Brush Stroke Parameterized Style Transfer

Uma Maheswara R Meleti
Clemson University
Clemson, SC, USA
umeleti@clemson.edu

Abstract

Computer Vision-based Style Transfer techniques have been used for many years to represent artistic style. However, most contemporary methods have been restricted to the pixel domain; in other words, the style transfer approach has been modifying the image pixels to incorporate artistic style. However, real artistic work is made of brush strokes with different colors on a canvas. Pixel-based approaches are unnatural for representing these images. Hence, this paper discusses a style transfer method that represents the image in the brush stroke domain instead of the RGB domain, which has better visual improvement over pixel-based methods. https://maheshmeleti.github.io/param-brushstroke/

1. Introduction

Style transfer has been an essential technique in computer vision, extensively researched for decades. It is now widely used in various applications such as animated content, fashion design, mobile app photo filters, etc. The technique enables the transformation of images by applying the visual style of one image (the style image) to another (the content image). The groundbreaking work of Gatys et al. [3], which introduced the concepts of content and style loss, has paved the way for many subsequent developments in the field. Most works before Gaty's method and subsequent methods operate within the pixel domain, where image pixels are manipulated to incorporate artistic styles.

However, all these contemporary methods fall short of mimicking the actual artistic work characterized by brush strokes and texture on a canvas. Pixel-based methods often produce images lacking the natural flow and feel of brush strokes.

In contrast to pixel-based methods, this paper introduces a novel approach that performs style transfer in the brush stroke domain rather than the RGB pixel domain. By parameterizing brush strokes and defining their location, color, width, and shape, our method aims to generate vi-

sually closer results to hand-painted artwork. This representation better preserves the artistic integrity of the original style, providing a more realistic and visually appealing output.

The brush strokes are modeled as a bezier curve with color, width, and location on the canvas, and these are optimized to produce stylized outputs. We have introduced a renderer that maps brush strokes into pixel values on canvas. The render is made differentiable, allowing gradients to backpropagate and optimize the brush stroke parameters for producing stylized outputs.

2. Related Work

In the earlier works on style transfer, Efros and Freeman [2] performed texture synthesis and transfer using image quilting, an algorithm that stitches small blocks of texture from an image in a way that resembles the input image. Hertzmann et al. [6] proposed a method that uses a pair of images—one being a filtered version of the other—to learn a transformation filter. Given an image pair, A and A', the algorithm learns how pixel neighborhoods (or textures) in image A correspond to those in image A'. This learned filter can then be applied to a new image, B, effectively transferring the learned texture transformation to the new image.

The most seminal work on style transfer was proposed by Gatys et al. in their paper *Image Style Transfer Using Convolutional Neural Networks* [3]. In this approach, the content image and style image are used as inputs to generate a new image that retains the content of the content image while adopting the style of the style image. The method introduces a novel loss function that separates content and style components, utilizing a pre-trained neural network to extract content and style representations from the given images.

Later, Huang and Belongie improved style transfer using feed-forward networks for arbitrary style transfer by introducing Adaptive Instance Normalization (AdaIN) [7]. AdaIN is a layer (or block) that aligns the mean and variance of the content features with those of the style features. This block transforms the extracted features of con-
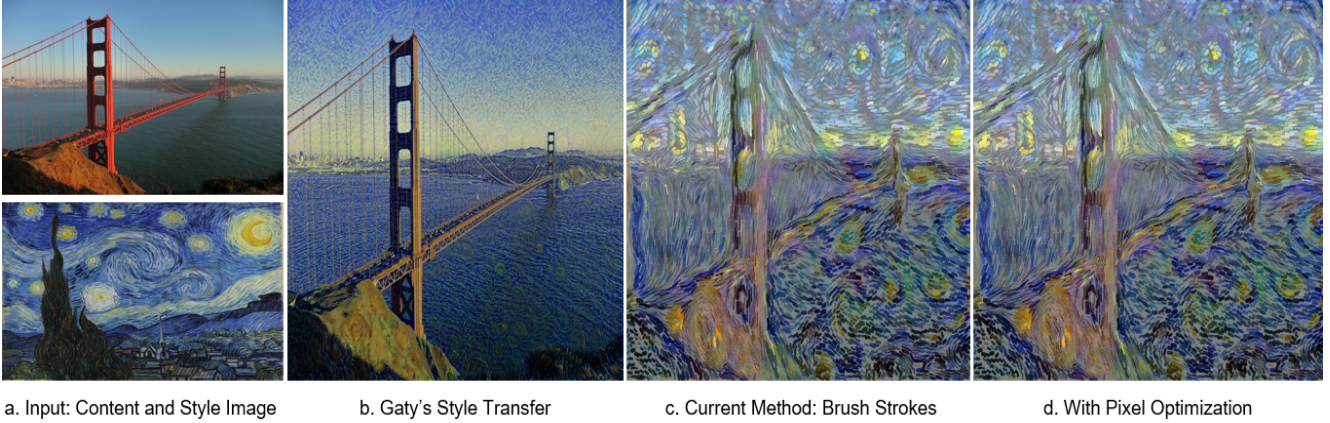
| a. Input: Content and Style Image | b. Gaty's Style Transfer | c. Current Method: Brush Strokes | d. With Pixel Optimization |

