Text-Based 3D Editing: Optimizing Gaussian Splatting with CLIP

Aditya Kompella(apk74) * **Zichen Wang(zw336)** * **Akhilesh Sharma(as3774)** * **Daniel Cao(dyc33)** *

1. Introduction

In recent years, the field of computer graphics and computer vision has seen significant advancements in the long-established multi-view stereo problem, ranging from 3D reconstruction to novel view synthesis. Last year, Kerbl et al. (Kerbl et al., 2023) proposed Gaussian Splatting that achieves state-of-the-art performance. This opens up a range of opportunities for downstream applications, such as 3D style transfer explored by this project. In this project, our primary objective is to integrate the Contrastive Language-Image Pre-training (CLIP) (Radford et al., 2021) model into the Gaussian splatting process, presenting a novel approach for artistic 3D styles. This unique application represents a departure from existing works (Gao et al., 2023) in the field, especially as there is currently a notable absence of text-driven 3D style transfer within the context of Gaussian splatting. While numerous studies (Lin et al., 2023; Tang et al., 2023; Yu et al., 2023) have explored the realms of 3D Gaussian splatting, this project pioneers the incorporation of text prompts to achieve distinctive artistic styles, marking an advancement in the intersection of language understanding and 3D graphical synthesis. We propose two potential approaches to this problem

- Given a set of images and a text prompt, we optimize a Gaussian point cloud that achieves the artistic style specified by the text prompt;
- Given a pre-trained Gaussian point cloud and a text prompt, we fine-tune the Gaussian point cloud to achieve the artistic style specified by the text prompt

Prior works have explored various techniques for improving point cloud rendering and image synthesis. These efforts have encompassed traditional methods like voxelization, as well as more recent approaches that leverage deep learning and neural networks. However, several limitations remain:

Image-Text Integration The connection between textual descriptions and visual data has been a long-studied and yet still ongoing field of research. Methods like StackGAN (Zhang et al., 2017) have made strides in bridging the gap between textual descriptions and 3D content, but further improvements are necessary for effective image-text integration such as better generalization to image-text rendering of photo-realistic images under arbitrary environments.

Regularization Given that Gaussian Splatting came out very recently, few works have systematically studied adding regularization during the training process. Most works (Mildenhall et al., 2020) in multi-view stereo focus on optimizing the 3D parameters so that the rendered images match the reference images. In this project, we identify the possibility of incorporating CLIP to enhance the overall training process.

This project overcomes these limitations by fusing the text-driven directives of CLIP with the advanced 3d rendering techniques of Gaussian Splatting, propelling the field of artistic 3d styles into unexplored categories. Please visit our Github repository for the code: `https://github.com/zichenwang01/gaussian-splatting`.

2. Related Works

Structure-from-Motion (SfM) The advent of SfM (Nyimbili et al., 2016) represents one of the first classic approaches to the long-established multi-view stereo (MVS) problem. SfM takes a batch of images as input and jointly estimates the camera parameters and a parse point cloud. It is common for recent MVS models to run SfM first to obtain the camera parameters, which is what Gaussian Splatting did.

Gaussian Splatting Last year, Kerbel et. al. proposed Gaussian splatting (Kerbl et al., 2023) that achieves state-of-art performance in terms of training speed, reconstruction quality, and real-time rendering. Given a set of images, Gaussian Splatting optimizes a Gaussian point cloud through minimizing the loss between the reference images and the rasterized images. Compared with NeRF (**?**) methods, the novel Gaussian point cloud representation addresses the sparsity issue of neural fields and enables faster rendering algorithms.

3D Style Transfer 3D style transfer aims to transform the appearance of a 3D scene so that its renderings from different viewpoints match the style of a desired image. Previous approaches represent real-world scenes using point clouds (Huang et al., 2021; Mu et al., 2022), triangle meshes (Yin et al., 2021; Michel et al., 2021), or neural radiance fields (Zhang et al., 2022). In this project, we explore the possibility of using the current state-of-the-art Gaussian point clouds in 3D style transfer.

