
3D Gaussian Splatting Editing with Diffusion Personalization

Wonsik Shin

Carnegie Mellon University
wonsiks@andrew.cmu.edu

Jessica Ruan

Carnegie Mellon University
jwruan@andrew.cmu.edu

Aradhya Talan

Carnegie Mellon University
atalan@andrew.cmu.edu

Brandon Dong

Carnegie Mellon University
bjdong@andrew.cmu.edu

1 Abstract

We propose an image-based 3D Gaussian Splat editing method that leverages Diffusion Personalization, a technique that trains diffusion models to recognize specific objects by introducing new concepts as new concept-token pairs to the model. Our approach combines personalized text-to-image diffusion models with 3D Gaussian Splats, enabling precise geometric editing while maintaining photorealistic quality and subject fidelity. By fine-tuning a pretrained diffusion model on a small set of reference images, we create a personalized model capable of generating novel views of specific subjects. We then leverage this model in the GaussCtrl framework to perform edits to 3DGS. Our method enables various editing operations, including object insertion, removal, and style transfer, while maintaining consistent lighting and perspective across multiple viewpoints. Experimental results demonstrate superior visual quality compared to existing methods, with particular strength in preserving subject identity and geometric consistency. We validate our approach on a diverse set of scenes and editing tasks, showing its practical applicability for both indoor and outdoor environments.

2 Introduction

Recent advancements in 3D representations, such as Neural Radiance Fields (NeRF) [5] and 3D Gaussian Splatting (3DGS) [4], have significantly improved the speed and accuracy of generating 3D models from 2D images. Building on these developments, recent research has increasingly focused on editing 3D representations using generative models, particularly Diffusion Models. Instruct-NeRF2NeRF [2], a pioneering method for 3D editing based on NeRF, was the first to propose such a task. The method involves rendering images from NeRF, editing these images using a Diffusion Model, and then feeding the edited images back into the NeRF model for further training. This pipeline has since been adopted and extended by many 3D editing methods [7, 9] and has even become a foundational approach for 3DGS-based editing methods [10, 11].

However, the approaches proposed so far [2, 10, 11] have primarily focused on text-prompt-based editing, making it impossible to edit the original 3D representation based on specific objects present in a given image. Additionally, since these methods employ Diffusion-based image editing [3] across multiple viewpoints, there is a high possibility of inconsistency when significant changes occur during editing, as each rendered image might be edited differently. VCEdit [10], a text-based editing method, attempted to address the issues that arise when editing 3DGS by introducing a Cross-Attention Consistency Module and an Editing Consistency Module. However, it has the limitation that inconsistencies within the 3DGS model emerge when the editing scope becomes larger. Similarly, GaussCtrl [11], another text-based editing method, extracts depth maps from the original 3DGS for

use during editing to maintain consistency in the edited model. However, this approach has the drawback of restricting the editing results from deviating significantly from the original shape.

We propose an image-based 3D editing method that leverages Diffusion Personalization. Diffusion Personalization is a technique that trains Diffusion models to recognize specific objects by introducing new concepts as new concept-token pairs to the model[8]. Using this approach, we aim to edit 3D representations by first training the Diffusion model on specific objects present in a given image and then applying the model to edit the 2D images encoded from 3DGS. This allows us to modify the 3DGS model according to the objects in the input image, and we expect it will also handle editing cases that require significant changes from the original 3DGS model.

3 Literature Review

A critical area of development in 3D editing emerged from the integration of generative models like Diffusion Models. Haque et al.’s Instruct-NeRF2NeRF [2] was one of the first to apply generative methods to 3D editing, leveraging Diffusion Models to manipulate NeRF representations. Instruct-NeRF2NeRF opened a pathway for further innovations that focused on the editing of 3D representations via multiple image viewpoints.

However, a key challenge in 3D image editing is maintaining consistency across different viewpoints, as Diffusion Models tend to introduce discrepancies when applied to multi-view image data.

