CLIP-based Neural Neighbor Style Transfer for 3D Assets

Shailesh Mishra[†1,2] and Jonathan Granskog[†3]

[1]German Research Center for Artificial Intelligence (DFKI), [2]NVIDIA, [3]Runway

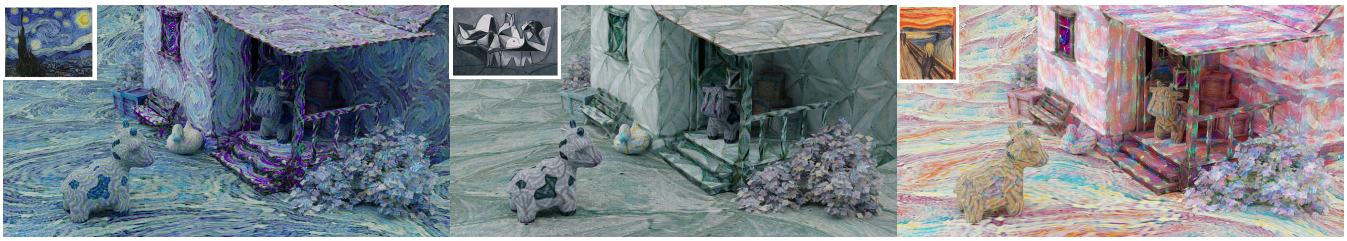

Figure 1: *Our method stylizes textures of objects such that they can be rendered easily with traditional 3D renderers. We show renders of a scene, created in Blender, containing objects that were independently processed by our method using various style images (top left corners).*

Abstract

We present a method for transferring the style from a set of images to the texture of a 3D object. The texture of an asset is optimized with a differentiable renderer and losses using pretrained deep neural networks. More specifically, we utilize a nearest-neighbor feature matching (NNFM) loss with CLIP-ResNet50 that we extend to support multiple style images. We improve color accuracy and artistic control with an extra loss on user-provided or automatically extracted color palettes. Finally, we show that a CLIP-based NNFM loss provides a different appearance over a VGG-based one by focusing more on textural details over geometric shapes. However, we note that user preference is still subjective.

CCS Concepts

• ***Computing methodologies*** → *Appearance and texture representations; Rasterization; Supervised learning by regression;*

1. Introduction

Artistic style becomes an increasingly important factor of differentiation from other digital content as the availability of physically-based rendering improves. Modern animated shows often leverage manual stylization to produce memorable visual qualities (see Spiderverse [Son18] or Arcane [RN21]). As stylized content becomes more common, manual processes might become too tedious especially for large asset libraries. Automatic methods can prove useful for either fully stylizing objects or providing great starting points. Our research presents an initial attempt to tackle automatic stylization of 3D assets using CLIP [RKH*21] based on a set of images (Figure 1). We motivate the study of CLIP for 3D asset stylization by its success in generative applications [RDN*22] and hypothesize that the features of VGG are less optimal for stylization due to classification being its target application. Our results show promise but are not clearly better than VGG yet.

† Work done while both were employed at NVIDIA

2. Related work

Neural style transfer applies the style of an input image to other content by minimizing the distance between statistics of feature maps, such as Gram matrices, in a pretrained convolutional neural network [GEB16]. Many recent works have improved on this framework; see [JYF*20] for a comprehensive review. CLIP [RKH*21] has proven powerful in both stylizing and generating images, meshes and neural fields using text input [JMB*22, KY21, MBOL*21, RDN*22]. Due to its ability to represent style, we investigate the use of CLIP for neural stylization of 3D assets. Similar to [KUH18], we optimize textures of 3D assets with differentiable rendering but use features from a pretrained CLIP-ResNet50 network instead of VGG16 [LD15]. We combine CLIP with a nearest-neighbor feature matching (NNFM) approach [KKP*22], as Gram matrices function poorly, and extend it to multiple style images and include control over colors. Albeit with VGG, [ZKB*22] show that NNFM can successfully stylize neural fields. Our results are not provably better than VGG-based NNFM, but we hope our contributions are still useful for others.