Figure 1. Illustration of our artistic Gaussians. On the left is one of the original input images; on the right is a rendered image of the Gaussians fine-tuned with text prompt "rococo painting of a garden."A rococo painting of a garden with abstract elements."

3. Methods

3.1. Joint Optimization

Gaussian Splatting first runs SfM to obtain the camera parameters and a sparse point cloud. Based on this initial point cloud, the model repeatedly does the following

- Given a point cloud Θ, rasterizes the point cloud to obtain images $\hat{I} = R(\Theta)$. Here R denotes the rasterization process as a function.
- Given the rasterized image $\hat{I}$, compute the loss, often L1 loss, between the rasterized image and the reference image $\mathcal{L}(\hat{I}, I)$. Here I denotes the reference image.
- Given the loss $\mathcal{L}$, back-propagate the loss to update the point cloud Θ.

In other words, Gaussian Splatting optimizes the following objective

$$\Theta^* = \arg\min_{\Theta} \mathcal{L}(R(\Theta), I) \tag{1}$$

Noticeably, the gradient back-propagation uses automatic differentiation to compute

$$\frac{\partial \mathcal{L}}{\partial \Theta} = \frac{\partial \mathcal{L}}{\partial \hat{I}} \cdot \frac{\partial R(\Theta)}{\partial \Theta} \tag{2}$$

This requires the loss function to be differentiable with respect to the images and the rasterization process to be differentiable with respect to the point cloud. We shall keep this in mind when modifying the loss function below.

To incorporate additional information into the optimization process, we propose to take the rasterization image as input to the CLIP model. Given a text prompt T specifying the artistic style, the CLIP model outputs a CLIP score CLIP$(\hat{I}, T)$ that tells us how well the text prompt describes the image. We use this CLIP score as the regularization term during optimization. This gives us the following loss function

$$\mathcal{L}(\hat{I}, I) = \mathcal{L}_1(\hat{I}, I) - k\text{CLIP}(\hat{I}, T) \tag{3}$$

Here k is a coefficient scaling the weight of the CLIP regularization.

Optimizing this loss function, then, jointly optimizes the Gaussian point cloud to align with the reference image and the spherical harmonics to align with the text prompt. This enables us to faithfully reconstruct the geometry as well as artistic appearance.

3.2. Fine-Tuning

Apart from adding a regularization term and training from scratch, we can also take a pre-trained Gaussian point cloud and fine-tune its spherical harmonics to achieve a style-transferred artistic appearance. Compared with jointly optimizing geometry and appearance, this method optimizes only the appearance and it is thus expected to be faster. In addition, fine-tuning is advantageous when we want to style-transfer between multiple artistic styles. On the flip side, fine-tuning means that we need to have a pre-trained Gaussian point cloud to start with. Thus, both these two methods have their merits and the choice between these two methods depends on the specific tasks.

Fine-tuning a pre-trained Gaussian point cloud for style transfer involves adjusting its spherical harmonics to align with a specified artistic style. This process is distinct from training from scratch, as it focuses solely on modifying the appearance attributes of the point cloud, rather than its underlying geometry. Mathematically, this can be represented as follows:

Let $\Theta = \{\theta_1, \theta_2, ..., \theta_n\}$ be the set of parameters of our pre-trained Gaussian point cloud, where each θ_i represents the parameters (like position, color, and density) of each

Gaussian element in the cloud. The spherical harmonics associated with these parameters, denoted as $H(\Theta)$, determine the appearance of the rendered image.

During fine-tuning, we aim to adjust $H(\Theta)$ to match a desired artistic style, specified by a text prompt T. This is achieved by optimizing the following objective function:

$$\Theta^* = \arg\min_{\Theta} \mathcal{L}_{style}(H(\Theta), T) \tag{4}$$

Here, $\mathcal{L}_{style}$ is a loss function that measures the discrepancy between the current style of the point cloud and the style described by the text prompt T. This function can be formulated using the CLIP model's ability to evaluate the similarity between text and images:

$$\mathcal{L}_{style}(H(\Theta), T) = -\text{CLIP}(R(H(\Theta)), T) \tag{5}$$

In this equation, R represents the rendering function that converts the point cloud with its current spherical harmonics into an image. The CLIP score then evaluates how well this rendered image aligns with the text prompt T.

