

Article

Comparing Neural Style Transfer and Gradient-Based Algorithms in Brushstroke Rendering Tasks

Artur Karimov ^{1,*}, Ekaterina Kopets ¹, Tatiana Shpilevaya ², Evgenii Katser ², Sergey Leonov ³
and Denis Butusov ^{2,*}

¹ Youth Research Institute, Saint Petersburg Electrotechnical University “LETI”,
197376 Saint Petersburg, Russia

² Computer-Aided Design Department, Saint Petersburg Electrotechnical University “LETI”,
197376 Saint Petersburg, Russia

³ Public Relations Department, Saint Petersburg Electrotechnical University “LETI”,
197376 Saint Petersburg, Russia

* Correspondence: aikarimov@etu.ru (A.K.); dnbutusov@etu.ru (D.B.)

Abstract: Non-photorealistic rendering (NPR) with explicit brushstroke representation is essential for both high-grade imitating of artistic paintings and generating commands for artistically skilled robots. Some algorithms for this purpose have been recently developed based on simple heuristics, e.g., using an image gradient for driving brushstroke orientation. The notable drawback of such algorithms is the impossibility of automatic learning to reproduce an individual artist’s style. In contrast, popular neural style transfer (NST) algorithms are aimed at this goal by their design. The question arises: how good is the performance of neural style transfer methods in comparison with the heuristic approaches? To answer this question, we develop a novel method for experimentally quantifying brushstroke rendering algorithms. This method is based on correlation analysis applied to histograms of six brushstroke parameters: length, orientation, straightness, number of neighboring brushstrokes (NBS-NB), number of brushstrokes with similar orientations in the neighborhood (NBS-SO), and orientation standard deviation in the neighborhood (OSD-NB). This method numerically captures similarities and differences in the distributions of brushstroke parameters and allows comparison of two NPR algorithms. We perform an investigation of the brushstrokes generated by the heuristic algorithm and the NST algorithm. The results imply that while the neural style transfer and the heuristic algorithms give rather different parameter histograms, their capabilities of mimicking individual artistic manner are limited comparably. A direct comparison of NBS-NB histograms of brushstrokes generated by these algorithms and of brushstrokes extracted from a real painting confirms this finding.

Keywords: non-photorealistic rendering; brushstroke rendering; neural style transfer; oil paintings; postimpressionism; realism; pointillism; brushstroke style; statistical analysis; painting robot

MSC: 90C90; 90C59; 68U10



Citation: Karimov, A.; Kopets, E.; Shpilevaya T.; Katser E.; Leonov S.; Butusov, D. Comparing Neural Style Transfer and Gradient-Based Algorithms in Brushstroke Rendering Tasks. *Mathematics* **2023**, *11*, 2255. <https://doi.org/10.3390/math11102255>

Academic Editor: Konstantin Kozlov

Received: 6 April 2023

Revised: 24 April 2023

Accepted: 30 April 2023

Published: 11 May 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Throughout history, technology has greatly expanded the creative and professional possibilities of artists, providing them with new and more powerful tools and enabling them to create novel artistic styles and art forms. Computer graphics, and in particular non-photorealistic rendering (NPR), have a great influence on the development of contemporary art and are often used to create web content [1–5], for robotic painting [6–8], as a tool for creating comics [9], and, of course, for imitating artistic paintings [10,11].

NPR algorithms can be divided into two main classes: pixel-based and stroke-based rendering. Pixel-based rendering (PBR) implies operation with bitmaps. Such types of algorithm are more widespread due to the way images are stored on computers. Nowadays,

the most frequently encountered pixel-based rendering technique for mimicking artistic paintings is neural style transfer (NST) [12,13]. Neural style transfer was first proposed in a paper by Gatys et al. [14]. The original algorithm used a convolutional neural network (CNN) trained on paintings by well-known artists. The network extracted the style from the paintings and applied it to the selected photo.

Since then, many improvements to neural style transfer have been proposed. For example, one of the possible solutions is the use of a generative adversarial neural network (GAN) instead of a convolutional neural network. A GAN, proposed by J. Goodfellow et al. [15], consists of two parts, a generator, and a discriminator. While training, they compete with each other: the generator learns to “fool” the discriminator, which in its turn learns to better distinguish the generated images from the real ones [16–18]. In the papers [19,20], a feed-forward neural network for style transfer was used, which made it possible to decrease the running time of the method.

Stroke-based rendering (SBR) is another approach to mimicking artistic paintings. The paper [21] defines SBR as a process of generating a digital artwork using special *rendering marks*, for example, hatches, brushstrokes, or other primitives. For convenience, we will further consider only brushstrokes as rendering marks. Such algorithms do not necessarily require machine learning and sometimes can synthesize brushstrokes using simple heuristics, for example, determine brushstrokes randomly, or apply them in accordance with image edges or image gradient, or using any other local image features [21–23]. This determined the relative popularity of such algorithms before the mid-2010s when the machine learning boom had begun. A special feature of these algorithms is that brushstrokes are generated in an explicit manner, contrary to PBR, where the image consists of pixels and brushstroke texture is only an illusion. This provides multiple options inaccessible to common PBR algorithms: generating realistic-looking artworks using brushstroke libraries [24,25], controlling painting robots equipped with artistic tools [6,26–29], or creating labeled data for training brushstroke extraction algorithms [30].

A promising idea is to combine machine learning and SBR. The first attempts at this go back to the mid-2010s [31], but only recently have several powerful algorithms of this type been developed [32]. One of the most interesting works in this field is [33], where the authors proposed training an NST algorithm to minimize the loss function associated not with pixels but with parameterized brushstrokes. This improved a common drawback of the traditional NST approach, which often ignores or distorts brushstroke patterns. However, it also opens prospects of the neural style transfer algorithms to be used in robotic artistic painting or even in robot-assisted painting restoration, once the program learns to capture an individual artist’s manner well.

Both the heuristic and the NST algorithms have their pros and cons. Heuristic algorithms are simpler to implement, have shorter running times, and need less powerful computers. However, they usually have a huge number of parameters, manually set by the developer or the user, and their automated tuning to the individual style of a particular artist is difficult. While the heuristic algorithm tries to mimic the way an artist thinks during painting, the NST algorithm provides a quite opposite approach, trying to imitate the final result, close to an original style, in terms of optimizing the loss function. Its design seems to be more versatile and should provide better image stylizing.

Nevertheless, no extensive research on quantifying the image stylizing qualities of the NST and heuristic approaches has been conducted. Thus, in order to clarify this question, we propose a new experiment design for comparing two brushstroke rendering algorithms. In our experiment, we consider six brushstroke features in images generated by these algorithms from a test image set. For each feature, a histogram is built, and then, four correlation tests for these histograms are carried out: similar artist test, method 1 vs. method 2 test, similar image test, and real painting test. A detailed description of these tests is given in Section 3. As the result of applying our experimental approach, we obtain numerical estimates of whether these brushstroke rendering algorithms have different

rendering styles, and whether one or another algorithm is superior in style variation and mimicking a real artist's style.

The main findings of the paper are as follows:

- a method for quantifying brushstroke rendering results based on correlation analysis of feature histograms,
- a comparison of the heuristic algorithm from [8] with the NST algorithm from [33], using the proposed method.

According to the results, future research directions can be outlined, including additional steps for proper style learning, the introduction of different loss functions, and so on.

The paper is organized as follows. Section 2 provides an overview of current approaches for the efficient comparison of NPR algorithms. In Section 3, the materials and methods are described, test images are presented, and the brushstroke features under investigation are described. In Section 4.2, the results of the brushstroke analysis are presented. In Section 6, a discussion and brief conclusions are given.

2. Related Work

Methods for Evaluating the Effectiveness of NPR Algorithms

Non-photorealistic rendering (NPR) techniques have gained increasing popularity in recent years due to their ability to produce images with unique and artistic style. Methods for evaluating non-photorealistic rendering algorithms usually examine human perception, where people are asked to compare stylized images of different approaches and give their opinion [34–39]. However, there are also quantitative approaches for evaluating non-photorealistic rendering algorithms.

For example, Mandryk et al. [34,35] presented a study that aimed to evaluate the impact of non-photorealistic rendering (NPR) algorithms on participants' emotional responses. The study included five commonly used NPR approaches, two blurring techniques, and the original image. Participants rated the four dependent measures (arousal rating, valence rating, dominance rating, aesthetic rating) on a nine-point scale. According to the results, the NPR algorithms had a dampening effect on participants' emotional responses in terms of arousal and valence. Santella et al., in [36], suggested that viewers' attention is drawn to areas where detail is locally preserved in meaningfully abstracted images, as opposed to uniformly high- or low-detail images. The results of the study support the idea that artists use details to control interest and understanding. Additionally, the study suggests that eye tracking can be a useful tool for evaluating non-photorealistic rendering (NPR) systems.

A method for comparing non-photorealistic rendering images with hand-drawn images is proposed by Maciejewski et al. [40]. They compared hand-drawn and computer-generated stippling images, with large sample sizes, using GLCM (gray-level co-occurrence matrix) texture analysis. The authors found discrepancies between hand-drawn and computer-generated texture statistics, and that hand-drawn texture statistics have a higher correlation with real textures.

The task of choosing metrics for quantifying non-photorealistic rendering algorithms is indeed complex, as the choice of these metrics often depends on the goal of creating a non-photorealistic image. Fréchet inception distance (FID) is used to evaluate the quality of images generated by GAN [41,42]. In the paper, the authors presented empirical evidence that FID is a reasonable metric due to its robustness to network selection for mode-off and encoding. The scoring issues for neural style transfer are well explored in [43], which proposes a method (ArtFID) for quantifying style transfer models with respect to stylization efficiency. In [44], the authors propose three factors to evaluate and improve the quality of neural style transfer: content fidelity factor, global effects such as colors and textures, and a local patterns factor which consists of two parts, one is to assess the similarity of the local pattern counterparts directly, and the other is to compare the diversity of the retrieved pattern categories.

3. Materials and Methods

This section describes the investigated algorithms, and the test sets of images, giving some examples of artworks generated from these images.

3.1. Neural Style Transfer with Explicit Brushstrokes

In the work by Kotovenko et al. [33], an approach was proposed for explicit brush-stroke generation using neural style transfer technology. This approach is based on the original method of neural style transfer by Gatys et al. [14], in which image content elements and image style elements are retrieved and stored, and then, the image is iteratively generated from white noise until content and style losses are minimized. The content loss $\mathcal{L}_{content}$ is defined as the Euclidean distance between the rendered image I_r and the content image I_s in the VGG feature space:

$$\mathcal{L}_{content} = ||I_r^l - I_s^l||_2, \quad (1)$$

where l denotes the layer number of the VGG-19 network.

To extract the style features of the image, a gradient descent optimization is used, starting from the white noise to find another image that matches the Gram matrix representation of the input image. This minimizes the root-mean-square distance between the elements of the Gram matrix of the style image G_s^l and the Gram matrix of the rendered image G_r^l from the l -th layer of the VGG-19 network. The contribution of a layer l to the total losses equals:

$$E_l = \frac{1}{2N_l^2 M_l^2} ||G_r^l - G_s^l||_2,$$

where N_l is the number of feature maps in a layer l , and M_l is the height multiplied by the width of the feature map. Then, the total style loss $\mathcal{L}_{style}$ is:

$$\mathcal{L}_{style} = \sum_{l=0}^L \omega_l E_l, \quad (2)$$

For more details, the reader is referred to the original work [45].

In contrast to the method of Gatys et al., instead of optimizing the generated image pixel by pixel to minimize content and style loss, Kotovenko et al. optimize parameterized brushstrokes, minimizing content and style loss defined by equations similar to Equations (1) and (2). Brushstrokes are parameterized by location, color, width, and shape. The brushstroke shape is modeled as a quadratic Bézier curve:

$$\mathbf{B}(t) = (1-t)^2 \mathbf{P}_0 + 2t(1-t) \mathbf{P}_1 + t^2 \mathbf{P}_2, \quad t \in [0, 1]$$

A set of parameterized brushstrokes is converted to pixel values on the canvas using an explicitly differentiable function. The renderer is a function:

$$\mathcal{R} : \mathbb{R}^{N \times F} \rightarrow \mathbb{R}^{H \times W \times 3},$$

where N is the number of strokes, F is the number of stroke parameters, and H and W are the height and width of the image to render.

