Joint Training for Style Transfer in NeRF

Felix O'Mahony
Princeton University
New Jersey, NJ 08544
fo8087@princeton.edu

Abstract

We present a method which permits the adaption of th style of a Neural Radiance Field (NeRF) 3D model. For the adaption of the style of one NeRF model, a pair of additional models are trained with one approximating the source style and the other approximating the target style. Our results demonstrate good performance across several applications.

1 Introduction

The use of NeRF as a method for generating novel views of a three-dimensional scene has shown remarkable performance. The method is able to generate very high quality new views of a scene from a relatively low number of original views Mildenhall et al. (2020). However, while the method has shown an ability to generate new views of a scene, the generated scenes show low malleability. It is difficult to transform their appearance since this is coded into the weights and biases of a multi-layer perceptron network.

We propose a method which permits the modification of the style of a three-dimensional scene generated with NeRF. Whereas existing methods for style transfer typically rely on transforming the style of a NeRF model's training image set to adapt styles, we propose a method which modifies the model directly. In our method, a reference scene is generated in two styles. In one, the style resembles that of the target scene. Another serves as a desired style, which resembles the desired appearance of the target scene. By training two networks in parallel, we are able to transfer the style from the reference model in the desired style to the target scene.

We test our model on a range of scenarios including changing colours, patterns and lighting. While the method demonstrates strong performance at transferring colours and lighting conditions, it exhibits a lower degree of success at transferring patterns to a NeRF model.

2 Related Work

2.1 3D Scene Representation

Neural Radiance Fields (NeRF), from Mildenhall et al. (2020) aims to represent a 3D scene as an overfit multi-layer perceptron which returns the colour radiated and volumetric density of a point in five-dimensional space. A single network is tuned to a specific scene. The scene is rendered from the network by sampling along camera rays and taking the cumulative emitted colour value at each location integrated along the ray. Several papers modify this approach, such as Yu et al. (2021) represent points in three-dimensional space as individual voxels, while the additional two dimensions of view direction are accommodated by applying spherical harmonics to the voxels. Müller et al. (2022) provide an alternative expansion to NeRF, allowing near-instant model rendering through the use of a smaller rendering network. This smaller network is facilitated by the use of sophisticated neural graphics primitives for input encodings.

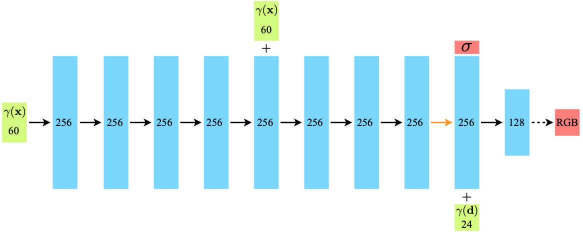


Figure 1: Original NeRF network. The network is a fully-connected multi-perceptron with 10 layers of illustrated widths. Network inputs are shown in green, and outputs (density σ and colour RGB) are shown in red.

2.2 Style Transfer

Several methods exist for style transfer in the realm of image manipulation which are of relevance to this work. Some models for style trasnfer, such as Gatys et al. (2016), develop specific neural networks which are capable of separating images into their style and their content. Images can then be transferred across styles by combining the style representation of one image with the content representation of another. The method applies convolutional neural networks are trained to learn generic feature representations of an image. The process then follows a standard texture transfer method while using deep representations for style features Heeger & Bergen (1995).

An alternative method implements an encoder-decoder structure for the same purpose. Zhang et al. (2020) propose a method which implements four neural networks: two encoders, one for style and one for content, one mixer and one decoder. A single image's style can be adapted by encoding the data, applying the mixer with a target style, and subsequently decoding.

Also similar to style transfer, Srinivasan et al. (2020) propose a method for relighting NeRF models by also training for a model's surface normal, which permits arbitrary relighting.

2.3 NeRF Style Adaption

Several methods have also been proposed to adapt the aesthetic style of NeRF models. Chen et al. (2022) propose a method of style transfer in which target styles may be drawn from sample images. The authors first pre-train a 2D photorealistic style transfer network. Next, a voxel representation of the NeRF scene is acquired, before a hypernetwork is used to constrain the NeRF rendering of the scene from various views.