DOI: 10.2312/egs.20231006

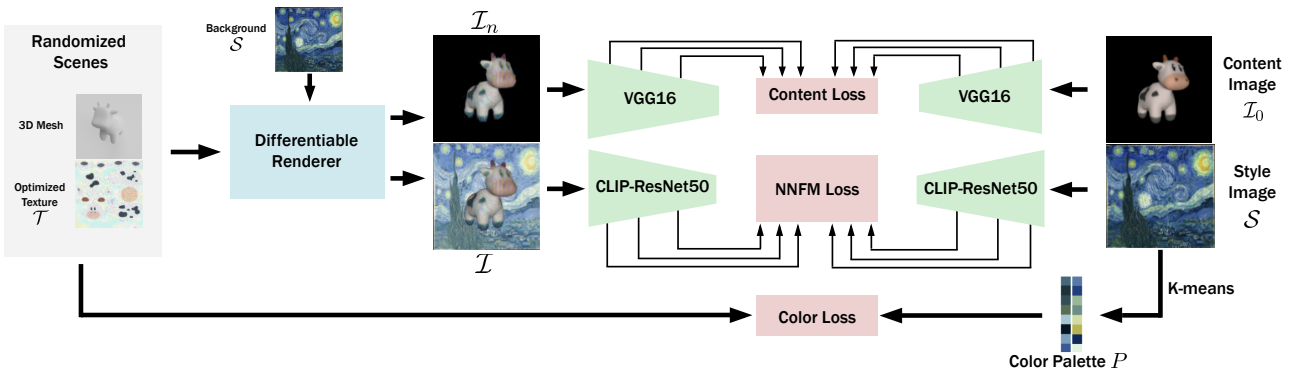

Figure 2: *We stylize textures by minimizing a nearest-neighbor feature matching loss with CLIP-ResNet50, a VGG-based content loss and a color palette loss. The NNFM style loss looks at rendered images with the style image as background whereas the content loss is computed on renders without background. The color palette loss ensures colors are accurate to the style image. The palette is derived from cluster centers predicted by K-means.*

3. Method

In this section, we describe our method for transferring the style from a set of images to the texture $\mathcal{T}$ of a 3D asset. We present three main contributions; a CLIP-ResNet50-based style loss using nearest-neighbor feature matching (NNFM), an extension of NNFM to multiple style images, and a color palette loss to improve color accuracy and artistic control. To match the style of an input image $\mathcal{S}$, we use a nearest-neighbor feature matching (NNFM) loss that minimizes distances in feature space while ignoring spatial locations [KKP*22]. We rely on the differentiable renderer from Laine et al. [LHK*20] due to its simplicity, but buffers output by a deferred renderer could also be used. Our loss consists of three components

$$\mathcal{L} = \lambda_S \mathcal{L}_S + \lambda_C \mathcal{L}_C + \lambda_P \mathcal{L}_P \tag{1}$$

where $\mathcal{L}_S$ is the style loss, $\mathcal{L}_C$ a VGG-based content loss and $\mathcal{L}_P$ our color palette loss. We describe these in the next sections. We use batched optimization, and for each batch element we randomize the camera and lighting to optimize over a diverse set of rendered images rather than a restricted setting. See Figure 2 for an overview of our approach and the supplementary material for more details.

CLIP-based NNFM style loss Here, we describe our application of the NNFM style loss [KKP*22]. Given a rendered image $\mathcal{I}$ and style image $\mathcal{S}$, we forward propagate both images through CLIP-ResNet50 and extract feature maps from a set of layers L. Then, for each layer $l \in L$, its feature maps are reshaped into sets of N and M feature vectors $F^l(\mathcal{S})$ and $F^l(\mathcal{I})$ respectively. We minimize the distance of each render feature vector to the nearest style feature vector in the same layer according to

$$\mathcal{L}_S = \sum_{l \in L} \frac{1}{M} \sum_{i}^{M} \min_{j}^{N} \mathcal{D}(F^l(\mathcal{I})_i, F^l(\mathcal{S})_j) \tag{2}$$

where $\mathcal{D}$ is the cosine similarity between the feature vectors and L consists of the second convolutions from each block in layer3 and layer4. We also follow Wang et al. [WLV21] and smooth the features with a Softmax before computing the style loss. We set the style image as the background to reduce the impact of a mismatched background on optimization. To account for multiple style images $\{\mathcal{S}_k\}_{k=1...K}$, we combine the reshaped sets of style feature vectors for layer l into a single larger set $F^l(\{\mathcal{S}_k\}_{k=1...K}) = \{F^l(\mathcal{S}_1), F^l(\mathcal{S}_2), ..., F^l(\mathcal{S}_K)\}$ and find the nearest feature vector from that set instead.