In order to extract brushstroke features, we supplied the program with an additional code, saving the brushstroke map generated by the program into a vector PLT file, as our previous program does [8,46].

Brushstrokes in PLT format are written in HP-GL/2 notation and require color, width, and sets of coordinates for each stroke. All the needed parameters are present in the renderer function. So, on the last call of the renderer function, the final parameters of the brushstrokes are duplicated into a file (see Figure 1) with the following modifications:

- for coordinates, 1 pixel is equated to 1 mm, which means coordinates are multiplied by 40 for the HP-GL coordinate system;
- for image readability, the initial range of widths is mapped onto an interval [4,8].

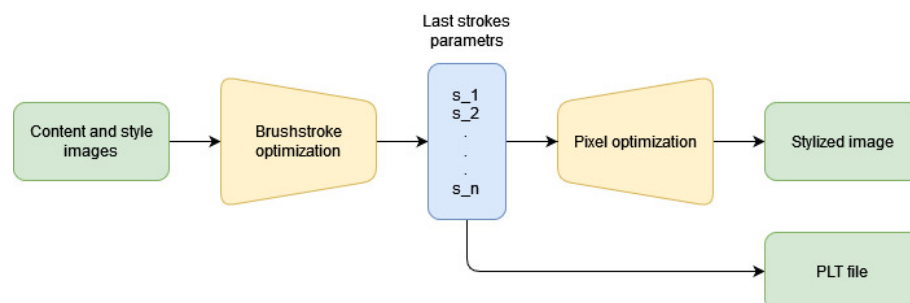


Figure 1. Fragment of the flowchart for the neural style transfer algorithm. Parameterized strokes are present only during brushstroke optimization; thus, they are exported right before the start of pixel optimization.

3.2. Gradient Algorithm for Brushstroke Rendering

There are some variants and modifications of heuristic algorithms for brushstroke rendering. Here, we describe the algorithm from [8] with slight simplifications, which are not principal in our case. The algorithm has several parameters, eight of which are explicitly determining the rendering process, and includes several subroutines, which are described further. The main feature which distinguishes this algorithm from variants given in works [23,30] is using a constant brush width. On the one hand, this was the requirement of our robotic setup, on the other hand, this slightly simplifies the rendering process while preserving good results.

For controlling local brushstroke direction, we first compute the brushstroke gradient of the source image Im with the function $[U, V] \leftarrow \text{Gradient}(Im)$, where U and V are matrices of partial derivatives:

$$U = \frac{\partial Im}{\partial x},$$

$$V = \frac{\partial Im}{\partial y}.$$

In order to improve brushstroke coherence, the matrices U and V are additionally filtered, as described in [7]. The parameter *brushSize* defines the diameter of the brush in pixels. In the current Algorithm 1, a brushstroke is a trace of a circle with diameter *brushSize* over a curved line—the brushstroke skeleton. The skeleton consists of small straight fragments of length from 1 to *maxFrag* pixels. During the rendering process, we attempt to start a brushstroke from each pixel $Im(i, j)$ of a picture, one by one. Having taken the current pixel, first, we determine whether the error between the canvas and the source image is greater than ϵ_{start} :

$$\|Cnvs(i, j) - Im(i, j)\| \geq \epsilon_{start}.$$

Algorithm 1 Gradient-Based Brushstroke Rendering**Set:***Im*—source $N \times M$ image,*Cnvs*—canvas ($N \times M$ bitmap), N_{iter} —number of iterations, C_{over} —overlap coefficient,*brushSize*—brush diameter,*maxFrag*—maximal fragment,*minLen*—minimal stroke length,*maxLen*—maximal stroke length, ε_{start} —local error tolerance for starting a stroke, ε_a —local error tolerance for accepting a stroke fragment,*Strokes*—structure for keeping strokes, $[U, V] \leftarrow \text{Gradient}(Im);$ **for** $k \leftarrow 1..N_{iter}$ **do** **for** $i \leftarrow 1..N$ **do** **for** $j \leftarrow 1..M$ **do** **if** $err(i, j) > \varepsilon_{start}$ **then**

// Begin a new stroke

 $col \leftarrow \text{MeanColor}(Im, i, j, brushSize)$ $stroke \leftarrow \text{InitializeStroke}(i, j, col)$ $flag \leftarrow 1;$ **while** $flag$ **do**

// Find a new stroke fragment

 $r \leftarrow maxFrag$ **while** $r \geq 1$ **do** $\alpha \leftarrow \text{GetDirection}(stroke, U, V)$ $nX \leftarrow i + \text{round}(r \cos \alpha)$ $nY \leftarrow j + \text{round}(r \sin \alpha)$ **if** $\text{TestFragment}(\varepsilon_a, stroke, nX, nY, brushSize, C_{over}, Cnvs)$ **then** $stroke.add(nX, nY)$ $break$ **else** $r \leftarrow r - 1$ **end** **end**

// Test if the stroke ended

if $\text{Length}(stroke) \geq maxLen$ **or** $r = 0$ **then** $flag \leftarrow 0$ **end** **end**

// Remember the stroke

if $\text{Length}(stroke) \geq minLen$ **or** $r = 0$ **then** $Strokes.add(stroke)$ $Cnvs.DrawStroke(stroke)$ **end** **end** **end** **end****end**

Then, we use a function $\text{MeanColor}(Im, i, j, brushSize)$ to average the color of the source image over a circle with diameter *brushSize*. It is used further as a brushstroke

color col . Then, a function $InitializeStroke(i, j, col)$ creates a stroke and assigns the current values of pixel coordinates i, j to its first point. Then, we try to find the first fragment of the brushstroke. A step of length $r = maxFrag$ pixels is made perpendicularly to the local gradient, and a new fragment of the brushstroke ending in points nX, nY is tested with the function $TestFragment(\epsilon_a, stroke, nX, nY, brushSize, C_{over}, Cnvs)$. This function estimates the error between the new fragment painted on the canvas $Cnvs$ and compares the error between this newly painted region of the canvas and a corresponding fragment of a source image. C_{over} is the overlap coefficient: a number that allows the current brushstroke to overlap other brushstrokes. It defines the area of the brushstroke S_{over} which is allowed to cover previously painted pixels of the canvas: $S_{over} \leq C_{over}S_b$, where S_b is the total area of the brushstroke. If the error is greater than the allowed value ϵ_a , r is decreased by 1 pixel, and new trials to find the next fragment of the brushstroke repeat until r becomes zero or a new fragment is eventually accepted.

We determine the brushstroke ending if its length is greater than the maximal allowed value $maxLen$ or if the new fragment was not accepted. If the generated brushstroke has a length greater than the minimal length $minLen$, then a function $Strokes.add(stroke)$ adds it into the set of brushstrokes $Strokes$ and this stroke is painted on a virtual canvas with the function $Cnvs.DrawStroke(stroke)$.

As the practice shows, several painting iterations help to cover the canvas with brushstrokes more densely. N_{iter} is the total number of iterations for painting over the canvas. Moreover, parameters could vary within different iterations which were used in the algorithm for generating strokes for artistic painting with a robot [29,46].

In the current research, the following parameters were used: $brushSize = 4$, $\epsilon_{start} = 2$, $\epsilon_a = 16$, $maxFragment = 4$, $maxLen = 15$, $minLen = 2$, $C_{over} = 0.4$, $N_{iter} = 1$.

The generated brushstrokes are then saved as lines with a predefined width in the PLT file, with the structure as described before.

3.3. Brushstroke Features

In order to read a PLT file with brushstrokes, a program in MATLAB 2021 was written. This program collects data and calculates specific features of brushstrokes, distinguishing the personal style of each painter as it was captured by the program. The works [47,48] give a comprehensive description of these features, and here we briefly review them.

1. *Length*. Brushstroke length along the skeleton of the brushstroke. For each brushstroke, consisting of N pixels with coordinates $x_k, y_k, k = 1 \dots N$, the length L_b is calculated as the sum of the distances between neighboring points:

$$L_b = \sum_{k=1}^{N-1} \|(x_{k+1} - x_k, y_{k+1} - y_k)^T\|.$$

2. *Straightness*. Brushstroke straightness is defined as the Pearson correlation coefficient (PCC) between the horizontal and vertical coordinates of pixels located on the brushstroke skeleton. If the skeleton is a perfectly straight line, the correlation coefficient will be equal to one; if the skeleton is curved, the absolute value of the coefficient will be less than one. Suppose the brushstroke contains N pixels, with coordinates $x_i, y_i, i = 1 \dots N$, the straightness S_t is defined as:

$$S_t = \frac{N \sum_{i=1}^N x_i y_i - \sum_{i=1}^N x_i \sum_{i=1}^N y_i}{\sqrt{(N \sum_{i=1}^N x_i^2 - (\sum_{i=1}^N x_i)^2)(N \sum_{i=1}^N y_i^2 - (\sum_{i=1}^N y_i)^2)}}.$$

3. *Orientation*. To obtain the brushstroke orientation, we use an alternative to the definition in [48]. For each brushstroke with x_i, y_i coordinates set, a linear least squares fit is found using the *polyfit* function in MATLAB. The brushstroke orientation is defined as the slope of the approximating linear polynomial, i.e., as $\alpha = \arctan(k)$, where k is the first coefficient of the linear polynomial.

4. *Number of brushstrokes in the neighborhood (NBS-NB)*. A brushstroke j is a neighbor to a brushstroke i if the distance between the centers of these brushstrokes does not exceed the threshold value s : $|x_{c_i} - x_{c_j}| < s$ and $|y_{c_i} - y_{c_j}| < s$, where the threshold value is set to 200, as in [48]. NBS-NB is the total number of strokes that are neighbors of i .
5. *Number of brushstrokes with similar orientations in the neighborhood (NBS-SO)*. A brushstroke j has a similar orientation as i if the difference between their orientations is below a threshold value. The threshold value is set to 0.35, as in [48].
6. *Orientation standard deviation in the neighborhood (OSD-NB)*. For any brushstroke i , we compute the orientation standard deviation for all brushstrokes in the neighborhood as:

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (\alpha_i - \bar{\alpha})^2},$$

where α_i is the orientation of the i -th stroke in the neighborhood, N is the number of brushstrokes in the neighborhood, and $\bar{\alpha}$ is the mean orientation of all neighboring brushstrokes.

Features such as *width*, *elongatedness* (the ratio between the length and the width), *broadness*, and *homogeneity* are less informative in the current study and are not taken into account.

3.4. Test Set

For tests, eight images have been considered. They include seven paintings by Vincent van Gogh, Georges Seurat, and Isaac Ilyich Levitan, and one standard test image, “Lenna”. The given three artists were chosen because they all have unique painting styles that are visually easy to distinguish. A popular and well-recognized standard image, “Lenna”, is needed for rendering in different styles. These images are given in Figure 2. Additionally, some information on the source image resolution is provided. As one can see, these images are similar in size.

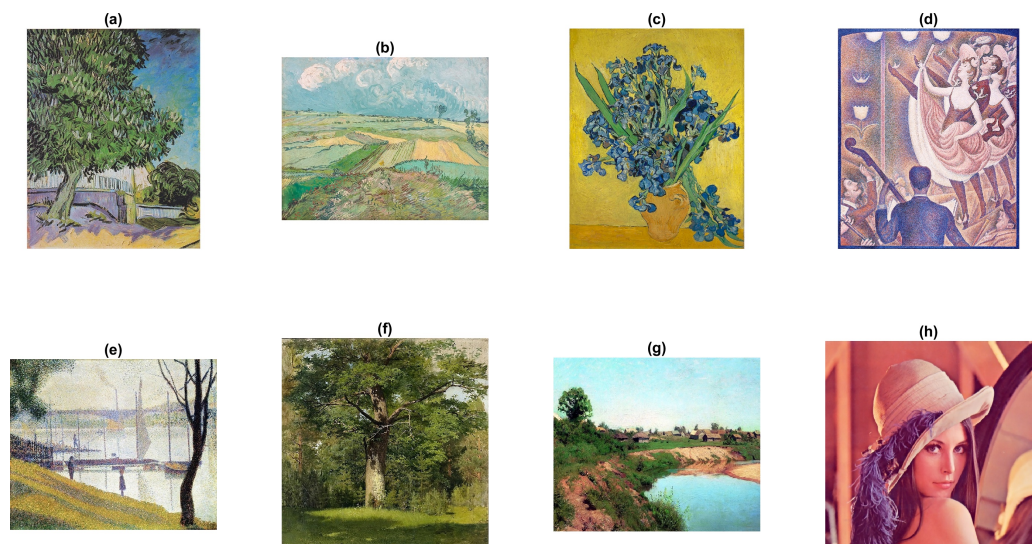


Figure 2. The test set: (a) Vincent van Gogh, “Chestnut Tree in Blossom”, 1890, 651 × 518 pixels; (b) Vincent van Gogh, “Wheat Fields at Auvers under Clouded Sky”, 1890, 516 × 654 pixels; (c) Vincent van Gogh, “Vase with Irises Against a Yellow Background”, 1890, 646 × 515 pixels; (d) Georges Seurat, “The Bridge at Courbevoie”, 1886–1887, 515 × 623; (e) Georges Seurat, “Le Chahut”, 1889–1890, 622 × 515 pixels; (f) Isaac Ilyich Levitan, “Oak”, 1880, 527 × 517 pixels; (g) Isaac Ilyich Levitan, “Village on the River Bank”, 1883, 515 × 616 pixels; (h) “Lenna”, 512 × 512 pixels.