Nguyen-Phuoc et al. (2022) expands on this method, reducing the computational demand of the process by alternately training the style transfer network and the NeRF model. As well as reducing computational demand, this method improves the style transfer as it reduces the jittering effect introduced by the simultaneous method of training a style transfer and neural rendering network.

3 Methods

Firstly in this section, a description is given in section 3.1 of the way in which the standard NeRF model is adapted, being divided in two. Section 3.2 describes how the network is trained.

3.1 NeRF Base

The standard NeRF model implements a standard multi-layer perceptral network to map a point in three-dimensional space with a view direction to the colour and spatial density σ of that point. The colour is represented by a three-dimensional vector made up of the (r, g, b) components of the colour. The network consists of 10 fully connected layers. A full description of the network is shown in figure 1.

The network is modified here for the purposes of style transfer. Firstly, the existing network is adapted so that rather than outputting a three-dimensional vector representing colour (RGB), it outputs a

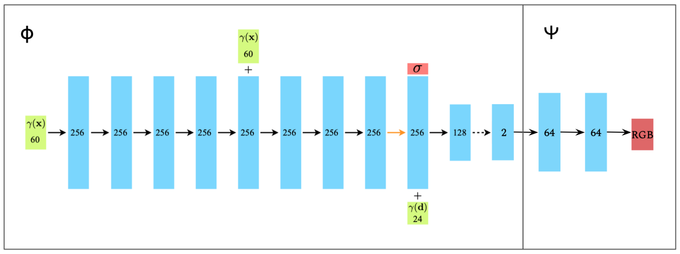


Figure 2: Proposed new network. The encoder on the left, Φ, maps a point in five-dimensional space to a point in a two dimensional feature space. This is decoded with a decoder network, shown under Ψ on the right.

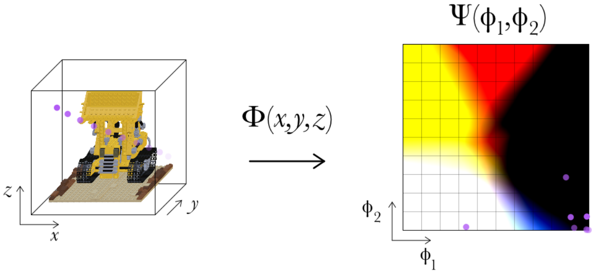


Figure 3: An illustration of the encoder-decoder setup. On the left, the original 3D model of the Lego Digger from Mildenhall et al. (2020). Purple dots represent a set of locations within the 3D space (and with an arbitrary viewing direction). These are mapped through Φ to the two-dimensional feature space on the right. This feature space is mapped through Ψ which returns the colour of the point in space. The colour represented at each point in the feature space is represented by the colours' representations on teh plane.

two dimensional vector $\phi = [\phi_1, \phi_2]^T$. The feature space is constrained to only two dimensions as this encourages the network to learn shared feature space representations for similar colours, improving the quality of style transfer. This is an arbitrary learned feature mapping of the content. The volumetric density σ is still output by the decoder, as this is considered a part of the content of the scene. This half of the network is termed the encoder and is represented by the function $\Phi : \mathbb{R}^5 \to (\mathbb{R}, \mathbb{R}^2)$.

This network is then appended with a second network which maps the feature vector representation of the input spatial coordinate to a colour. The network is a simple multi-layer perceptron. The network complexity was selected to maximise the expressive capacity of the network while minimising the risk of breakdown which was found to occur when larger network layers were used. This second network is referred to as the decoder, as it transforms a point in the feature space of the network to a colour. The function of the network is represented by $\Psi : \mathbb{R}^2 \to \mathbb{R}^3$. The full new network is shown in figure 2.

Figure 3 shows the network as it might be applied to learn the features of a single standard NeRF model.

3.2 Parallel Training

The model shown in figure 2 shows a single connected network, it is possible to separate the model between the encoder and decoder. For our purposes, we use the decoder to change the style of a model encoded by the encoder. To produce a second decoder to perform this style transfer, a second scene is introduced, which must be rendered separately using conventional software such as Blender Foundation (2023). To distinguish the two scenes from one another, we refer to the original scene as the target, and the second scene as the reference.