Fine-tuning is particularly advantageous when dealing with multiple style transfers, as it allows for rapid adjustments to the appearance without the need to retrain the entire model. However, it requires a well-trained base Gaussian point cloud as a starting point. The choice between fine-tuning and training from scratch depends on the specific requirements and constraints of the task at hand.

3.3. Differentiability

Given the new loss function, we shall carefully check that it is differentiable with respect to the point cloud.

$$\frac{\partial \mathcal{L}}{\partial \Theta} = \frac{\partial \mathcal{L}_1}{\partial \Theta} + \frac{\partial}{\partial \Theta}\text{CLIP}(R(\Theta), T) \tag{6}$$

Gaussian Splatting guarantees that $\frac{\partial \mathcal{L}_1}{\partial \Theta}$ is well-defined. This term typically involves standard operations in neural networks and is well-defined under the framework of automatic differentiation. The differentiability of this term is crucial for applying gradient-based optimization methods effectively.

The second term, involving the CLIP model, requires a more nuanced analysis. To compute the CLIP score, CLIP passes the text and the image into a $Text\_Encoder$ and an $Image\_Encoder$. This step outputs two vectors $u = Text\_Encoder(T)$ and $v = Image\_Encoder(\hat{I})$ that represent the embedding of the text/image. Finally, the CLIP score is the dot product $u \cdot v$. CLIP guarantees that the encoders are differentiable with respect to their input, and the dot product is also a smooth function.

However, the differentiability of the CLIP term hinges on the differentiability of the $Image\_Encoder$ with respect to its input image $\hat{I}$. Assuming that the $Image\_Encoder$ is a differentiable function, as is typical in deep learning models, the gradient $\frac{\partial}{\partial \hat{I}}\text{CLIP}(\hat{I}, T)$ is well-defined. Additionally, the rasterization process $R(\Theta)$, which converts the point cloud to an image, also needs to be differentiable with respect to the point cloud parameters Θ. This is a key requirement for Gaussian Splatting, where the rasterization process involves rendering the point cloud into an image.

Therefore, the overall differentiability of the loss function $\mathcal{L}$ depends on the differentiability of both reconstruction loss and the CLIP regularization term. By ensuring that all components of the loss function are differentiable with respect to the point cloud parameters, we can effectively apply gradient-based optimization techniques to train the model.

The differentiability of our new loss function is supported by the well-defined gradients of both the reconstruction loss and the CLIP regularization term, enabling effective gradient back-propagation for optimizing the Gaussian point cloud in alignment with both the reference image and the artistic style specified by the text prompt.

4. Results

To evaluate our model qualitatively we trained a Gaussian Splat on the common NeRF Train Data set and fine-tuned the Gaussians using various prompts. We also compare our model with the previous CLIP3Dstyler (Gao et al., 2023) on the Trunk scene in the Trunk and Temples data set.

4.1. Joint Optimization vs. Fine-Tuning

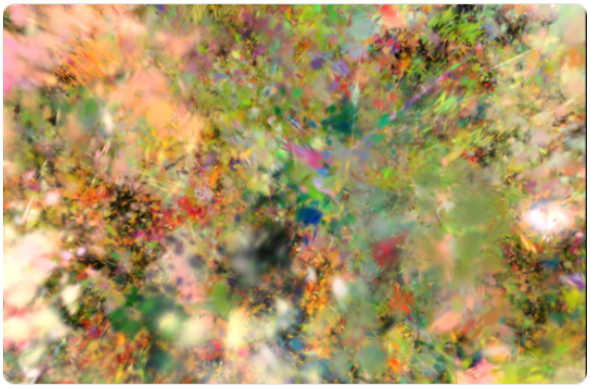

Figure 2. Illustration of the joint optimization scheme. Jointly optimizing both objectives does not lead to results that have good structure."