Figure 1. B represents the output of Gatys' style transfer method, while C is the result of our proposed approach. Visually, our method produces an image that more closely resembles artistic brush strokes compared to Gatys'. D is the output after pixel optimization, where the brush strokes are blended, resulting in a more cohesive and refined representation.

tent and style images from a fixed VGG-19 network, and a decoder is used to invert the AdaIN output back into the image space.

Traditional content loss measures how well the structure of the content is preserved in the stylized output. However, this approach doesn't always account for the effects of stylization, which can sometimes result in a blurred or less accurate rendering, particularly in high-resolution images. To address this, Sanakoyeu et al. and Kotovenko et al. introduced style-aware content losses [11] to enhance stylization quality. These losses modify traditional content loss by considering the style features, which better balance content preservation and stylization.

In addition to pixel-based approaches, some methods focus on rendering brush strokes. Early works in this area include an interactive technique by Haeberli [4], where a program follows the cursor across the canvas, samples the source image at each point to obtain a color, and then paint a brush stroke of that color. Hertzmann [5] extended this approach with an automated algorithm that takes a source image and a list of brush sizes, then paints multiple layers—one for each brush size—on a canvas to recreate the source image with a hand-painted effect. In contrast to these stroke-based rendering methods, some attempts focus on detecting and extracting brush strokes from paintings. POET [1] uses classical computer vision techniques, such as circular filters and orientation phase extractors, to automatically extract brush strokes. Jia Li et al. [9] employ edge detection and clustering-based segmentation methods to extract brush strokes, enabling comparisons between artistic works and other artists' styles.

Even though there has been a significant amount of work on artistic style transfer, the authenticity of brush strokes in paintings is often not preserved on the canvas during stylization. In contrast to the above-mentioned methods, the pro-

posed method utilizes parameterized brush strokes to retain the artwork's original texture and stroke characteristics. Using these parameterized brush strokes, the method aims to better preserve the artistic integrity and hand-painted quality of the source image while applying the desired style.

3. Approach

The proposed method is inspired by the iterative style transfer approach of Gatys et al. [3], where a white noise image is optimized via gradient descent using content and style loss (see Section 3.2) to stylize the content image. In the proposed approach, the optimization process begins by initializing a neural network with N brush stroke parameters — location, color, width, and curve shape, where N represents the desired number of brush strokes to fill the stylized canvas.

These parameters are iteratively optimized using the same content and style losses (as discussed in Section 3.2), ensuring the resulting canvas captures the content of the original image while applying the style of the reference image. The canvas is generated by a differentiable renderer, which takes the optimized brush stroke parameters and produces an RGB image. The brush strokes in this image are guided by their respective locations, widths, and colors. Further details on the differentiable renderer can be found in Section 3.1.

The shape of each brush stroke is modeled using a Bézier curve, as shown in eq. 1:

$$B(t) = (1-t)^2 P_0 + 2(1-t)t P_1 + t^2 P_2 \quad \text{for} \quad t \in [0, 1]. \tag{1}$$