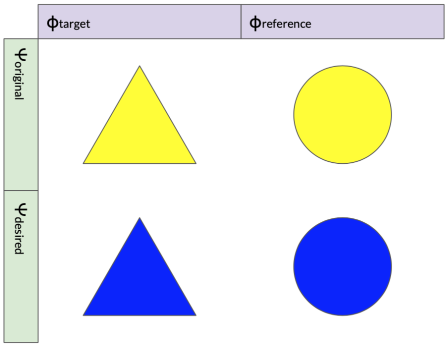


Figure 4: Principle of the style transfer principle outlined in section 3.2. In this case the yellow triangle is our target scene, which we with to adapt to a target state in which the style of the triangle is changed to blue. To do this, a reference scene is created with arbitrary geometry in two styles: one yellow and one blue. The three scenes are used to train two encoders and two decoders (Φ_{target}, $\Phi_{\text{reference}}$ and Ψ_{original}, Ψ_{desired} respectively). To generate the target scene in the desired style, Φ_{target} is used in combination with Ψ_{desired}.

The reference scene is rendered in two versions, with the first designed so that its style resembles that of the target scene. The second version of the scene is rendered in the desired modified style. In total three sets of rendered scenes are therefore used to train this style transfer model. The first set is the target scene in the original style. The second is the reference scene in the original style. The third is the reference scene in the target style.

These three scenes are used to train a NeRF module consisting of two encoders and two decoders. The target scene is used to train an encoder Φ_{target} while the reference scene is used to train an encoder $\Phi_{\text{reference}}$ in both the original style and the desired style. The target scene in the original style and the reference scene in the original style are both used to train decoder Ψ_{original} while the reference scene in the desired style is used to train decoder Ψ_{desired}.

Once these networks are trained, it is possible to construct the target scene in the desired style by applying the encoder Φ_{target} with decoder Ψ_{desired}. A very simple illustration of this principle is shown in figure 4.

4 Analysis

The method proposed relies on the assumption that two similarly-coloured points in two different scenes will be encoded to two similar points in the feature space of the model. This facilitates the style change, as it means that when a point in the feature space is remapped by changing the decoder for the reference scene, a similar change in colour occurs in the target scene. The assumption is based on Occam's Razor: that representing two points of similar colour in the same location in a feature space demands a simpler network than separating them, as to separate them would require creating two distinct regions in the feature space which map to the same colour through the decoder.

To test this assumption, testing was conducted using the reference scene described in section 5.1 and the standard lego digger described in Mildenhall et al. (2020). Two encoders and a single decoder was used to decode both scenes. This test was conducted four separate times.

4.1 Kullback-Liebler Divergence Testing

The Kullback-Liebler Divergence of two probability distributions $p(x)$ and $q(x)$ is given by:

$$D_{KL}(p \parallel q) = \int_{-\infty}^{\infty} p(x) \ln \frac{p(x)}{q(x)} dx \tag{1}$$

For our assessment, we fit a bivariate normal distribution to a sample of points in the feature space of both models given a random subset of input points to the encoder sampled from points used to

Table 1: Comparison of KL-divergence of a random sample of points in the feature spaces of the two encoders. This metric is shown tested across four separate tests but with identical configurations.

	Run 1	Run 2	Run 3	Run 4
KL Divergence	101	86	69	84

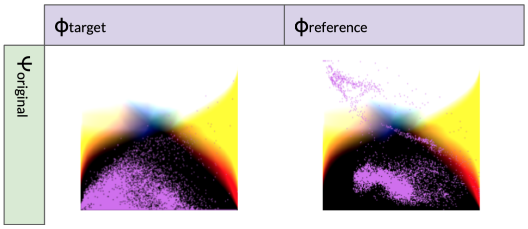


Figure 5

render the models. The Kullback-Liebler divergence of a multivariate distribution is found according to Duchi (2016):

$$D_{KL}(p \parallel q) = \frac{1}{2}(\log \frac{\det \Sigma_q}{\det \Sigma_p} - n + \text{tr}(\Sigma_q^{-1}\Sigma_p) + (\mu_q - \mu_p)^T \Sigma_q^{-1}(\mu_2 - \mu_1)) \quad (2)$$

Conducting analysis by observing only one divergence would be inappropriate, since it could be unreasonable to expect that a high KL-divergence is necessarily associated with a poor correlation between the sets, as this measure is vulnerable to externalities such as the semantic similarity of the scenes. However, if it is found that the KL-divergence changes significantly when the method is applied several times to the same set of scenes, this would suggest that the assumption made is false, since no trends in the overlapping regions of the scenes emerge. Results from the test are shown in table 1.