Several strategies have been proposed to address these challenges. GaussCtrl [11], for example, relies on depth map extraction from 3DGS models to maintain consistency during editing. This technique improves coherence across viewpoints by referencing depth data, but at the cost of limiting the extent of potential modifications, as the edited model must still adhere to the original shape. Similarly, VCEdit [10] introduced mechanisms like the Cross-Attention Consistency Module and the Editing Consistency Module to ensure more uniform editing results across different perspectives. Despite these efforts, inconsistencies remain problematic when attempting more substantial changes to the 3D model.

In addition to consistency issues, another limitation is the focus on text-prompt-based editing, which restricts flexibility when attempting to manipulate specific objects within the scene based on visual data rather than textual descriptions. Methods like Custom-Edit [1] have made advancements in parameter tuning and smart retrieval of reference images to enhance control over 3D editing, but these remain reliant on textual input and fail to offer a solution for image-based object-specific manipulation.

The limitations of text-based editing and the trade-offs involved in depth-map-driven consistency methods point to the need for new approaches that can balance both flexibility and accuracy, particularly when significant changes to object structure are required.

A promising alternative is the application of Diffusion Personalization [8]. This approach, exemplified by methods like DreamBooth, allows for the fine-tuning of Diffusion Models by introducing new concept-token pairs that represent specific objects or styles. The primary advantage of this technique is the ability to incorporate new visual concepts into the generative process, which could enable more granular editing of 3D models based on image data rather than text prompts.

As we aim to advance this area, Diffusion Personalization presents an exciting avenue, offering the potential to train models specifically for object-based editing tasks, which could resolve many of the current challenges related to viewpoint consistency and model flexibility.

4 Baseline Selection and Model Description

To implement this project, we propose two baseline model ideas: Gaussian Control[4] and View-Consistent Edit[10]. These models aim to solve challenges in editing 3D models generated through NeRF[5] and 3D Gaussian Splatting (3DGS) techniques, by leveraging diffusion-based models and advanced control mechanisms.

Gaussian Control is a text-driven method for editing 3D Gaussian splats. It focuses on ensuring consistency in 3D edits using depth maps and attention-based latent code alignment. Depth maps are used to enhance the geometry of 3D models, ensuring that edits are applied consistently in terms of

spatial alignment. Meanwhile, attention-based modifications of latent codes provide more precise control when manipulating features like texture or shape and ensures the unified appearance of the splat during editing. The goal is to maintain coherence in the model’s overall structure while enabling fine-grained changes, especially when working with complex shapes and varying depth information.

View-Consistent Edit addresses the issue of multi-view consistency through cross-attention consistency and editing consistency mechanisms. A common challenge in 3D model editing is ensuring that changes made from one perspective propagate correctly to other views without introducing artifacts. VcEdit uses cross-attention modules to enforce consistent attention maps across views, ensuring that edits made to one view, such as texture changes or shape adjustments, are reflected coherently across all angles. Additionally, its editing consistency module helps in maintaining uniform transformations like color or texture changes across different areas of the model, avoiding distortions.

These two baseline models, Gaussian Control and View-Consistent Edit, offer unique approaches to balancing consistency, control, and flexibility in editing 3D models generated from NeRF and 3DGS techniques.

Out of these two ideas, we have selected GaussianCtrl as our baseline model on account of its higher efficiency and more streamlined approach. In particular, for our intended use case of making smaller edits with minimal geometric changes using a personalized diffusion model, as well as the limits of the computational resources we’ve been provided, GaussianCtrl is the easiest of the two models to experiment with. Furthermore, for 360 degree scenes GaussianCtrl shows particularly strong results both qualitatively and quantitatively, due to its depth-conditioned editing, and so is more appropriate for the personalization model.

Before we continue, we’ll briefly describe the main contributions of GaussianCtrl and how they’ll integrate into our approach.

4.1 Depth-conditioned editing

This process attempts to maintain geometric consistency when editing the 2D views rendered from the 3DGS. While rendering the 2D views, depth maps $\mathcal{D}$ are also rendered along with the images, and the paper uses the encoder portion of the ControlNet[12] VAE to get a latent code. Then, the latent code undergoes iterative DDIM inversion via Gaussian noise, the typical process for diffusion, and then the edited image is generated through denoising the noised latent with the edited prompt $\hat{p}$. For our approach, we’ll replace this denoising step with our Custom Edit personalized 2D diffusion model.