3.5. Design of the Experiment

We propose the following experimental design. Considering two programs, one using the NST algorithm and another using the heuristic algorithm, we run them on various images from a test set and investigate the parameters of brushstrokes generated by these programs. For a numerical assessment of the individual manner of drawing brushstrokes, we consider six brushstroke features, adopted from work [48]: length, straightness, orientation, orientation standard deviation in the neighborhood (OSD-NB), number of brushstrokes in the neighborhood (NBS-NB), and number of brushstrokes with similar orientations in the neighborhood (NBS-SO). Brushstroke color is out of the scope of this research, partially because NST and heuristic approaches give totally different results as seen by an eye, which is because color rendering is a separate difficult problem. To capture and estimate differences and similarities between parameters, we first build parameter histograms for each image, and then we perform a correlation analysis of these histograms. We perform the following tests:

1. *Similar artist test.* Given nine paintings in three different styles, three artworks in each one, we render them to themselves with the NST algorithm and compare brushstroke parameter distributions. This test aims to learn whether the implementation of the style transfer algorithm from [33] is capable of adapting its results to an individual brushstroke rendering manner.
2. *NST vs. heuristic test.* Given three paintings in different styles, we render them with the NST and heuristic algorithms and compare the brushstroke parameter distributions. This test aims to quantify similarities and differences between the two considered approaches.
3. *Similar image test.* Given one standard image, we render it with the NST program in three different styles and with the heuristic program and then compare brushstroke parameter distributions. This test aims to determine whether differences between these algorithms are substantiated mostly by their design or mostly by the content image.
4. *Real painting test.* Given one image and one parameter which is not radically different in the results given by the two considered algorithms, we compare its distribution with that of the real image investigated in [48]. This test aims to determine which approach gives results closer to the real painting, or whether both are closer to each other than to the real painting.

So, while the results have been obtained on a relatively small sample set, we can answer the questions of whether the obsolete (heuristic) and the state-of-the-art (NST) approaches are comparable in their ability to render brushstrokes, or whether one is better, and whether it is possible, at least in some cases, to render brushstrokes in the manner of a particular artist well enough, as the proposed tests allow.

4. Results

This section reports the results of brushstroke rendering for both programs and gives their comparison in terms of the previously introduced metrics.

4.1. Examples of Brushstroke Rendering

Images from the test set were rendered by both programs. In order to estimate the quality of both programs visually, we show two examples. The first example concerns the work “Chestnut Tree in Blossom” by Vincent van Gogh. Figure 3 illustrates the original image, the results of two renderings, and also a brushstroke map (brushstroke skeletons on a distinctively visible background) which helps to see the structure of brushstrokes for the rendering by the gradient algorithm. This image was used in the first and the second tests, see the further text of the paper.

The second example concerns the image “Lenna” (Figure 4). Using the neural style transfer algorithm, this image was stylized with artworks of the three considered artists: Van Gogh’s “Vase with Irises Against a Yellow Background”, Seurat’s “The Bridge at Courbevoie”, and Levitan’s “Oak”. For comparison, the image “Lenna” was also ren-

dered with the gradient algorithm. These four renders are then used in the first, third, and fourth tests.

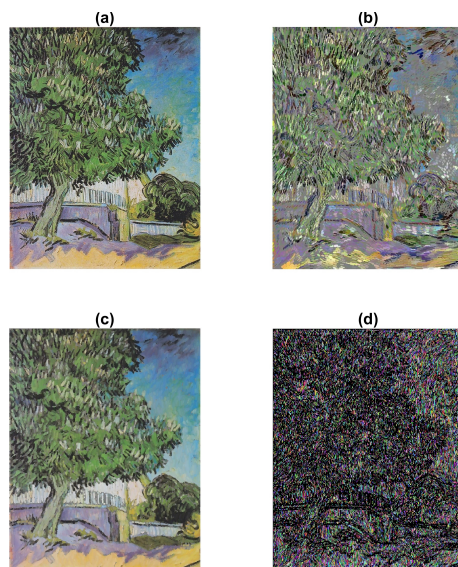


Figure 3. Example of rendering the work “Chestnut Tree in Blossom” by Van Gogh: (a) an original image, (b) a render with the NST algorithm (painting to itself), (c) a render with the gradient algorithm, (d) brushstrokes map for the result of the gradient algorithm.

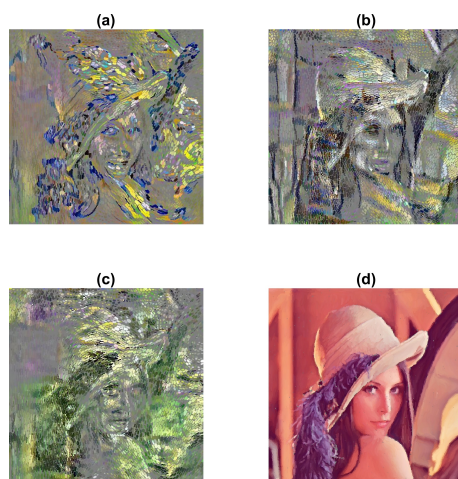


Figure 4. Example of rendering the image “Lenna”: (a) a render with the NST algorithm, stylized with Van Gogh’s “Irises...”, (b) a render with the NST algorithm, stylized with Seurat’s “Bridge...”, (c) a render with the NST algorithm, stylized with Levitan’s “Oak”, (d) a render with the gradient algorithm.

4.2. Painting-to-Itself by the NST Algorithm Test

The first test is aimed at determining whether there is distinguishable evidence of affecting the style image on the brushstroke features when using the NST algorithm. This test involved the entire test set. The images “Chestnut...” and “Wheat...” by Van Gogh, “Bridge...” and “Le Chahut” by Seurat, and “Oak” and “Village...” by Levitan were styled by themselves with the NST algorithm. The image Lenna was stylized with Van Gogh’s “Irises...”, Seurat’s “Bridge...”, and Levitan’s “Oak”, as shown in Figure 4a–c. Examples of parameter histograms and correlation tables are given in Figures 5 and 6. Figure 5 gives histograms of brushstroke lengths within nine rendered images, and Figure 6 shows correlation coefficients between histograms. One can see that there is no obvious dependency between the artist and the length distribution.

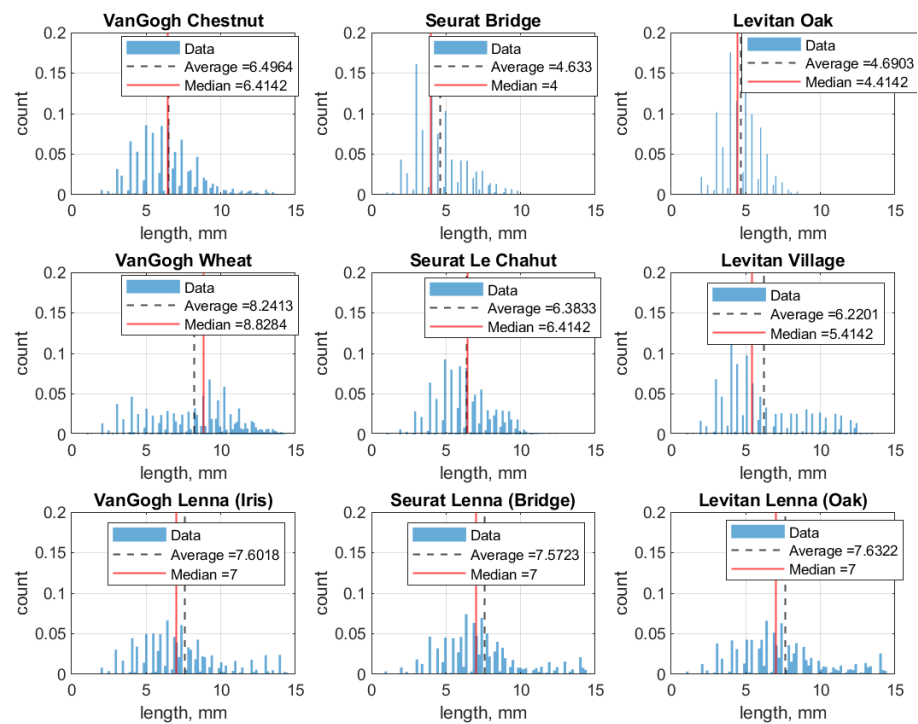


Figure 5. Brushstroke length for three paintings by three different painters: Levitan, Seurat, Van Gogh.

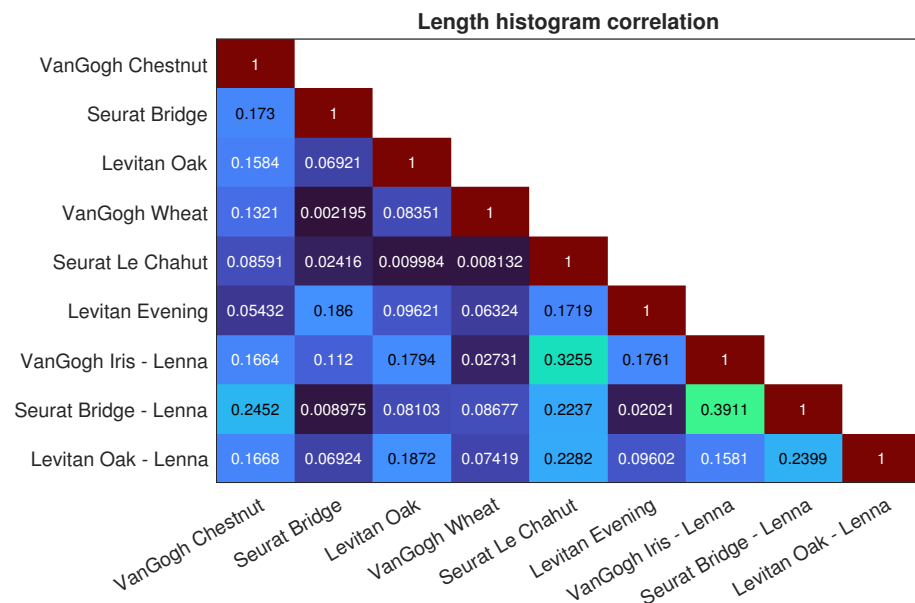


Figure 6. Absolute values of correlations for brushstroke lengths distributions from Figure 5.

The other histograms and correlation tables are presented in Appendix A. In order to summarize the results of the first test, we present the most significant absolute values of correlations $|\rho|$ in Table 1. No relation between style and absolute values of correlations can be found. For some parameters, such as length and orientation, the source image matters. Nevertheless, the main hypothesis, that the style image affects the brushstroke features, is disproved.

Table 1. Maximal absolute values of correlations of the first test.