The high variance found of the KL divergence implies that the assumption does not hold strongly. This would justify the use of regularisation of this feature, which is considered in section 6.2. The concern is illustrated in figure 5. Purple dots show a random set of input points mapped to the feature space by the two encoders. The difference in the distribution of these points limits the quality of style transfer which will be possible.

5 Results

In this section we describe three different style transfers which were performed using the methodology above. In all cases, each network was trained over 500,000 iterations.

Before these results are shared, a short description is given of the reference scene in the original style. All style transfers were performed with a target scene consisting of the Lego digger from Mildenhall et al. (2020).

5.1 Reference Style

The reference scene was designed so that it contained the colours represented in the target scene, and so that it shows these colours over a wide range of lighting conditions. As such, the reference scene consists of a set of five isometric spheres of colours yellow, red, brown, black, and grey. These colours were selected since they are all represented prominently in the target scene. The spherical shape was chosen since it means that the colours are represented at all possible lighting angles. The scene was lit with a combination of direct lighting and HDRI lighting from Sergej Majboroda (2023). The reference scene is illustrated in its original style in figure 6.

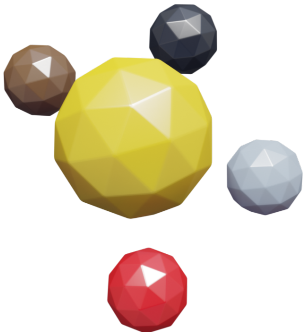
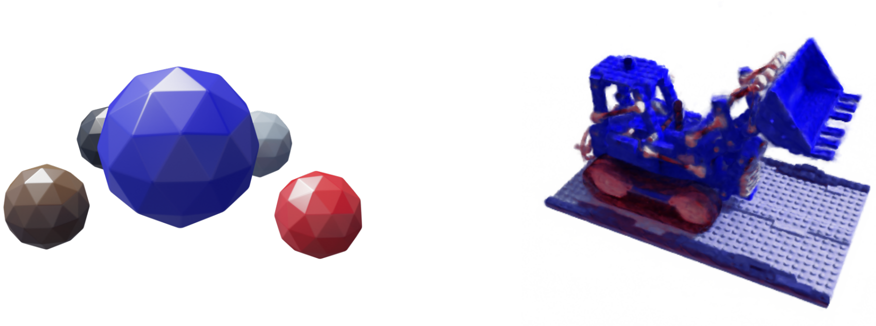


Figure 6: Reference scene in original style.



(a) Reference scene in blue desired style.

(b) Target scene in desired style.

Figure 7: The colour change style.

5.2 Desired Style: Colour Change

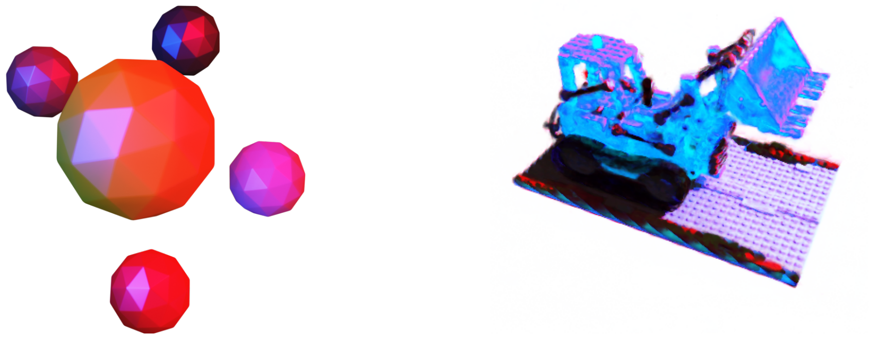
In the first style adaption, the reference scene was modified so that the central, largest sphere was changed to be blue rather than yellow. The reference scene in the desired style is shown in figure 7a.