Content loss To retain the characteristic content of the original texture, we utilize a standard VGG16-based content loss which minimizes the L2 distance between feature maps of a content image $\mathcal{I}_0$, the 3d object without any stylization, and the current rendered $\mathcal{I}_n$ image: $\mathcal{L}_C = \sum_{v \in V} L_2(F^v_{vgg}(\mathcal{I}_0), F^v_{vgg}(\mathcal{I}_n))$. The set of layers V are layers 11, 13 and 15 from VGG16. Note that we do not add the style image as the background of $\mathcal{I}_0$ and $\mathcal{I}_n$ (Figure 2).

Color palette loss CLIP struggles to match colors in the style image accurately without an additional color loss term to direct the texture stylization (Figure 4). We include a loss that matches texture colors to a color palette P, either automatically extracted from the style image or user-provided for artistic control. For each pixel in the texture $\mathcal{T}$, we minimize the L2 distance to the nearest RGB color in the palette

$$\mathcal{L}_P = \frac{1}{WH} \sum_{i,j}^{W,H} \min_{k}^{|P|} L_2(\mathcal{T}_{i,j}, P_k) \tag{3}$$

We apply the loss to the texture $\mathcal{T}$, rather than the output render $\mathcal{I}$, to ignore the effects of shading and background. To automatically extract the color palette from an image, we cluster colors based on K-means and the final cluster centers become the palette. With multiple style images, we select a primary style image for the color palette extraction, but other methods would also work well.

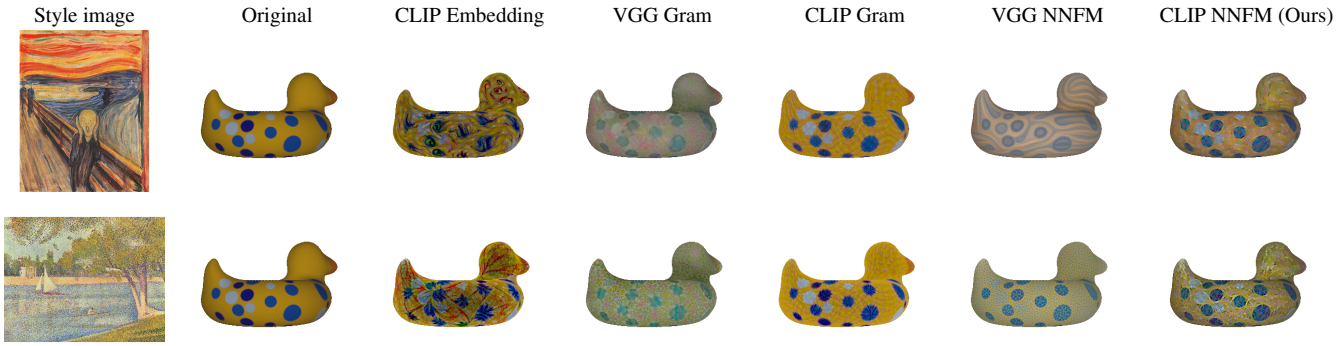

Figure 3: *Comparison of different style losses without our color palette loss. Directly using the CLIP embedding also extracts content from the style image, in this case eyes and leaves, and prevents functioning transfer of style. Both Gram matrix-based methods show poor transfer of style whereas the NNFM-based methods work better. However, VGG and CLIP extract different appearances from the style image. CLIP focuses more on texture whereas VGG captures lines and points.*

4. Results

In this section, we present results from our style transfer method for textures of 3D assets. We compare each style transfer approach and demonstrate the impact of each loss term. Then, we show artistic control of colors and the effect of multiple style images. Additional comparisons can be found in the supplemental material.

Style transfer comparison In Figure 3, we show a comparison of various style transfer methods. A direct loss on the CLIP embedding matches both content and style. Therefore, eyes and leaves are visible in the textures. The standard Gram-matrix-based approach with both VGG16 [GEB16] and CLIP-ResNet50 [RKH*21] is unable to properly learn a global style appearance. The NNFM with VGG16 [KKP*22] manages to capture brush strokes and geometric shapes well whereas CLIP-ResNet50 NNFM learns a more textural appearance. In the top row, VGG NNFM captures long brush strokes flowing along the mesh but lacks texture. CLIP on the other hand produces a painterly appearance but is missing longer strokes. See also Figure 5. We hypothesize that the difference is due to the different training objectives; classification for VGG and text-image pairing for CLIP. We follow Zhang et al. [ZKB*22] and use feature maps from layers 11, 13 and 15 for VGG16-based style losses. The supplemental material also provides a comparison of CLIP scores.