Feature	Images	$ \rho $
Length	Seurat Lenna (Bridge), Van Gogh Lenna (Iris)	0.3911
	Seurat Le Chahut, Van Gogh Lenna (Iris)	0.3255
	Van Gogh Chestnut, Seurat Lenna (Bridge)	0.2452
Straightness	Levitan Lenna (Oak), Seurat Lenna (Bridge)	0.99564
	Seurat Le Chahut, Seurat Lenna (Bridge)	0.98714
	Levitan Lenna (Oak), Seurat Le Chahut	0.987
Orientation	Levitan Lenna (Oak), Seurat Lenna (Bridge)	0.93126
	Van Gogh Lenna (Iris), Seurat Lenna (Bridge)	0.90229
	Van Gogh Chestnut, Seurat Le Chahut	0.8997
NBS-NB	Levitan Village, Seurat Bridge	0.94412
	Levitan Lenna (Oak), Seurat Lenna (Bridge)	0.9184
	Seurat Lenna (Bridge), Levitan Lenna (Oak)	0.9082
NBS-SO	Seurat Le Chahut, Seurat Bridge	0.90952
	Levitan Lenna (Oak), Seurat Bridge	0.86419
	Levitan Oak, Van Gogh Lenna (Iris)	0.8639
OSD-NB	Levitan Village, Seurat Le Chahut	0.83429
	Levitan Oak, Van Gogh Chestnut	0.7028
	Seurat Lenna (Bridge), Van Gogh Lenna (Iris)	0.6884

4.3. Painting-to-Itself by NST and Gradient Algorithms Test

In this test, a comparison was made between two brushstroke generation algorithms: the neural style transfer and the heuristic-based algorithms. This test aimed to quantify similarities and differences between rendering results of the two investigated algorithms.

According to the experimental design, images of different styles are needed in this experiment. So, three paintings by three different artists were selected: “Chestnut...” by Vincent van Gogh, “Bridge...” by Georges Seurat, and “Oak” by Isaac Levitan. Figures 7–9 show histograms of the most informative features: length, straightness, and OSD-NB, for the two algorithms under study, where the upper plots correspond to the neural style transfer, and the lower plots correspond to the gradient algorithm. Figures 10–12 show the correlations of the obtained histograms for each selected feature. Histograms and their correlations for other features (orientations, NBS-NB) are presented in Appendix B.

The length histograms in Figure 7 show that the NST algorithm tends to generate averagely longer strokes while the gradient algorithm prefers shorter strokes. In Figure 7, one also can see how similar the histograms from the same algorithm are to each other, and there is almost no difference between the histograms of different patterns within the same algorithm. This is also confirmed by the correlation diagram in Figure 10. This is especially notable for the gradient algorithm, which generates approximately the same variability in brushstroke lengths.

From Figure 8 it is seen that the gradient algorithm generates much more straight strokes.

Histograms for the orientation standard deviation, as one can see from Figure 9, are sufficiently different for the two investigated algorithms. Larger and more diverse values of the OSD-NB feature are given by the NST algorithm. The more coherent brushstrokes given by our implementation of the gradient algorithm are partially due to the nature of the algorithm itself and partially due to high noise suppression in the gradient obtained by filtering, as described in [7].

Table 2 summarizes the results of the correlation analysis.

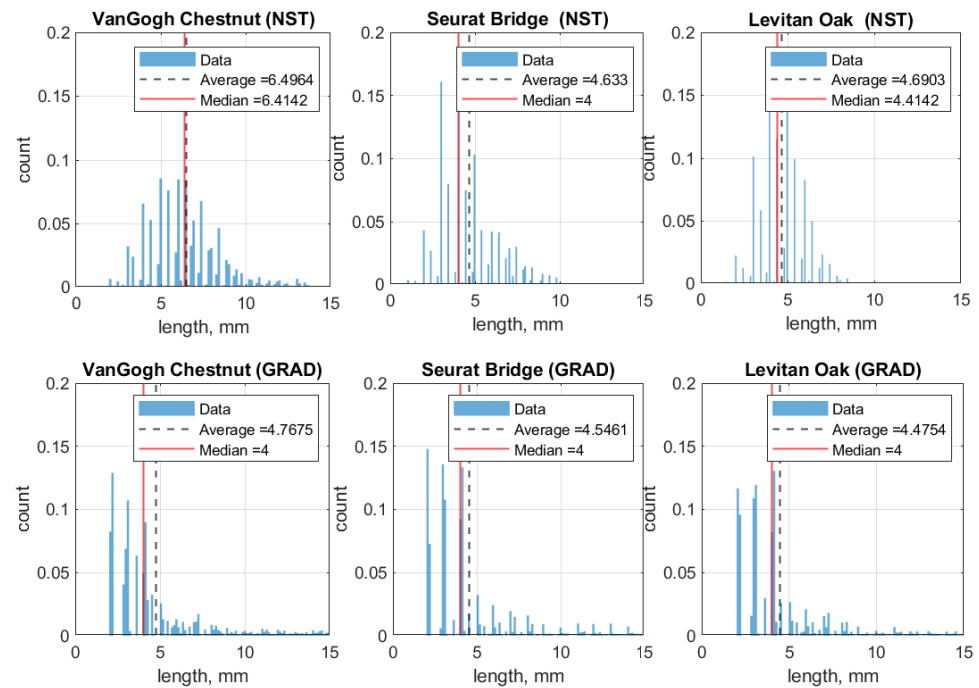


Figure 7. Brushstroke length for two paintings by three different painters: Levitan, Seurat, Van Gogh.

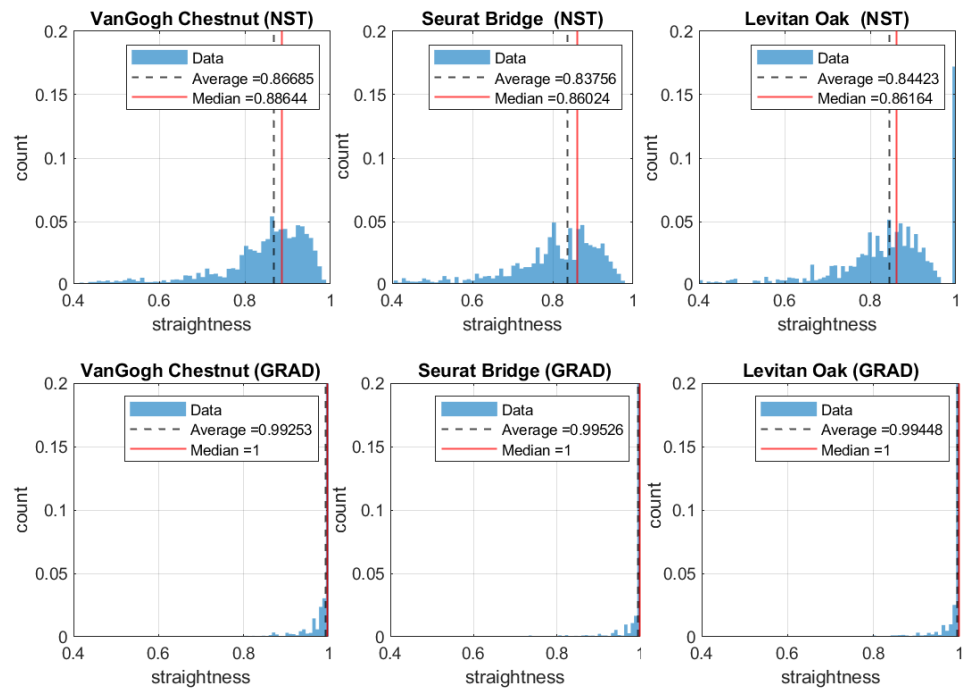


Figure 8. Brushstroke straightness for three paintings by different painters: Levitan, Seurat, Van Gogh.

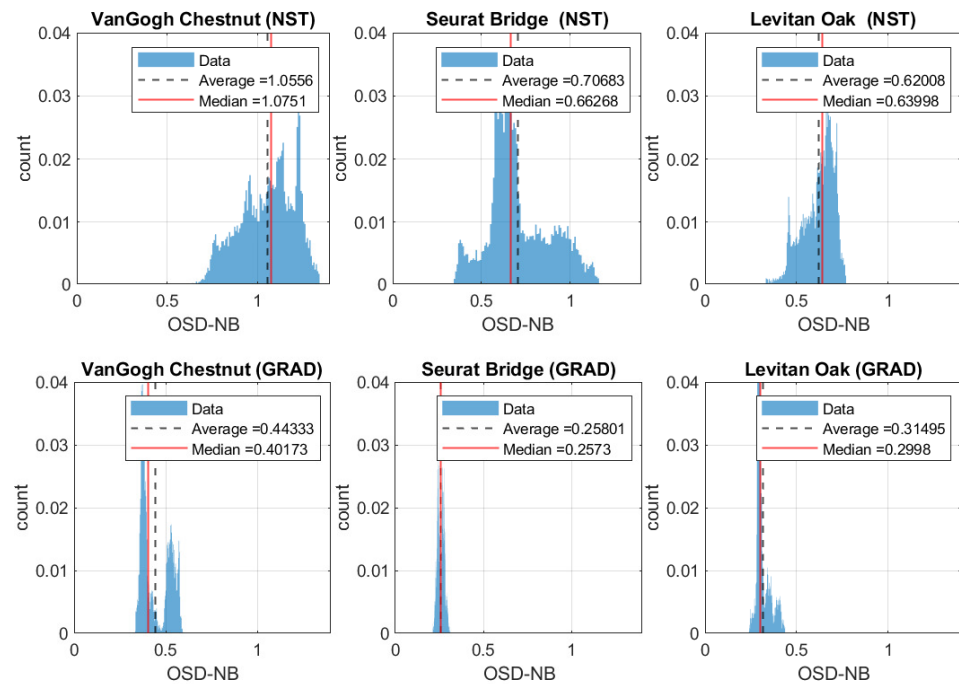


Figure 9. Brushstroke OSD-NB for two paintings by three different painters: Levitan, Seurat, Van Gogh.

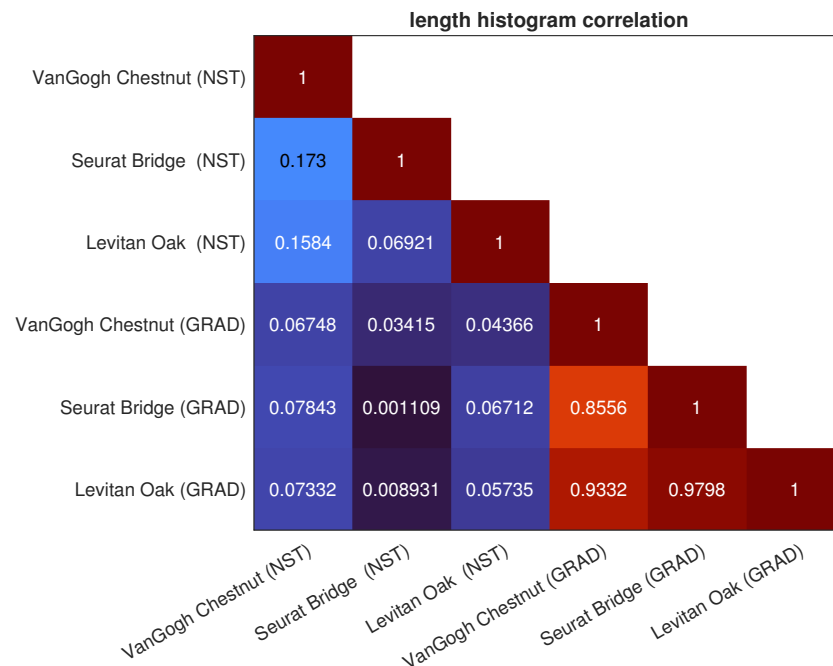


Figure 10. Brushstroke length for three paintings from different painters: Levitan, Seurat, Van Gogh.

From Table 2 one can see that there is a strong correlation between the length, straightness, and orientation histograms generated with the gradient algorithm, but the other parameter histograms are correlated more randomly. This means that the rendering results from the gradient algorithm are closer to each other in these features. Furthermore, this means that there is a notable difference between brushstrokes generated by these two algorithms, at least, in the investigated features.