Our initial comparison reveals a significant disparity between the joint optimization and fine-tuning approaches—

joint optimization proves to be less effective. When the CLIP loss is too small, the Gaussians remain unchanged from the original state. In other words, attempting to minimize the CLIP loss to capture specific artistic effects results in substantial image loss. The model tends to ignore the CLIP loss in favor of preserving the image content. Conversely, when the CLIP loss weight is too large, the Gaussian rapidly transforms into a random pattern (Figure 2). In such cases, the model prioritizes optimizing the CLIP loss, neglecting the image loss. Despite the attempts to fine-tune the weight parameter, optimization stays quite sensitive to the weight, so conclude that fine-tuning is the more suitable option for this task.

4.2. Artistic Effects of Fine-Tuning

To show the artistic effects of our model, we fine-tuned the Train scene under various text prompts (Figure 3). In each example, the overall semantics of the original image are well-preserved. The train remains distinctly recognizable, and finer elements, such as handrails and roadblocks, are also retained with impressive detail. For instance, the moon head in the "day of the dead" example, harmonizes well with the image, conveying a sense of impending doom and eerie nights without overpowering the original composition. Other examples include the starfish in the undersea scene and the plants at the head of the train in the rococo example. Furthermore, the model demonstrates a commendable optimization of places of **low frequency**. For example, the sky in the original image is largely made up of the same kind of blue, but in the "day of the dead" example, it turns into dark blue to convey to sense of dark nights. Similarly in the undersea example, the model introduces more variation to the original sky, effectively portraying the vibrant and colorful underwater world.

The model also maintains the **high frequency** details exceptionally well. In Figure 4, we highlight the effective preservation of the letters on the side of the train. Although the color of the letters has been adjusted to better align with the desired art style, the specific number of letters remains the same. Like many other generative models, the preservation of the letters is not guaranteed in all situations, it is worth noting our model exhibits superior performance in this aspect compared to most generative models.

In conclusion, freezing opacity is a design choice and yields images more faithful to the original scene, preserving details such as the letters on the train. Additionally, freezing the opacity results in fewer "spikes" in the image as the optimized opacity allows Gaussians to blend seamlessly in the original scene. Conversely, unfreezing the opacity provides the model with greater creative freedom, introducing more intricate details to the Gaussians, particularly visible at a distance.

4.3. Comparison with Previous Methods

A key distinction between 2D image style transfer and 3D model style transfer is the ability to effortlessly render from novel views. This necessitates **geometric consistency** in the novel views and, consequently, the assurance that the new 3D model maintains the accurate representation of the overall structure. In this context, we compare our approach with previous methods to highlight the advantage of using Gaussian point clouds as a better shape representation. The method we compare with, CLIP3DStyler employs a more intricate network architecture. However, its fundamental approach aligns with ours, utilizing CLIP to train a Neural Radiance Field (NeRF), making it a suitable benchmark for evaluating our results. In all three instances, our results exhibit a clear differentiation between the foreground trunk and the background trees. Conversely, CLIP3Dstyler, in all three cases, tends to blur the trunk head, making it hard to distinguish it from the background. Moreover, in two out of three cases (the purple brush and the watercolor painting), one still retains the existence of a tree in the background. Only in cases where trees significantly deviate from the text prompts (the pop art of the night city) does our model transform the tree into a different element. This is in contrast to the CLIP3Dstyler, where the presence of the tree becomes indiscernible in all three instances. Furthermore, it is noteworthy that our model not only exhibits more artistic details but also demonstrates superior performance in capturing the overall art style, as seen in the purple brush case where our trunk exhibits more brush-like textures.