Loss ablation Our style transfer method consists of three losses; a NNFM style loss, a VGG16-based content loss and a color palette loss (section 3). Figure 4 shows the impact of each loss on our style transfer method. The content loss constrains the optimization such that we do not lose characteristic features from the original texture, such as eyes of the cow. The color palette loss helps match the colors of the style image better. Figure 5 shows that artistic control over colors is possible by providing a custom palette instead of the one extracted automatically by K-means.

Impact of multiple style images The nearest-neighbor feature matching loss extends naturally to a larger number of style images simply through concatenation as shown in section 3. In Figure 6, we show the impact that additional images have on NNFM-based style transfer. As additional style images are included, the style transfer appearance changes and new patterns and colors can appear. In these results, we used the first style image as the background and for color palette extraction.

Figure 4: *The impact of each loss in CLIP-based NNFM style transfer when using a black and white line art drawing as the style. The color palette loss ensures colors match the style image whereas the content loss ensures the content is preserved.*

Conclusion and Future Work

Our method is an initial attempt at stylization of 3D assets with CLIP. The nearest-neighbor feature matching loss produces a better style compared to CLIP with Gram matrices. A CLIP-ResNet50-based NNFM loss extracts a more textural appearance than NNFM with VGG16, which focuses on geometric patterns. NNFM with multiple style images enables generating textures with the combined appearance of multiple styles. Our color palette loss improves the color accuracy of the stylized texture and provides additional artistic control. However, our results are not objectively better than VGG16 NNFM. Results would likely improve if a stylization method leverages the vision transformer architectures of CLIP instead. Neural style fields [MBOL*21] could also be beneficial, especially for scene-level stylization, as texture coordinates might not be optimal (Figure 1, 6). Stylizing BRDFs, e.g. for toon shading, also presents an interesting opportunity for novel research.

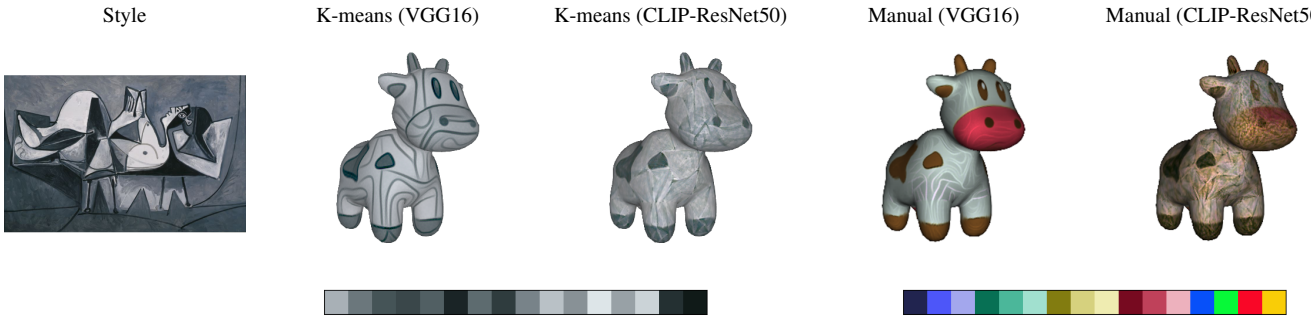

Figure 5: *Our color palette loss allows controlling the color of the end result, even if the selected palette is very different from the colors found in the style image. Columns labeled manual use a manually-selected color palette and the other columns use K-means extraction. One can also see that CLIP-ResNet50 focuses more on texture whereas VGG16 prefers synthesizing geometric features such as lines.*