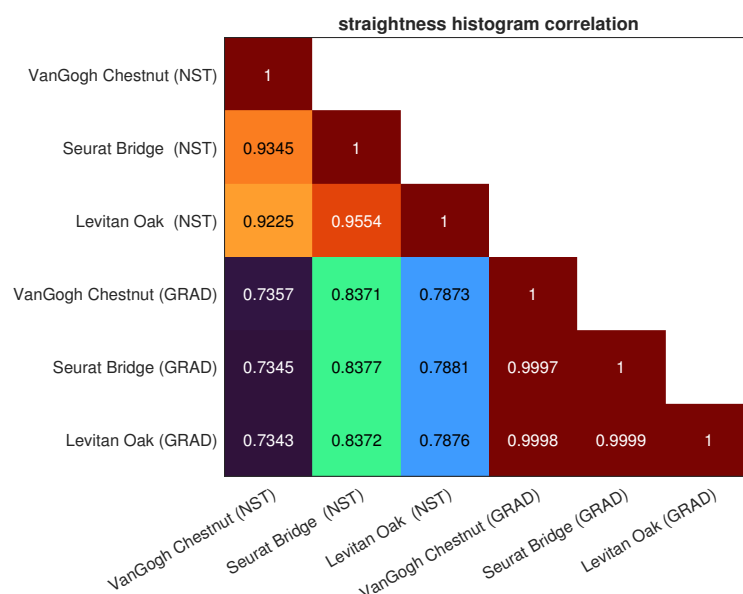


Figure 11. Brushstroke straightness for two paintings by three different painters: Levitan, Seurat, Van Gogh.

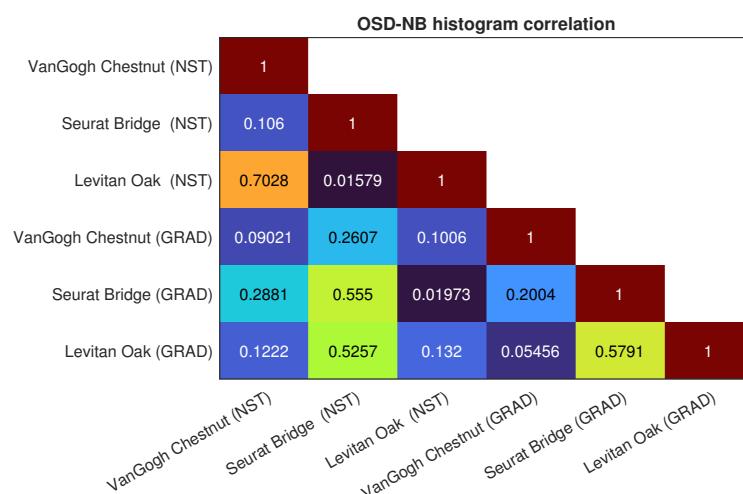


Figure 12. Brushstroke OSD-NB for three paintings from different painters: Levitan, Seurat, Van Gogh.

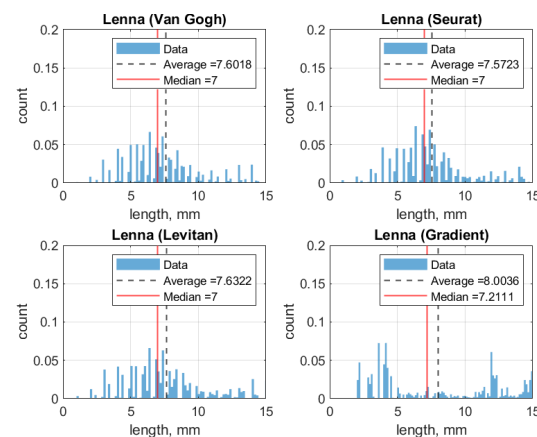
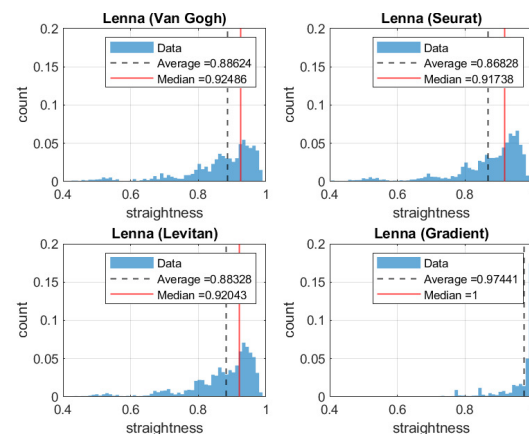
4.4. Similar Image Test

In this test, an image, “Lenna”, was rendered with the NST algorithm using three style images and with the gradient algorithm, as shown in Figure 4. This test aimed to find out whether there could be a notable difference between brushstroke features rendered with different styles using the NST algorithm, and the gradient algorithm was used as a referent rendering solution. The test determines whether the difference between two results from the NST algorithm is more significant than the difference between results from the NST and the gradient algorithms.

After running the test, we found the most significant differences in the following features, given in Figures 13–15: length, straightness, and orientation.

Table 2. Maximal absolute values of correlations of the second test.

Feature	Images	$ \rho $
Length	Levitan Oak (GRAD), Seurat Bridge (GRAD)	0.97981
	Van Gogh Chestnut (GRAD), Levitan Oak (GRAD)	0.93316
	Seurat Bridge (GRAD), Van Gogh Chestnut (GRAD)	0.8556
Straightness	Levitan Oak (GRAD), Seurat Bridge (GRAD)	0.99993
	Van Gogh Chestnut (GRAD), Levitan Oak (GRAD)	0.99981
	Seurat Bridge (GRAD), Van Gogh Chestnut (GRAD)	0.9997
Orientation	Levitan Oak (GRAD), Seurat Bridge (GRAD)	0.98387
	Van Gogh Chestnut (GRAD), Levitan Oak (GRAD)	0.95169
	Van Gogh Chestnut (GRAD), Seurat Bridge (GRAD)	0.9377
NBS-NB	Van Gogh Chestnut (GRAD), Seurat Bridge (NST)	0.92878
	Levitan Oak (NST), Seurat Bridge (NST)	0.90668
	Van Gogh Chestnut (GRAD), Levitan Oak (NST)	0.8839
NBS-SO	Van Gogh Chestnut (GRAD), Van Gogh Chestnut (NST)	0.8371
	Levitan Oak (NST), Van Gogh Chestnut (GRAD)	0.82775
	Levitan Oak (NST), Seurat Bridge (NST)	0.82
OSD-NB	Levitan Oak (NST), Van Gogh Chestnut (NST)	0.70276
	Levitan Oak (GRAD), Seurat Bridge (GRAD)	0.5791
	Seurat Bridge (NST), Seurat Bridge (GRAD)	0.55502

**Figure 13.** Brushstroke length histograms for rendering results for “Lenna” image.**Figure 14.** Brushstroke straightness histograms for rendering results for “Lenna” image.

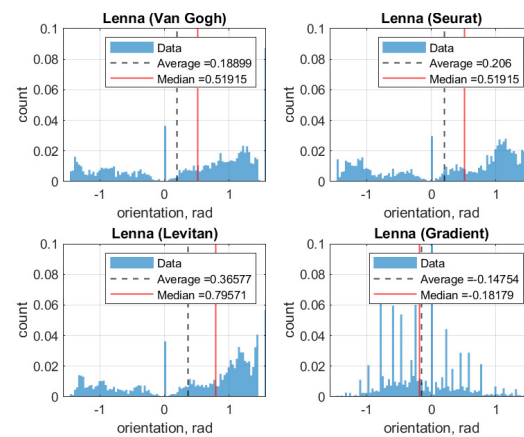


Figure 15. Brushstroke orientation histograms for rendering results for “Lenna” image.

The most significant correlations are summarized in Table 3.

Table 3. Maximal absolute values of correlations of the third test.

Feature	Images	$ \rho $
Length	Lenna (Seurat), Lenna (Van Gogh)	0.3911
	Lenna (Levitan), Lenna (Seurat)	0.2399
	Lenna (Van Gogh), Lenna (Seurat)	0.1856
Straightness	Lenna (Levitan), Lenna (Seurat)	0.99564
	Lenna (Van Gogh), Lenna (Seurat)	0.96133
	Lenna (Levitan), Lenna (Van Gogh)	0.9591
Orientation	Lenna (Levitan), Lenna (Seurat)	0.93126
	Lenna (Van Gogh), Lenna (Seurat)	0.90229
	Lenna (Levitan), Lenna (Van Gogh)	0.8639
NBS-NB	Lenna (Levitan), Lenna (Seurat)	0.91844
	Lenna (Van Gogh), Lenna (Seurat)	0.90822
	Lenna (Levitan), Lenna (Van Gogh)	0.898
NBS-SO	Lenna (Levitan), Lenna (Van Gogh)	0.84841
	Lenna (Seurat), Lenna (Levitan)	0.83553
	Lenna (Van Gogh), Lenna (Seurat)	0.8088
OSD-NB	Lenna (Gradient), Lenna (Seurat)	0.70927
	Lenna (Van Gogh), Lenna (Seurat)	0.68831
	Lenna (Gradient), Lenna (Van Gogh)	0.6879

From Table 3 it is clear that, in all tests the correlations between the NST results are more significant, which points out significant mutual similarities between the brushstroke structures generated with the NST algorithm. So, there is no determined evidence that different style images lead to a significant difference in rendering results. Correlations for the Lenna images are presented in Appendix C.

4.5. Real Painting Test

In this test, the histogram of the parameter NBS-NB, available from the literature [48], is compared to histograms obtained with the NST and the gradient algorithms. This test aims to find out whether the algorithms match an observation from the real painting.

From Figure 16, one can see that the NBS-NB histograms for the synthetic images are significantly different from the NBS-NB histogram of the real painting. Moreover, histograms for both of the synthetic images are more similar. Figure 17 confirms this observation numerically: the correlation coefficient between histograms for both algorithms is notably higher in comparison with the correlation coefficients between the real and

synthetic images. This means that the algorithms gave results more different from the real painting than from each other.

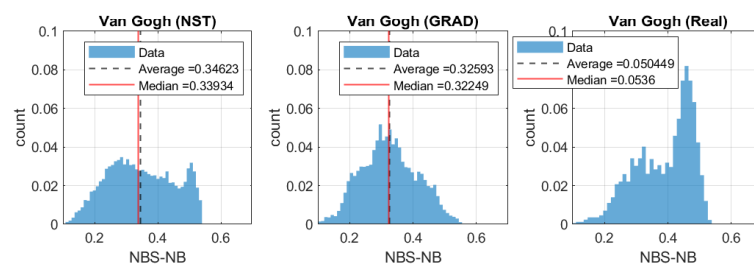


Figure 16. Brushstroke NBS-NB histograms for rendering results for “Chestnut...” image and its value estimated from the real painting.

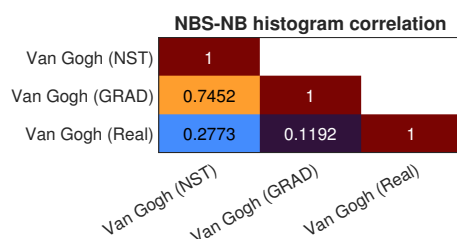


Figure 17. Absolute values for the correlation coefficients between NBS-NB histograms for rendering results for “Chestnut...” image and its value estimated from the real painting.

5. Discussion

While in the current study we compare only two algorithms: the gradient-based and the neural style transfer algorithms, there is a number of other approaches to stroke-based rendering. One approach that was out of the scope of this research is optimization-based rendering. As an optimization method, the genetic algorithm (GA) is often proposed as a reliable and parallelizable global search method [49–51]. Recently, many modifications of GAs have been proposed, for example, the Cartesian genetic programming, which uses a more economical implementation of paint rendering and can be easily accelerated on GPU [49]. It was shown that GAs outperform the stochastic descent method [52], but may be less efficient than other simple algorithms in the artistic rendering problem. For example, in [5] it was shown that the simulated annealing algorithm could obtain better rendering results in terms of mean squared error when compared to the HillClimber and plant propagation evolutionary algorithms.

In most research, the target function is complex, being designed to minimize the mean squared error between the source and rendered images while satisfying a certain aesthetics criterion, e.g., closeness of fit between the actual image gradient distribution and the normal distribution [53], using a minimal number of strokes and occupying a maximum area of canvas [51]. Since the design of optimization-based methods allows an arbitrary construction of the target function, other criteria for the “goodness” of the rendering results can also be easily incorporated into the optimization-based rendering algorithm. In the current study, the criteria for “goodness” were outlined explicitly in terms of histogram correlations, but modern machine learning techniques may provide these criteria implicitly by constructing a classifier that would be capable of distinguishing a particular artistic style without the need to interpret it by a human [54].

6. Conclusions and Future Research

In this paper, a novel method for quantifying brushstroke rendering results, based on correlation analysis of feature histograms, is proposed. This is the first time when the correlation of histograms is used for this purpose. This method ignores the horizontal and vertical scale of histograms and therefore is applicable to rendering results of images with different sizes and with a different number of brushstrokes.