After the brush stroke optimization is completed, a pixel-level optimization step is applied, akin to Gatys et al.'s style transfer method. This step blends the brush strokes and adds
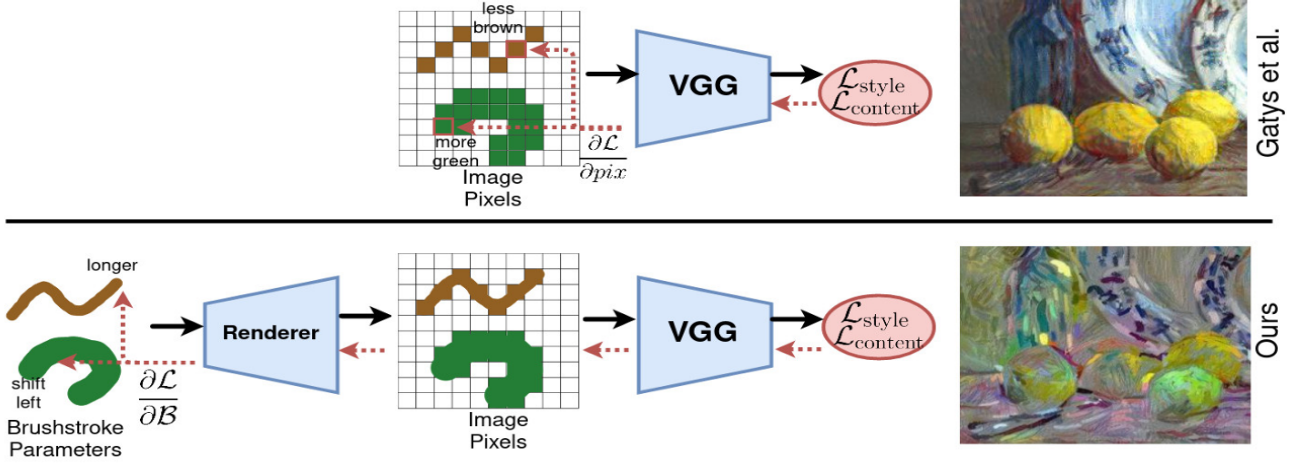
Figure 2. The top row shows the results of Gatys' style transfer approach, while the bottom row displays the output of the proposed method. In our approach, parameterized brush strokes are passed through a differentiable renderer, which maps them onto the canvas. The content and style losses are then calculated, and gradients are backpropagated through the renderer to optimize the brush strokes. The image is taken from [8]

finer texture details to the final image. Both outputs are presented in the experiments section 4.

3.1. Differentiable Renderer

A differentiable renderer maps a collection of brushstrokes (trained) parameterized by location, shape, width and color into pixel values on a canvas. A renderer can be described as a function:

$$R : \mathbb{R}^{N \times F} \to \mathbb{R}^{H \times W \times 3}$$

Where N is the number of brush strokes, F is the number of brushstroke parameters, which is 12 - location: 2, shape: 6 ($P_0 : 2, P_1 : 2, P_2 : 2$), width : 1, color : 3.

Let's try to understand how the rendering works with a simple example of rendering a flat disk parameterized by its color C, radius R, and location (1, 1, and 2 scalars). We assume our canvas is grayscale, which is just a 2D matrix of pixel values (this can be generalized to RGB space). Our task is to decide whether each pixel belongs to the disk; we compute the L2 distance matrix D from the disk's center and mask all the values with the color C where $D < R$. If I_1 and I_2 are two images obtained for two disks using this procedure, we can get the final image as $I_1 + I_2$, assuming the disks do not overlap.

If the regions overlap, we compute an assignment matrix A, where the assignment is made to a particular disk if the distance is less than the other disk's. If there are two disks with distance matrices D_1 and D_2, we define $A := \{1 \text{ if } D_1 \leq D_2, 0 \text{ otherwise}\}$. This operation can be generalized to N objects.

$$A(i, j, n) := \begin{cases} 1 & \text{if } D_n(i, j) < D_k(i, j) \, \forall k \neq n, \\ 0 & \text{otherwise.} \end{cases}$$

The final image for N discs $I_1, \ldots, I_N$ is computed as

$$I(i, j) := \sum_{n=1}^{N} I_n(i, j) \cdot A(i, j, n)$$

To render the disk, we calculate distance matrices, which form the cornerstone of the rendering process. The same principle applies to rendering brushstrokes (or Bézier curves): we compute the distance matrix D_B by point sampling S equidistant points $p_1, \ldots, p_S$ along the curve and mask the values that are less than the brushstroke's width.

For practical implementation, the renderer must be differentiable. However, the masking and assignment operations are discontinuous, making them non-differentiable. To resolve this, we make both operations continuous by using a sigmoid function for masking and a softmax function with high temperature for assignment. Additionally, computing the distances between each brush stroke and every pixel is computationally expensive and redundant, as each brush stroke only affects nearby areas of the canvas. Therefore, the distance computation is limited to the K nearest brush strokes.

3.2. Loss Functions

Content and style losses are essential for neural style transfer, guiding the balance between preserving the content im-

age's structure and applying the style image's artistic features.