4.2 Attention-based latent code alignment

This is one of the key advancements of GaussCtrl, and its goal is similar to those in View-Consistent Edit. As we’re considering GaussCtrl, this is the one we’ll discuss. GaussCtrl’s latent code alignment module attempts to ensure consistency across the multiple edited 2D views by aligning their appearances through cross attention. First, the softmax cross attention matrix between all latent codes z_i and z_j , generated as mentioned above, is computed. Then, for an image that is about to be edited with latent code z_e^t and several reference views $z_{r,i}^t$, the alignment module computes both the self attention and cross view attention, $\text{AttnAlign}_e = \lambda \cdot \text{Attn}_{e,e} + (1 - \lambda) \cdot \frac{1}{N_r} \sum_{i=1}^{N_r} \text{Attn}_{e,i}$. The authors find that the self attention and cross attention helps align the appearance of all edited images to the reference views while maintaining the distinct nature of each edited image. In particular, they replace all image self attention modules with the proposed new alignment module in the U-Net of the diffusion model and the ControlNet block while denoising (see above). One of the key challenges will be fusing this attention alignment approach with our own personalized diffusion model.

4.3 Limitations

To compensate for limitations in GaussianCtrl, we propose the experimentation of personalization techniques as proposed by Custom-Edit. Custom-Edit takes an innovative approach by introducing smart prompting and reference image retrieval. This model allows users to guide the editing process by providing prompts or reference images, which are then integrated into the model using prompt-to-prompt editing with attention map injection. The ability to retrieve and use reference images ensures that the model adapts to specific details, allowing highly personalized and context-sensitive edits.

Custom-Edit is particularly suited for projects requiring precise, controlled edits that focus on very specific areas or features of the model.

5 Baseline Implementation Completeness

We develop a framework from GaussCtrl with the intent of salving the high-deviation edits that GaussCtrl suffers from, by leveraging the diffusion personalization techniques in Custom-Edit to maintain faithfulness to the original reference images.

Custom-Edit works as follows:

1. Fine tune a diffusion model from Custom-Diffusion.
2. Preprocess images with Null-Text Inversion. [6]
3. Edit images with the Prompt-to-Prompt framework [3], using the fine-tuned model from step 1.

5.1 Custom-Diffusion

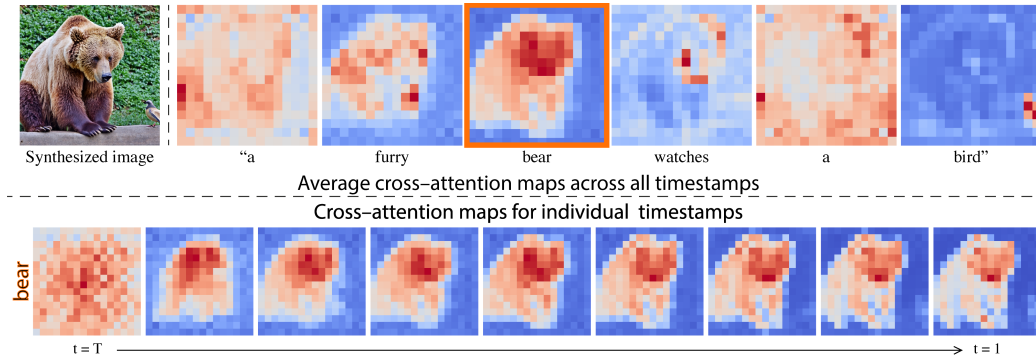
We have successfully reproduced Custom-Diffusion such that we have a full training workflow for fine-tuning diffusion models. Custom Diffusion is a Diffusion Personalization method designed to enable a pre-trained diffusion model to learn and associate a specific object concept with a unique text token. Through this targeted training process, the model becomes capable of generating images corresponding to prompts that include this specific token, thus facilitating the generation of particular objects on demand. Unlike DreamBooth, which was the first method to propose Diffusion Personalization, Custom Diffusion takes a more efficient approach by fine-tuning only the key and value projection matrices within the Cross Attention module. Additionally, it jointly optimizes the embedding of the text token itself, allowing for a streamlined and resource-efficient approach to diffusion personalization.