Using this method, extensive research has been performed for comparing rendering results by the neural style transfer (NST) algorithm and the heuristic gradient algorithm. The research involved distributions of six features of brushstrokes, including length, straightness, orientation, number of neighboring brushstrokes (NBS-NB), number of brushstrokes with similar orientations in the neighborhood (NBS-SO), and orientation standard deviation in the neighborhood (OSD-NB). Using the test set of eight test images, four various tests have been carried out: painting-to-itself by the NST algorithm (test 1), painting-to-itself by the NST and gradient algorithms (test 2), rendering the similar image (test 3), and comparison with the real image (test 4). The results of all of the tests point out a notable difference between paintings rendered with the NST algorithm and the gradient algorithm. The tests also discovered the insufficient diversity between paintings rendered with different style images for the NST algorithm, and, predictably, even more similarity between paintings rendered with the gradient algorithm. The main contributions of this paper are:

- A novel method for quantifying brushstroke rendering results based on correlation analysis of feature histograms is proposed.
- A comparison of the heuristic gradient-based algorithm with the NST stroke-based algorithm using the proposed method is carried out.
- As a result, the paper offers valuable insights into the limitations of both current style transfer and heuristic techniques and highlights the need for further research to improve their effectiveness.

The advantage of our method for experimental evaluation of non-photorealistic rendering algorithms is the ability to comprehensively assess how different algorithms can mimic such a local style pattern as a brushstroke. Using correlation histograms instead of the histograms themselves allows for comparing images with different numbers of brushstrokes, resolution, and other parameters without any special normalization, and effectively distinguishes between brushstroke patterns in paintings obtained by the compared algorithms.

The limitations of this method are the difficulty in comparing more than two methods, and the need to perform many tests on different test images, since a high histogram correlation coefficient may be obtained in a single experiment by chance.

The obtained results are controversial: while both the NST and the heuristic approaches gave distinctively different results, they both failed to render particular artistic styles and did not succeed in producing results close to the real painting. Probably the main reason for this result is that proper style learning needs some additional procedures, for instance, extracting brushstrokes from the source image and the corresponding processing procedures. Brushstroke-map extraction is a challenging task itself. Simple deterministic approaches, such as the ones described in [47,48], can distinguish only prominent top-layer brushstrokes, comprising approximately no more than 10% of all brushstrokes in the painting. A possible step towards proper style learning was recently proposed in [30], where a neural network DStroke was introduced, capable of extracting a dense brushstroke map from a painting. In addition to brushstroke extraction, a different loss function could be introduced, taking into account all the features investigated in the current research. The difficulty of finding a proper loss function is well known in machine learning, so a number of alternative loss function designs should be tested. In this regard, an idea to use histogram correlation may be also fruitful for this task.

An alternative approach could be based on generating a machine-derived heuristic. This heuristic may be obtained via optimization methods, e.g., genetic programming, particle swarm optimization, etc. The objective function for this optimization procedure would include the error between the brushstroke features of the style image and the rendering result. This approach may be even more efficient if it would include some information on the intermediate steps of a painting by an artist, or even a record of a full painting process from scratch, since the heuristic is aimed at simulating an artist's actions during painting. To our knowledge, no open database exists which would provide such data. One possible way to collect the required records is involving volunteer painters and

recording all their actions via some digital data collecting tools. While this is not technically difficult in digital painting, it becomes a challenge when using real media such as oil paints. So, a special experimental setup and experiment design would need to be developed.

Author Contributions: Conceptualization, A.K. and D.B.; data curation, E.K. (Evgenii Katser); formal analysis, E.K. (Evgenii Katser) and S.L.; funding acquisition, A.K.; investigation, T.S. and E.K. (Evgenii Katser); methodology, A.K.; project administration, A.K. and D.B.; resources, A.K.; software, E.K. (Ekaterina Kopets), T.S. and E.K. (Evgenii Katser); supervision, D.B.; validation, E.K. (Ekaterina Kopets) and S.L.; visualization, E.K. (Ekaterina Kopets), T.S. and S.L.; writing—original draft, A.K., E.K. (Ekaterina Kopets) and D.B.; writing—review and editing, S.L. and D.B. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported by a grant of the Russian Science Foundation, RSF grant number 22-79-00171.

Data Availability Statement: The source images, rendered PLT files, and codes for generating all the images in the paper are free to access via GitHub <https://github.com/aikarimov/Style-Transfer> (accessed on 5 April 2023).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

NST	Neural style transfer
NBS-NB	Number of neighboring brushstrokes
NBS-SO	Number of brushstrokes with similar orientations in the neighborhood
OSD-NB	Orientation standard deviation in the neighborhood

Appendix A. Results for Test 1

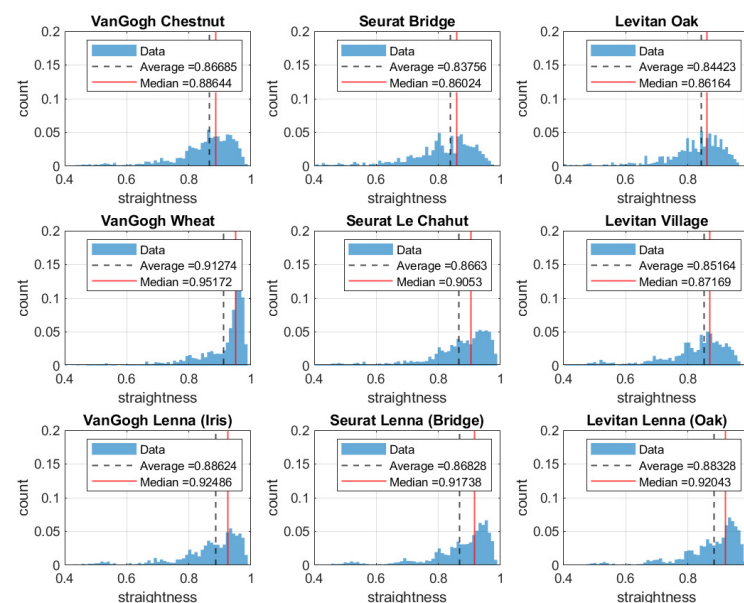


Figure A1. Brushstroke straightness histograms for images in test 1.

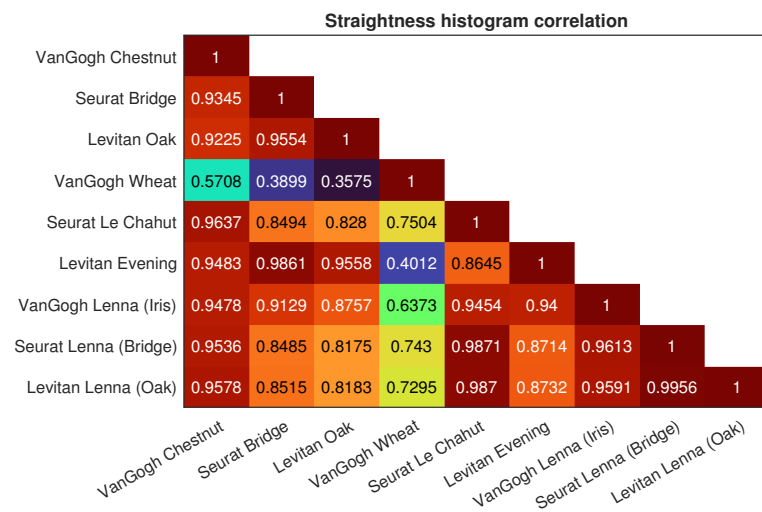


Figure A2. Brushstroke straightness histogram correlations for images in test 1.

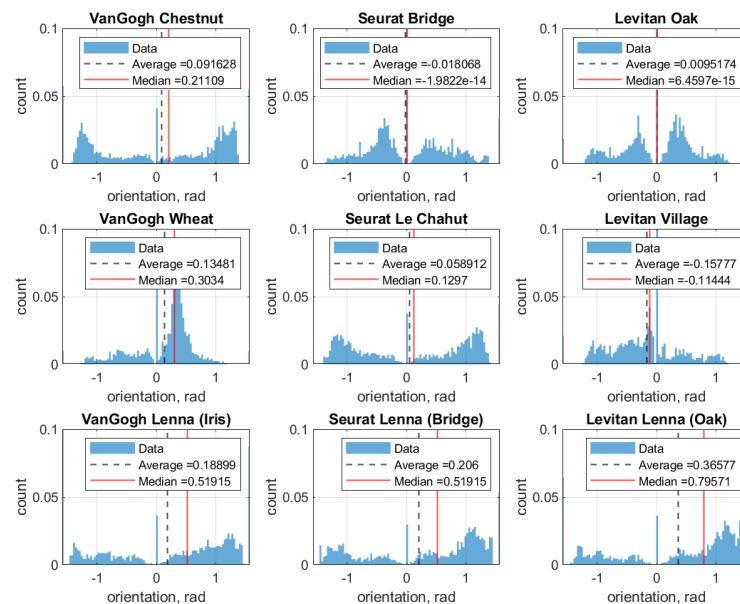


Figure A3. Brushstroke orientation histograms for images in test 1.

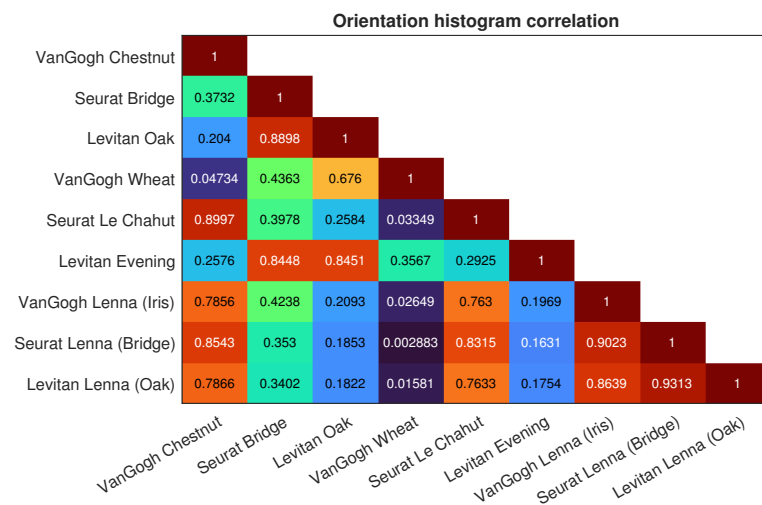


Figure A4. Brushstroke orientation histogram correlations for images in test 1.

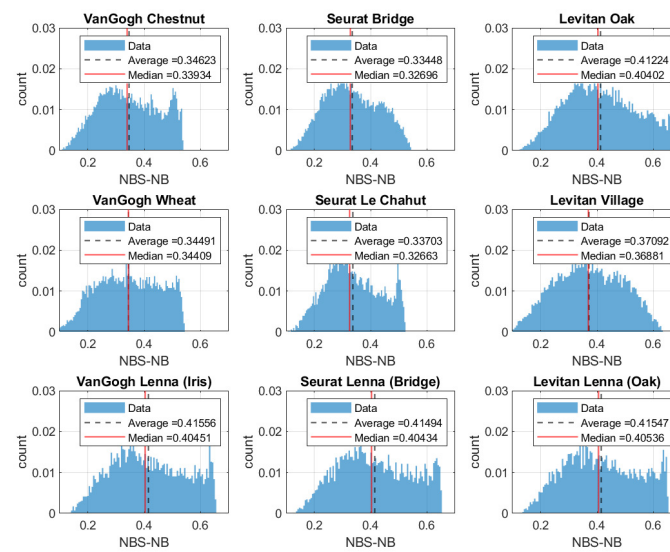


Figure A5. NBS-NB histograms for images in test 1.