Following model training, the target scene in the desired style is shown in figure 7b. The results are mixed. The digger has changed colour successfully, and is evidently blue, rather than yellow. Promisingly, the lighting and shadows on the digger have evidently been preserved, indicating that the transfer of style was successful. However, there are clearly inaccuracies in the rendered model. Some elements, such as the red light on top of the digger, have been rendered in blue rather than red.

5.3 Desired Style: Lighting Change

In this second scene, the natural environment lighting was replaced with two orthogonal lights; one red and one blue. Each light illuminates a different side of each sphere. The scene is shown in figure 8a.

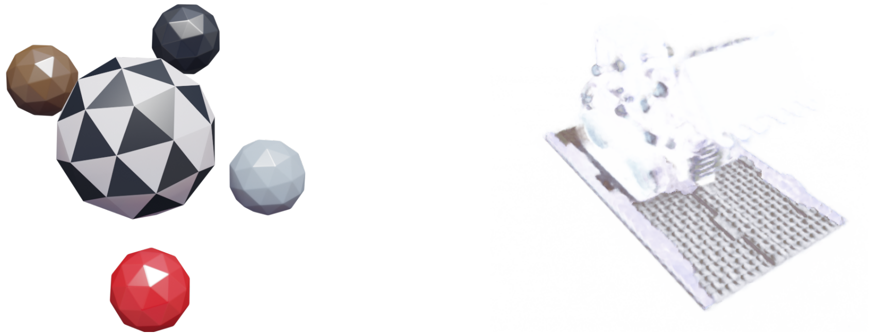
Once again, the model is trained and results are shown in figure 8b. The results are mixed. As before, the digger has changed colour successfully. Particularly promisingly, the colour shifts dramatically as the camera angle changes, quickly moving between red and blue in many cases (such as the blade of the digger's shovel). However, there is a mismatch between the desired style of the target scene and the reference scene. The target scene is brighter than the reference scene, and it does not appear that the yellow of the digger has been re-lit. Rather, it appears re-coloured.



(a) Reference scene in desired re-lit style.

(b) Target scene in desired re-lit style.

Figure 8: The re-lit style.



(a) Reference scene in desired check style.

(b) Target scene in desired check style.

Figure 9: The re-lit style.

5.4 Desired Style: Pattern Change

Finally, in the most complex case, the pattern of the yellow sphere in the original scene is modified to be a white and black check pattern. This requires a more complex decoder than previous networks. The scene is shown in figure 9a.

Results following training, shown in figure 9b show that this test failed entirely. This exemplifies the failure of this model to adapt to very complex style adaptions. The model has clearly experienced a form of catastrophic failure, since the deck of the model has been rendered in white, despite the reference colour for the deck not changing in the reference scene.

5.5 Quantitative Comparison

While performing a quantitative comparison of the target scene rendered in the desired style is not possible since this scene has no ground truth, it is possible to compare the peak signal to noise ratio (PSNR) across each of the three training scenes. In addition to the original training scenes, the original NeRF model is trained over the same number of epochs on the target and reference scenes in the original styles. This serves as a point of comparison for the quality of the network outputs. This comparison is shown in table 2.

These results show that, when the models are trained as joint models (i.e. three models are trained simultaneously while sharing encoders and decoders), the PSNR reduces compared to when training is undertaken without the joint property. This is expected, revealing slight inconsistencies between the styles of the reference model compared to the target model, which reduces the representational capacity of the encoder-decoder arrangement. Despite this reduction in quality, the results are

Table 2: Comparison of the PSNR of each of the style-adapted scenes as described above. Original indicates the original NeRF model (without encoder/decoder structure) trained on the two scenes in the original style.

	PSNR (↑)		
Scene	$\Phi_{\text{target}}, \Psi_{\text{original}}$ (Original Digger)	$\Phi_{\text{reference}}, \Psi_{\text{original}}$ (Original Spheres)	$\Phi_{\text{reference}}, \Psi_{\text{desired}}$ (Desired Spheres)
Without Joint Training	27.7	36.6	N/A
Colour	25.7	36.0	35.3
Lighting	25.1	31.3	30.7

promising as they reveal that training the various models simultaneously while sharing modules does not reduce the quality of the models to a very significant extent.