Content Loss: The content loss ensures that the generated image G retains the structure of the content image C. It is computed by comparing the feature maps ϕ_l of the content and generated images at a specific layer l of a pre-trained network (e.g., VGG-19):

$$\mathcal{L}_{\text{content}}(C, G) = \frac{1}{2} \sum_{i,j} (\phi_l(G)_{ij} - \phi_l(C)_{ij})^2 \quad (2)$$

Style Loss: The style loss measures how well the generated image captures the style image S. It uses the Gram matrix $G_l(x)$ of feature maps to compute correlations between filter responses. The style loss is defined as:

$$\mathcal{L}_{\text{style}}(S, G) = \sum_l w_l \cdot \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_l(G)_{ij} - G_l(S)_{ij})^2 \quad (3)$$

where N_l and M_l are the number of filters and the spatial size of the feature maps at layer l, and w_l is the weight for each layer.

Total Loss: The total loss $\mathcal{L}_{\text{total}}$ is a weighted sum of the content and style losses:

$$\mathcal{L}_{\text{total}} = \alpha \mathcal{L}_{\text{content}} + \beta \mathcal{L}_{\text{style}} \quad (4)$$

where α and β control the trade-off between content preservation and style transfer.

4. Experiments and Results

The model is trained on Nvida-a100 GPUs using the pytorch framework. Optimizing 5000 brush strokes with 10 samples per curve took around 138 seconds.

Figure 1 presents the results from both methods, where our approach yields visually superior representations that are artistically closer to hand-painted works. Figure 3 is the zoomed images highlighting the brush strokes generated by our method, followed by the pixel optimization applied to enhance the output.
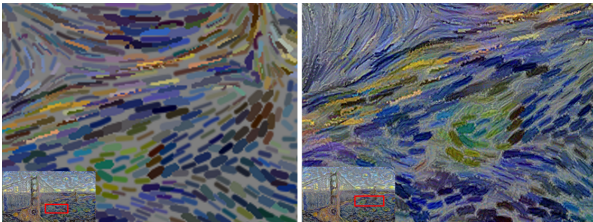

Figure 3. Zoomed view of brush strokes with pixel optimization.

The pixel-optimized output blends the brush strokes, producing a realistic image that mimics the texture and look of a painting on canvas.

Figure 4 is the result of applying our algorithm to a human image. While the algorithm effectively captures the brushstroke representation, finer details, such as facial features, are lost.

Figure 4. Style transfer applied to a human image.

This indicates that the current method struggles to preserve intricate details, suggesting the need for improvements in handling high-frequency content.

5. Conclusions

In this paper, we proposed a new approach for representing artistic style by switching from pixel-based methods to parameterized brush strokes for more natural representations. An explicit rendering mechanism is implemented that can be applied beyond style transfer.

Even though we obtained a brush stroke-based representation, qualitative results indicate some fine details of the content are missing. Implementing CNN-based feed-forward architectures can help mitigate these issues, as the convolutional inductive biases in CNNs effectively preserve rich information on image attributes. By leveraging the hierarchical feature extraction capabilities of CNNs, we can better capture subtle textures and fine-grained information.

Integrating image and text-based methods such as CLIP [10] can significantly advance research by enabling more sophisticated image editing techniques, offering users finer control over the generated results to better align with their specific goals with the help of language.

References

[1] Igor Berezhnoy, Eric Postma, and H. Herik. Automatic extraction of brushstroke orientation from paintings : Pppoet: Prevailing orientation extraction technique. *Machine Vision and Applications*, 20:1–9, 2008. 2

[2] Alexei A. Efros and William T. Freeman. Image quilting for texture synthesis and transfer. *Proceedings of SIGGRAPH 2001*, pages 341–346, 2001. 1

[3] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2414–2423, 2016. 1, 2

[4] Paul Haeberli. Paint by numbers: abstract image representations. In *Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques*, page 207–214, New York, NY, USA, 1990. Association for Computing Machinery. 2

[5] Aaron Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, page 453–460, New York, NY, USA, 1998. Association for Computing Machinery. 2

[6] Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. *Image Analogies*. Association for Computing Machinery, New York, NY, USA, 1 edition, 2023. 1

[7] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization, 2017. 1

[8] Dmytro Kotovenko, Matthias Wright, Arthur Heimbrecht, and Björn Ommer. Rethinking style transfer: From pixels to parameterized brushstrokes, 2021. 3

[9] Jia Li, Lei Yao, Ella Hendriks, and James Z. Wang. Rhythmic brushstrokes distinguish van gogh from his contemporaries: Findings via automated brushstroke extraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1159–1176, 2012. 2

[10] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 4

[11] Artsiom Sanakoyeu, Dmytro Kotovenko, Sabine Lang, and Björn Ommer. A style-aware content loss for real-time hd style transfer, 2018. 2