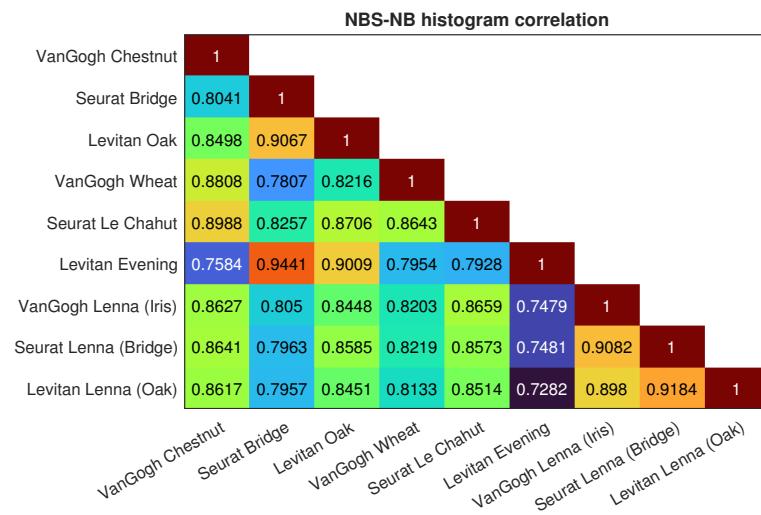


Figure A6. NBS-NB histogram correlations for images in test 1.

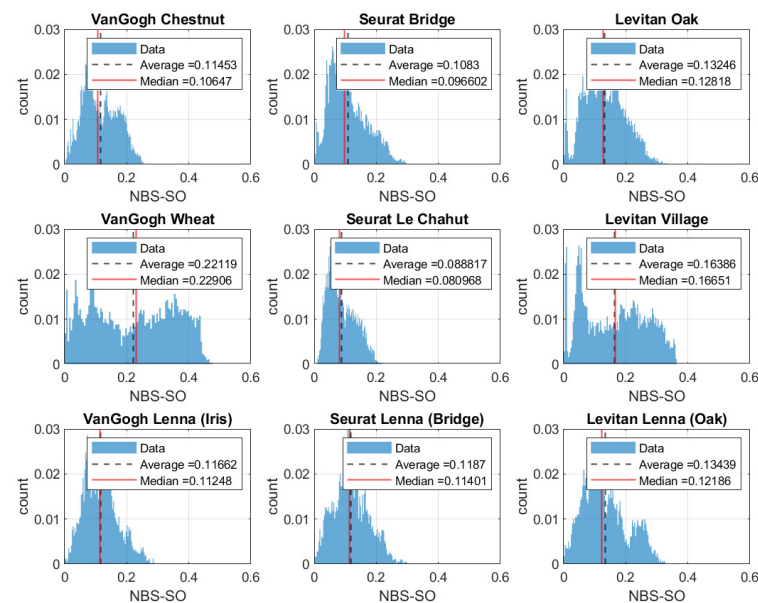


Figure A7. NBS-SO histograms for images in test 1.

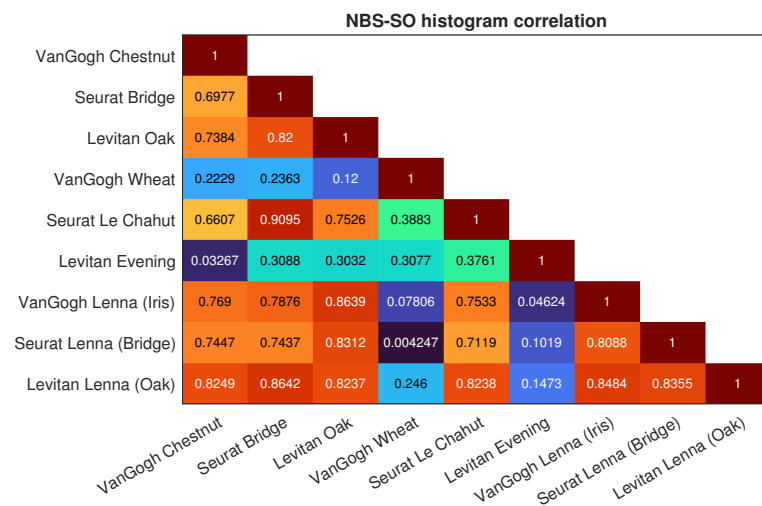


Figure A8. NBS-SO histogram correlations for images in test 1.

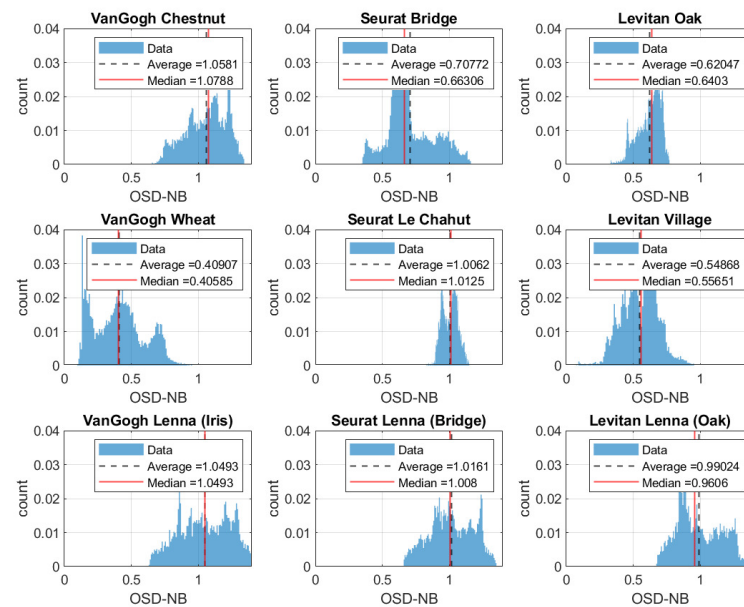


Figure A9. OSD-NB histograms for images in test 1.

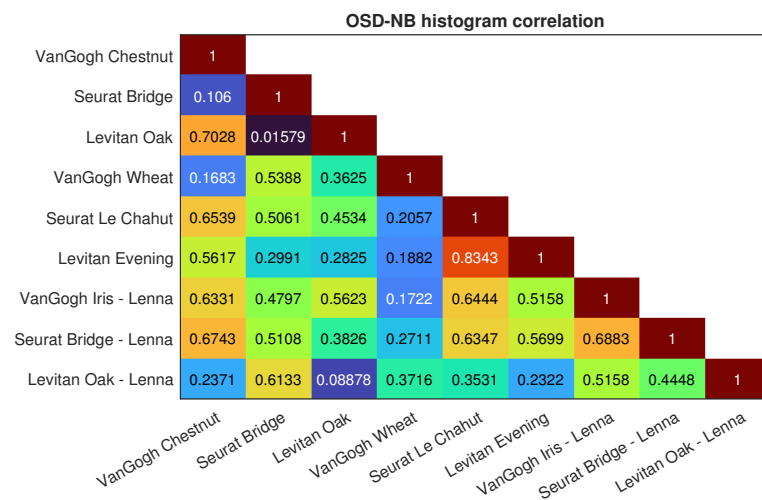


Figure A10. OSD-NB histogram correlations for images in test 1.

Appendix B. Results for Test 2

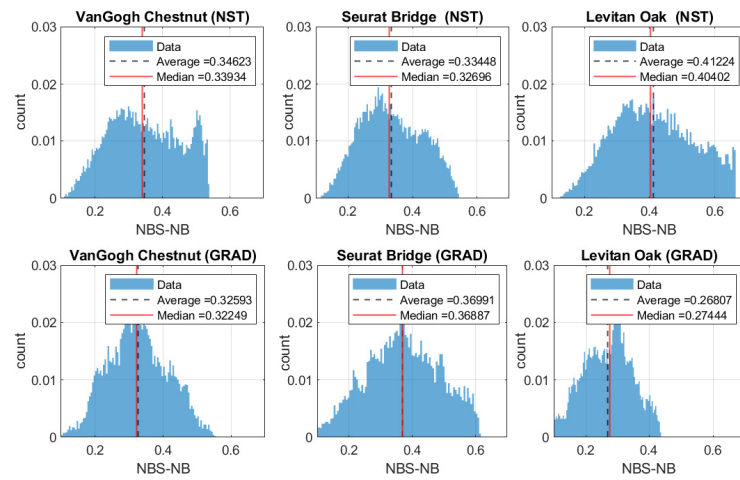


Figure A11. NBS-NB for three paintings of different painters in test 2.

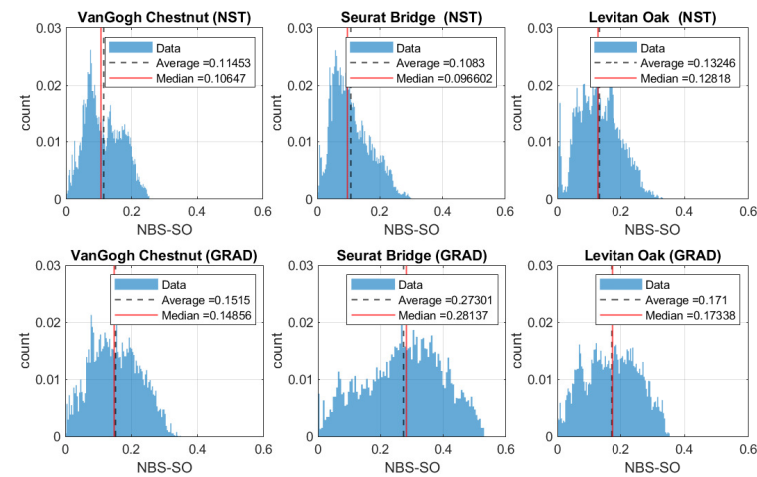


Figure A12. NBS-SO for three paintings of different painters in test 2.

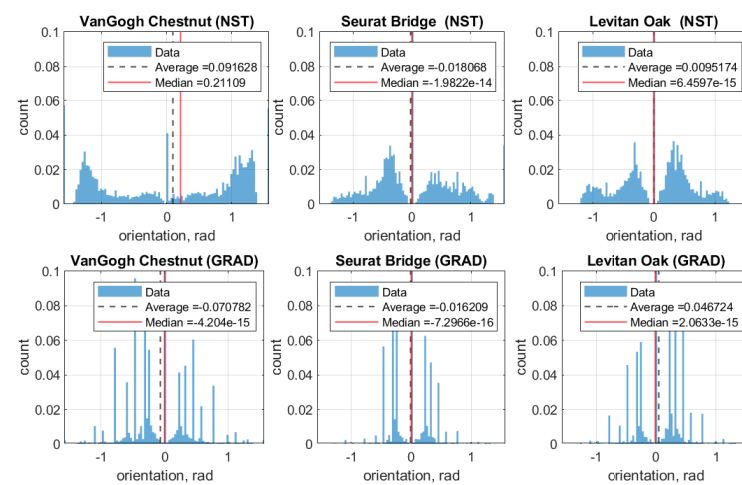


Figure A13. Brushstroke orientation for three paintings of different painters in test 2.

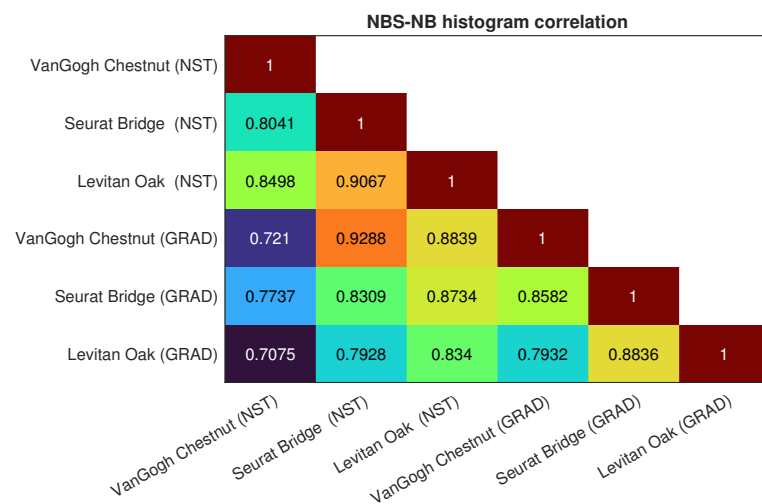


Figure A14. NBS-NB histogram correlation for three paintings of different painters in test 2.