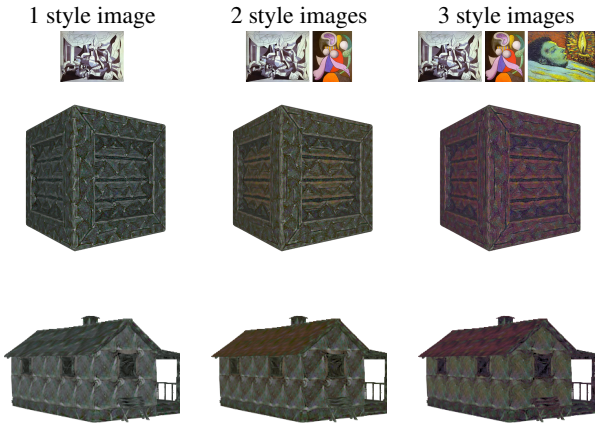

Figure 6: *Additional style images affect the optimization result as there are a larger number of feature vectors to compare against in the NNFM loss. The repetition artifacts on the house are due to the texture coordinates of the asset.*

Acknowledgements

We thank Philipp Fischer, Guilin Liu, Jacob Munkberg, Timo Roman, Towaki Takikawa and Kangxue Yin for insightful discussions, guidance and feedback. We thank the creators of the following assets; Spot and Bob from Keenan Crane, the hibiscus bush from Jacob Munkberg, the wooden box from Turbosquid user JeanSouza, the cottage model from user nukemut and the elephant drawing from publicdomainpictures.net.

References

[GEB16] GATYS L. A., ECKER A. S., BETHGE M.: Image style transfer using convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 2414–2423. doi:10.1109/CVPR.2016.265. 1, 3

[JMB*22] JAIN A., MILDENHALL B., BARRON J. T., ABBEEL P., POOLE B.: Zero-shot text-guided object generation with dream fields. 1

[JYF*20] JING Y., YANG Y., FENG Z., YE J., YU Y., SONG M.: Neural style transfer: A review. *IEEE Transactions on Visualization and Computer Graphics 26*, 11 (2020), 3365–3385. doi:10.1109/TVCG.2019.2921336. 1

[KKP*22] KOLKIN N., KUCERA M., PARIS S., SYKORA D., SHECHTMAN E., SHAKHNAROVICH G.: Neural neighbor style transfer, 2022. URL: https://arxiv.org/abs/2203.13215, doi:10.48550/ARXIV.2203.13215. 1, 2, 3

[KUH18] KATO H., USHIKU Y., HARADA T.: Neural 3d mesh renderer. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 3907–3916. 1

[KY21] KWON G., YE J. C.: Clipstyler: Image style transfer with a single text condition. *arXiv preprint arXiv:2112.00374* (2021). 1

[LD15] LIU S., DENG W.: Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)* (2015), pp. 730–734. doi:10.1109/ACPR.2015.7486599. 1

[LHK*20] LAINE S., HELLSTEN J., KARRAS T., SEOL Y., LEHTINEN J., AILA T.: Modular primitives for high-performance differentiable rendering. *ACM Trans. Graph. 39*, 6 (nov 2020). URL: https://doi.org/10.1145/3414685.3417861, doi:10.1145/3414685.3417861. 2

[MBOL*21] MICHEL O., BAR-ON R., LIU R., BENAIM S., HANOCKA R.: Text2mesh: Text-driven neural stylization for meshes. *arXiv preprint arXiv:2112.03221* (2021). 1, 3

[RDN*22] RAMESH A., DHARIWAL P., NICHOL A., CHU C., CHEN M.: Hierarchical text-conditional image generation with clip latents, 2022. URL: https://arxiv.org/abs/2204.06125, doi:10.48550/ARXIV.2204.06125. 1

[RKH*21] RADFORD A., KIM J. W., HALLACY C., RAMESH A., GOH G., AGARWAL S., SASTRY G., ASKELL A., MISHKIN P., CLARK J., KRUEGER G., SUTSKEVER I.: Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning* (18–24 Jul 2021), Meila M., Zhang T., (Eds.), vol. 139 of *Proceedings of Machine Learning Research*, PMLR, pp. 8748–8763. URL: https://proceedings.mlr.press/v139/radford21a.html. 1, 3

[RN21] RIOT GAMES, NETFLIX: Arcane: League of Legends, 2021. 1

[Son18] SONY PICTURES: Spider-Man: Into the Spider-Verse, 2018. 1

[WLV21] WANG P., LI Y., VASCONCELOS N.: Rethinking and improving the robustness of image style transfer. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2021). 2

[ZKB*22] ZHANG K., KOLKIN N., BI S., LUAN F., XU Z., SHECHTMAN E., SNAVELY N.: ARF: Artistic radiance fields, 2022. URL: https://arxiv.org/abs/2206.06360, doi:10.48550/ARXIV.2206.06360. 1, 3