5.2 Prompt-to-Prompt

We also have Custom-Edit implemented on top of our Custom-Diffusion code. It takes the fine-tuned models from our Custom-Diffusion workflow and seats them in the driver’s seat of the Prompt-to-Prompt editing framework.

To understand the Prompt-to-Prompt image editing framework, we must first understand the cross-attention module in UNet, the denoising network of the diffusion model. A cross-attention map is a tensor of weights that indicate which pixels in the input image strongly correlate with a given token in the text prompt. The cross-attention module computes cross-attention maps using the attention mechanism, with the query being the input image and the key being the encoded text prompt.

Below is an illustration of the cross-attention maps. Notice how the token “bear” strongly correlates with the bear’s face, the token “bird” strongly correlates with the bird, and the token “furry” strongly correlates with the furry edges of the bear.



The idea behind Prompt-to-Prompt’s image editing mechanism is that we compute new cross-attention maps based on our input image and our modified text prompt.

To preserve integrity to the original image, we do not replace all original attention maps with our newly computed attention maps. Instead, we only replace the attention maps associated with the modified tokens of our prompt, making for a partial cross-attention replacement operation.

Three such partial cross-attention replacement operations are:

- **Replace:** Replacing selected tokens in the prompt, such as replacing teddy bear with kitty cat in “a smiling teddy bear” to produce “a smiling *kitty cat*”
- **Refine:** Extending the prompt with new words, such as “a smiling teddy bear *sitting under a tree*”
- **Reweight:** Enhancing or downplaying the significance of certain tokens by increasing or decreasing their respective weights in the cross-attention map, such as “a *smiling* (↑) teddy bear”

The cross-attention replacement operation is defined under the abstract interface `AttentionControlEdit`,

```
1 class AttentionControlEdit(AttentionStore, abc.ABC):
2     ...
3     @abc.abstractmethod
4     def replace_cross_attention(self, attn_base, att_replace):
5         raise NotImplementedError
6     ...
```

and individually implemented by each of the three respective classes for each type of replacement operation `Replace`, `Refine`, `Reweight`.

Replace

```
1 class AttentionReplace(AttentionControlEdit):
2
3     def replace_cross_attention(self, attn_base, att_replace):
4         return torch.einsum('hpw,bwn->bhpn', attn_base, self.mapper)
```

Refine

```
1 class AttentionRefine(AttentionControlEdit):
2
3     def replace_cross_attention(self, attn_base, att_replace):
4         attn_base_replace = attn_base[:, :, self.mapper].permute(2, 0, 1, 3)
5         attn_replace = attn_base_replace * self.alphas + att_replace * (1 - self.alphas)
6         return attn_replace
```

Reweight

```
1 class AttentionReweight(AttentionControlEdit):
2
3     def replace_cross_attention(self, attn_base, att_replace):
4         if self.prev_controller is not None:
5             attn_base = self.prev_controller.replace_cross_attention(attn_base, att_replace)
6             attn_replace = attn_base[None, :, :, :] * self.equalizer[:, None, None, :]
7         return attn_replace
```

5.3 Custom Edit

We implemented a custom edit for use in a 3D editing pipeline, leveraging a Personalized Diffusion Model to facilitate image editing. Unlike conventional text-based image editing methods, our approach allows us to incorporate specific objects from a reference image into a target image. For instance, we can replace a cat in an image with the likeness of a specific cat, such as a pet, by training a personalized diffusion model on photos of the pet.