5. Conclusion

Our project introduces a novel approach to artistic 3D style transfer by seamlessly integrating the CLIP model into the Gaussian splatting process. We redefine the landscape by introducing a hybrid loss function that combines reconstruction and CLIP losses, ensuring the generated Gaussian splats excel not only in geometric accuracy but also in capturing rich visual details guided by textual prompts. Our loss function and its constituent CLIP term and reconstruction loss also have mathematical guarantees with respect to the point cloud allowing for efficient backpropagation for optimizing the alignment capabilities of our model. By overcoming the limitations of previous works and harnessing CLIP's robust image-text alignment capabilities, we see substantial enhancements in the quality, precision, and expressive richness of 3D data rendering and synthesis. This research promises widespread applications across computer graphics and computer vision, from 3D reconstruction to image synthesis and styling, pushing the boundaries of the field and paving the way for dynamic and interactive 3D environments online.

References

Gao, M., Xu, Y., Zhao, Y., Hou, T., Zhao, C., and Gong, M. Clip3dstyler: Language guided 3d arbitrary neural style transfer, 2023.

Huang, H.-P., Tseng, H.-Y., Saini, S., Singh, M., and Yang, M.-H. Learning to stylize novel views. *arXiv preprint arXiv:2105.13509*, 2021.

Kerbl, B., Kopanas, G., Leimkühler, T., and Drettakis, G. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

Lin, Y., Dai, Z., Zhu, S., and Yao, Y. Gaussian-flow: 4d reconstruction with dynamic 3d gaussian particle. *arXiv:2312.03431*, 2023.

Michel, O., Bar-On, R., Liu, R., Benaim, S., and Hanocka, R. Text2mesh: Text-driven neural stylization for meshes. *arXiv preprint arXiv:2112.03221*, 2021.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

Mu, F., Wang, J., Wu, Y., and Li, Y. 3d photo stylization: Learning to generate stylized novel views from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16273–16282, 2022.

Nyimbili, P. H., Demirel, H., Seker, D., and Erden, T. Structure from motion (sfm)-approaches and applications. In *Proceedings of the international scientific conference on applied sciences, Antalya, Turkey*, pp. 27–30, 2016.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision, 2021.

Tang, J., Ren, J., Zhou, H., Liu, Z., and Zeng, G. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*, 2023.

Yin, K., Gao, J., Shugrina, M., Khamis, S., and Fidler, S. 3dstylenet: Creating 3d shapes with geometric and texture style variations. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2021.

Yu, H., Julin, J., Milacski, Z. , Niinuma, K., and Jeni, L. A. Cogs: Controllable gaussian splatting, 2023.

Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., and Metaxas, D. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017.

Zhang, K., Kolkin, N., Bi, S., Luan, F., Xu, Z., Shechtman, E., and Snavely, N. Arf: Artistic radiance fields, 2022.

(a) Original 3D Gaussian for Train Scene

(b) Prompt: "A rococo painting of a garden with abstract elements and high resolution"

(c) Prompt: "day of the dead, in the style of unreal engine 5, caricature-like illustrations, 32k uhd, dark atmosphere"

(d) Prompt: "Under the sea background, Marine Life Landscape, Cartoon style, Disney"

Figure 3. Experiments Finetuning Gaussians

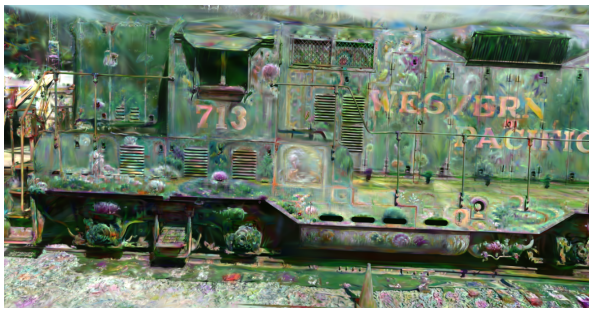

Figure 4. The Text on the side of the train stays well preserved when not fine-tuning the opacity

Figure 5. The Text on the side of the train becomes slightly warped when finetuning opacity of the Gaussian but more of the 3D structure is modified

(a) Our Work

(b) CLIP3Dstyler

Figure 6. Prompt: ”Pop art of night city”

(a) Our Work

(b) CLIP3Dstyler

Figure 7. Prompt: ”A watercolor painting with purple brush”

(a) Our Work

(b) CLIP3Dstyler

Figure 8. Prompt: ”Watercolor painting”