6 Discussion and Conclusion

In this section we begin by briefly discussing the results above. We proceed to provide an illustrative example of the origin of some of the errors which arose above. Finally, we discuss improvements which could be made to correct these errors.

6.1 Discussion

The principle hypothesis of this report is that we may adapt the style of a three-dimensional model rendered using NeRF by training a separate reference scene in two styles, and use the desired style of this reference scene to adapt the style of the target scene. The evidence found partially supports this hypothesis, but drawbacks to the method proposed mean that significant revisions would be required to the model before an entirely successful implementation could be made.

The method used above is successful insofar as it shows a strong capacity for transforming the colour space of an existing scene. This is supported by the evidence from the blue scene, which was successfully transformed by the style transfer. Further, the model shows high expressive capacity for transferring styles with complex view-dependent lighting conditions. This is evidenced in the second scene, where the lighting changed dramatically across view directions, and this was replicated successfully in the target scene.

However, when adaptions of the scene are complex, such as the third scene tested, the model shows a low capacity for adequate expression. Indeed, in these cases it appears that the network has a tendency towards catastrophic failure, in which no aspect of the style is adequately transferred. In a second category of failures, there were several instances in less complex scenes where elements in the target scene with desired style did not render properly, such as the red light on top of the digger in the scene with a colour change.

6.2 Future Work

There are two potential solutions which might be considered for improving the quality of style transfer, correcting issues introduced in section 4. Firstly, we might consider a simple loss term attached to the extent of overlap between the two sets of points in the feature space. This would encourage the network to learn equivalent mappings for equivalent colours in the 3D space.

Secondly, we might consider the use of a discriminator to improve overlap. In this case, we might train a discriminator network as outlined in Goodfellow et al Goodfellow et al. (2014) to attempt to discern mappings of points in the reference scene to those in the target scene. Training to minimise the discriminator's ability to discern the mappings of one set of points from the other would also improve the extent of overlap between the two sets of points.

References

Chen, Y., Yuan, Q., Li, Z., Liu, Y., Wang, W., Xie, C., Wen, X., and Yu, Q. Upst-nerf: Universal photorealistic style transfer of neural radiance fields for 3d scene, 2022.

Duchi, J. C. Derivations for linear algebra and optimization. 2016.

Foundation, B. Blender foundation, 2023. URL `https://www.blender.org/about/foundation/`.

Gatys, L. A., Ecker, A. S., and Bethge, M. Image style transfer using convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2414–2423, 2016. doi: 10.1109/CVPR.2016.265.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks, 2014.

Heeger, D. and Bergen, J. Pyramid-based texture analysis/synthesis. In *Proceedings., International Conference on Image Processing*, volume 3, pp. 648–651 vol.3, 1995. doi: 10.1109/ICIP.1995.537718.

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, 2020.

Müller, T., Evans, A., Schied, C., and Keller, A. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. doi: 10.1145/3528223.3530127. URL `https://doi.org/10.1145/3528223.3530127`.

Nguyen-Phuoc, T., Liu, F., and Xiao, L. Snerf: Stylized neural implicit representations for 3d scenes, 2022.

Sergej Majboroda, J. G. Industrial sunset (pure sky) hdri • poly haven, 2023. URL `https://polyhaven.com/a/industrial_sunset_puresky`.

Srinivasan, P. P., Deng, B., Zhang, X., Tancik, M., Mildenhall, B., and Barron, J. T. Nerv: Neural reflectance and visibility fields for relighting and view synthesis, 2020.

Yu, A., Fridovich-Keil, S., Tancik, M., Chen, Q., Recht, B., and Kanazawa, A. Plenoxels: Radiance fields without neural networks, 2021.

Zhang, Y., Zhang, Y., and Cai, W. A unified framework for generalizable style transfer: Style and content separation. *IEEE Transactions on Image Processing*, 29:4085–4098, 2020. doi: 10.1109/TIP.2020.2969081.