The process for image editing using the Custom Edit method is as follows: (1) The reference image is trained on a specific token (e.g., <new1>) in a pretrained diffusion model using custom diffusion

method. (2) We obtain the initial latent representation and attention map of the target image to be edited through Null-text Inversion. (3) Utilizing the Prompt-to-Prompt method, we replace the attention map of the object in the target image with the attention map corresponding to the trained token. The following example demonstrates editing a cat in an image to match the cat from the reference image.



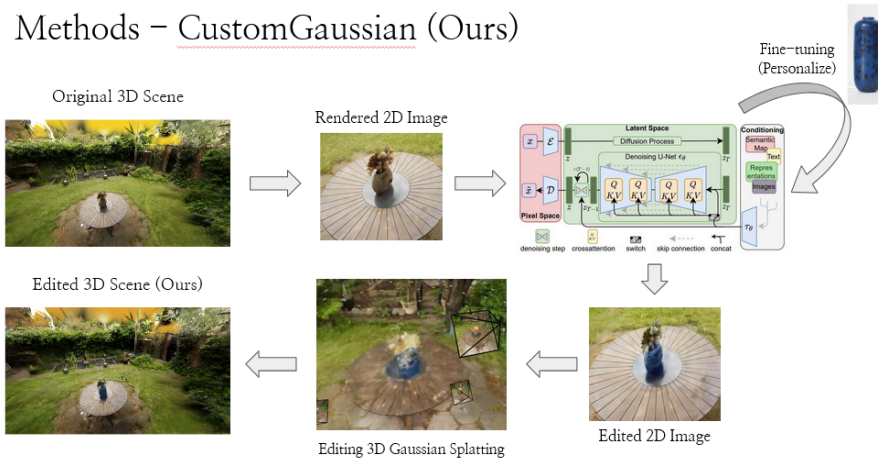
5.4 GaussCtrl

We have an implementation of GaussCtrl and have reproduced the results of the authors on the bear model, with the remaining work being to integrate it with our Custom-Edit implementation and resolve dependencies. We should note that in both CustomEdit and GaussCtrl, the quantitative measures such as $CLIP_{dir}$ do not fully capture the relative success of these methods because it focuses on the global similarity between the text prompt and edited images, over maintaining fine-grained detail and geometric similarity.

6 Model Description

We explore the viability of the diffusion personalization technique in reducing high-deviation edits in GaussCtrl. In this way, we retain GaussCtrl’s gift of multi-view consistency in the object’s 3D representation, while at the same time stabilizing the edits such that the 3D representation remains recognizable to its original form. Here is a summary of our framework:

Methods – CustomGaussian (Ours)



7 Results

We demonstrate SOTA results, superior qualitatively and quantitatively, to existing methods for 3DGS editing with a custom subject. Below is a qualitative comparison:

Results

Reference Image:



Original 3D Scene



Edited (Baseline)



Edited (Ours)



References

- [1] Jooyoung Choi, Yunje Choi, Yunji Kim, Junho Kim, and Sungroh Yoon. Custom-edit: Text-guided image editing with customized diffusion models, 2023.
- [2] Ayaan Haque, Matthew Tancik, Alexei A. Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions, 2023.
- [3] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control, 2022.
- [4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering, 2023.
- [5] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis, 2020.
- [6] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models, 2022.
- [7] Jangho Park, Gihyun Kwon, and Jong Chul Ye. Ed-nerf: Efficient text-guided editing of 3d scene with latent space nerf, 2024.
- [8] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation, 2023.
- [9] Dongqing Wang, Tong Zhang, Alaa Abboud, and Sabine Süssstrunk. Innerf360: Text-guided 3d-consistent object inpainting on 360-degree neural radiance fields, 2024.
- [10] Yuxuan Wang, Xuanyu Yi, Zike Wu, Na Zhao, Long Chen, and Hanwang Zhang. View-consistent 3d editing with gaussian splatting, 2024.
- [11] Jing Wu, Jia-Wang Bian, Xinghui Li, Guangrun Wang, Ian Reid, Philip Torr, and Victor Adrian Prisacariu. Gaussctrl: Multi-view consistent text-driven 3d gaussian splatting editing, 2024.
- [12] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023.