Graphics (TOG), 41(4):1–13, 2022.

[8] R. Goel, D. Sirikonda, S. Saini, and P. Narayanan. Interactive segmentation of radiance fields. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4201–4211, 2023.

[9] A. Haque, M. Tancik, A. A. Efros, A. Holynski, and A. Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 19740–19750, 2023.

[10] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or. Prompt-to-prompt image editing with cross attention control. arXiv preprint arXiv:2208.01626, 2022.

[11] J. Ho and T. Salimans. Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598, 2022.

[12] S. Hong, D. Ahn, and S. Kim. Debiasing scores and prompts of 2d diffusion for view-consistent text-to-3d generation. Advances in Neural Information Processing Systems, 36:11970–11987, 2023.

[13] H. Kamata, Y. Sakuma, A. Hayakawa, M. Ishii, and T. Narihira. Instruct 3d-to-3d: Text instruction guided 3d-to-3d conversion. arXiv preprint arXiv:2303.15780, 2023.

[14] B. Kawar, S. Zada, O. Lang, O. Tov, H. Chang, T. Dekel, I. Mosseri, and M. Irani. Imagic: Text-based real image editing with diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 6007–6017, 2023.

[15] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. ACM Trans. Graph., 42(4):139–1, 2023.

[16] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik. Lerf: Language embedded radiance fields. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 19729–19739, 2023.

[17] G. Kim, T. Kwon, and J. C. Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 2426–2435, 2022.

[18] S. Kobayashi, E. Matsumoto, and V. Sitzmann. Decomposing nerf for editing via feature field distillation. Advances in neural information processing systems, 35:23311–23330, 2022.

[19] K. Liu, F. Zhan, J. Zhang, M. Xu, Y. Yu, A. El Saddik, C. Theobalt, E. Xing, and S. Lu. Weakly supervised 3d open-vocabulary segmentation. Advances in Neural Information Processing Systems, 36:53433–53456, 2023.

[20] K. Mazur, E. Sucar, and A. J. Davison. Feature-realistic neural fusion for real-time, open set scene understanding. In 2023 IEEE International Conference on Robotics and Automation (ICRA), pages 8201–8207. IEEE, 2023.

[21] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM, 65(1):99–106, 2021.

[22] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. arXiv preprint arXiv:2112.10741, 2021.

[23] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193, 2023.

[24] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. arXiv preprint arXiv:2209.14988, 2022.

[25] M. Qin, W. Li, J. Zhou, H. Wang, and H. Pfister. Langsplat: 3d language gaussian splatting. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 20051–20060, 2024.

[26] R.-Z. Qiu, G. Yang, W. Zeng, and X. Wang. Feature splatting: Language-driven physics-based scene synthesis and editing. arXiv preprint arXiv:2404.01223, 2024.

[27] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125, 1(2):3, 2022.

[28] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 10684–10695, 2022.

[29] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. Advances in neural information processing systems, 35:36479–36494, 2022.

[30] E. Sella, G. Fiebelman, P. Hedman, and H. Averbuch-Elor. Vox-e: Text-guided voxel editing of 3d objects. In Proceedings of the IEEE/CVF international conference on computer vision, pages 430–440, 2023.

[31] N. M. M. Shafiullah, C. Paxton, L. Pinto, S. Chintala, and A. Szlam. Clip-fields: Weakly supervised semantic fields for robotic memory. arXiv preprint arXiv:2210.05663, 2022.

[32] L. Song, L. Cao, J. Gu, Y. Jiang, J. Yuan, and H. Tang. Efficient-nerf2nerf: streamlining text-driven 3d editing with multiview correspondence-enhanced diffusion models. arXiv preprint arXiv:2312.08563, 2023.

[33] N. Tsagkas, O. Mac Aodha, and C. X. Lu. Vl-fields: Towards language-grounded neural implicit spatial representations. arXiv preprint arXiv:2305.12427, 2023.

[34] N. Tumanyan, O. Bar-Tal, S. Bagon, and T. Dekel. Splicing vit features for semantic appearance transfer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10748–10757, 2022.

[35] N. Tumanyan, M. Geyer, S. Bagon, and T. Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1921–1930, 2023.

[36] T. Ukarapol and K. Pruvost. Gradeadreamer: Enhanced text-to-3d generation using gaussian splatting and multi-view diffusion. arXiv preprint arXiv:2406.09850, 2024.

[37] Y. Wang, X. Yi, Z. Wu, N. Zhao, L. Chen, and H. Zhang. View-consistent 3d editing with gaussian splatting. In European Conference on Computer Vision, pages 404–420. Springer, 2024.

[38] J. Wu, J.-W. Bian, X. Li, G. Wang, I. Reid, P. Torr, and V. A. Prisacariu. Gaussctrl: Multi-view consistent text-driven 3d gaussian splatting editing. In European Conference on Computer Vision, pages 55–71. Springer, 2024.

[39] J. Ye, N. Wang, and X. Wang. Featurenerf: Learning generalizable nerfs by distilling foundation models. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 8962–8973, 2023.

[40] L. Yu, W. Xiang, and K. Han. Edit-diffnerf: Editing 3d neural radiance fields using 2d diffusion model. arXiv preprint arXiv:2306.09551, 2023.

[41] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 586–595, 2018.

[42] S. Zhou, H. Chang, S. Jiang, Z. Fan, Z. Zhu, D. Xu, P. Chari, S. You, Z. Wang, and A. Kadambi. Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 21676–21685, 2024.

[43] J. Zhuang, C. Wang, L. Lin, L. Liu, and G. Li. Dreameditor: Text-driven 3d scene editing with neural fields. In SIGGRAPH Asia 2023 Conference Papers, pages 1–10, 2023.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduce well.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitation of this work would be in Supplementary.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We include quantitative and qualitative comparison with previous works.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The implementation details of our main experimental results would be in Supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We will release the code and data after accept.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide implementation details in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification:

   Guidelines: Could not report due to computation cost. Will report next time.

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The experiments compute resources would be in supplementary.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform with NeurIPS code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss this in the supplementary.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We will include this in the supplementary.

    Guidelines:

    - The answer NA means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We attribute the original oweners of the data and code.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [No]

    Justification: We will provide an anonymized URL for the required content after accept.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing experiments.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.