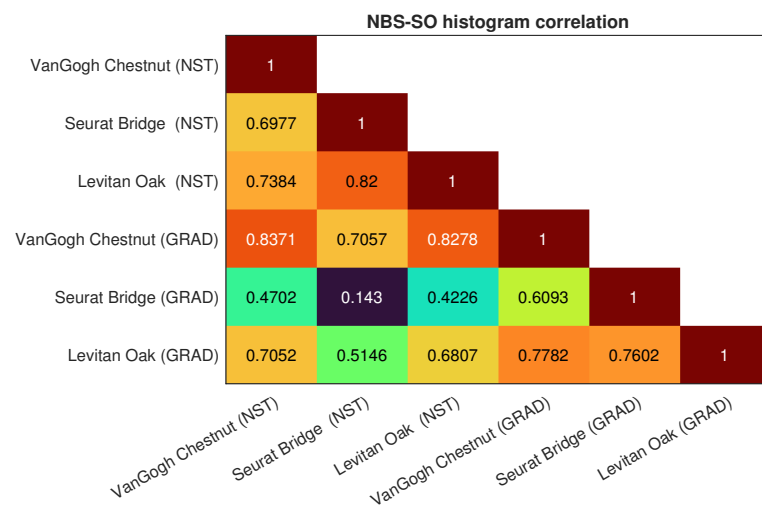


Figure A15. NBS-SO histogram correlation for three paintings of different painters in test 2.

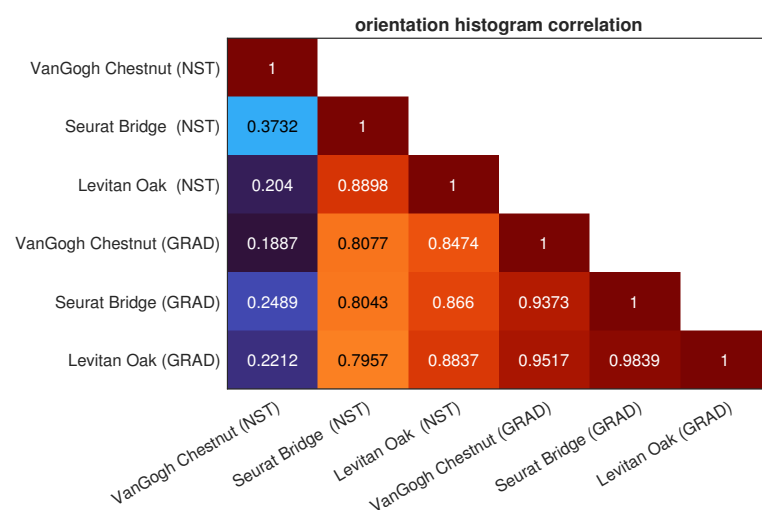


Figure A16. Brushstroke orientation histogram correlation for three paintings of different painters in test 2.

Appendix C. Results for Test 3

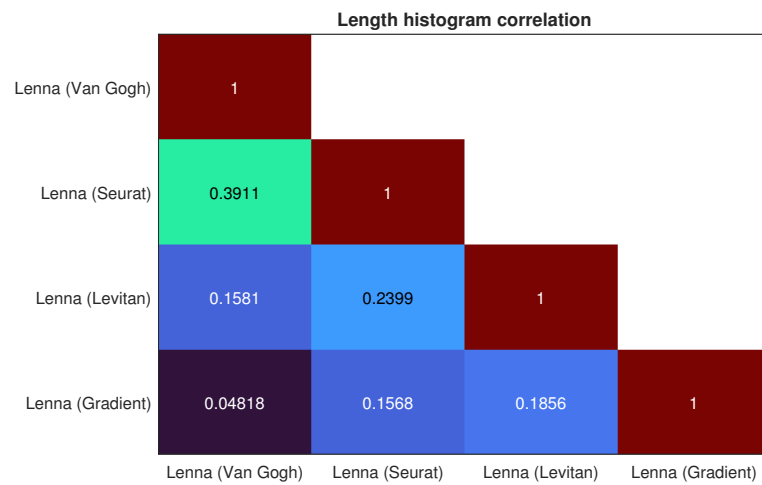


Figure A17. Brushstroke length histogram correlation for three paintings rendered with the NST algorithm and one painting with the gradient algorithm in test 3.

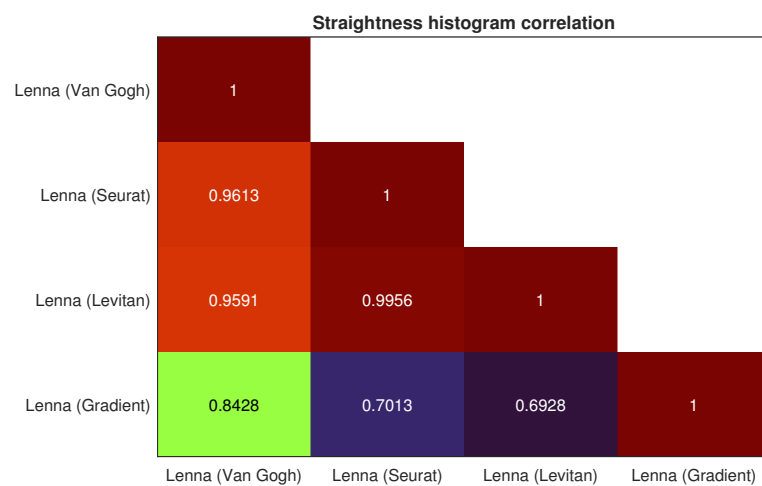


Figure A18. Brushstroke straightness histogram correlation for three paintings rendered with the NST algorithm and one painting with the gradient algorithm in test 3.

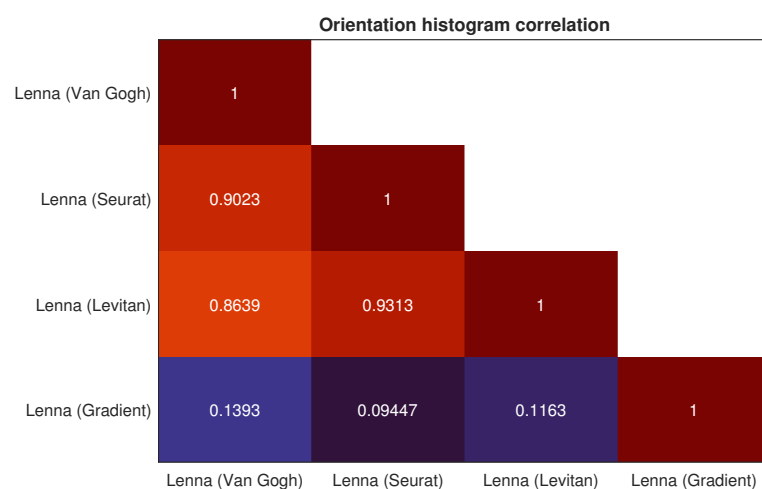


Figure A19. Brushstroke orientation histogram correlation for three paintings rendered with the NST algorithm and one painting with the gradient algorithm in test 3.

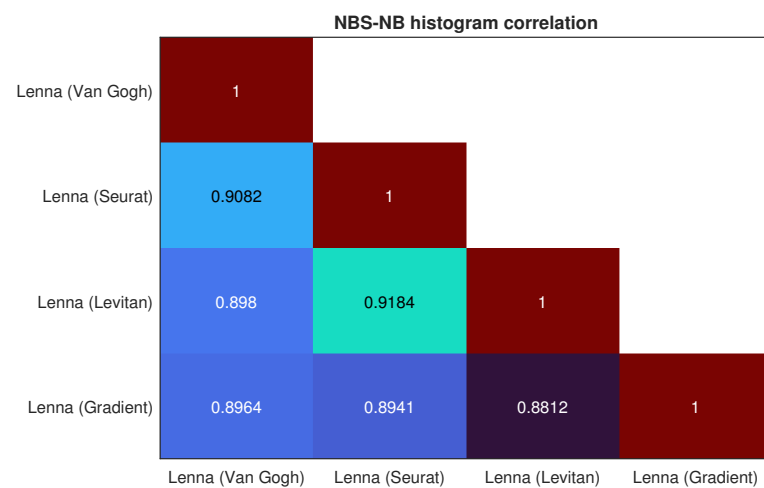


Figure A20. NBS-NB histogram correlation for three paintings rendered with the NST algorithm and one painting with the gradient algorithm in test 3.

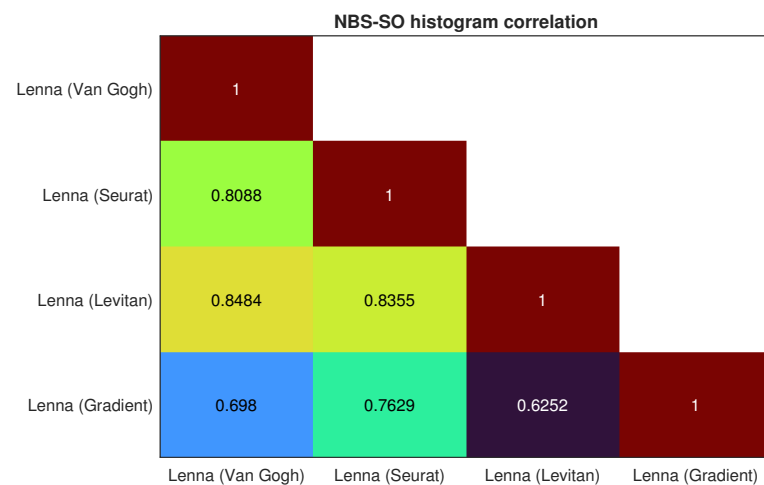


Figure A21. NBS-SO histogram correlation for three paintings rendered with the NST algorithm and one painting with the gradient algorithm in test 3.

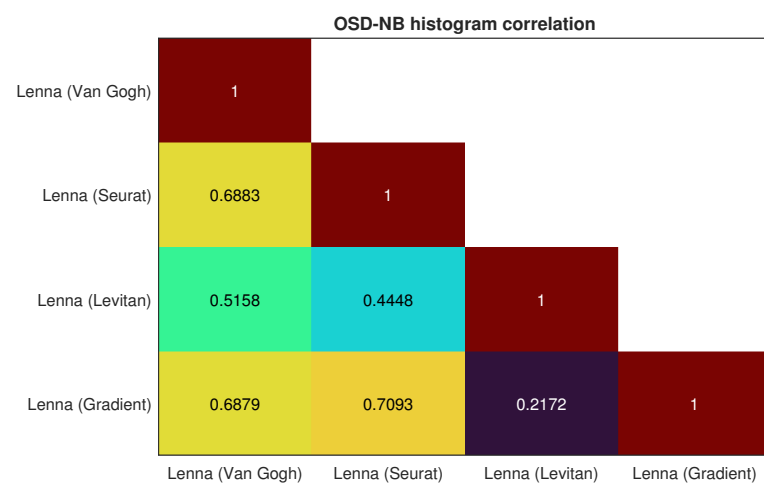


Figure A22. OSD-NB histogram correlation for three paintings rendered with the NST algorithm and one painting with the gradient algorithm in test 3.

References

1. Gooch, B.; Gooch, A. *Non-Photorealistic Rendering*; CRC Press: Boca Raton, FL, USA, 2001.
2. Hertzmann, A. Non-photorealistic rendering and the science of art. In Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering, Annecy, France, 7–10 June 2010; pp. 147–157.
3. Hertzmann, A. Can computers create art? *Arts* **2018**, *7*, 18. [\[CrossRef\]](#)
4. Kumar, M.P.; Poornima, B.; Nagendraswamy, H.; Manjunath, C. A comprehensive survey on non-photorealistic rendering and benchmark developments for image abstraction and stylization. *Iran J. Comput. Sci.* **2019**, *2*, 131–165. [\[CrossRef\]](#)
5. Dijkzeul, D.; Brouwer, N.; Pijning, I.; Koppenhol, L.; Van den Berg, D. Painting with evolutionary algorithms. In Proceedings of the Artificial Intelligence in Music, Sound, Art and Design: 11th International Conference, EvoMUSART 2022, Held as Part of EvoStar 2022, Madrid, Spain, 20–22 April 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 52–67.
6. Scalera, L.; Seriani, S.; Gasparetto, A.; Gallina, P. Non-photorealistic rendering techniques for artistic robotic painting. *Robotics* **2019**, *8*, 10. [\[CrossRef\]](#)
7. Karimov, A.; Kopets, E.; Kolev, G.; Leonov, S.; Scalera, L.; Butusov, D. Image preprocessing for artistic robotic painting. *Inventions* **2021**, *6*, 19. [\[CrossRef\]](#)
8. Karimov, A.I.; Pesterev, D.O.; Ostrovskii, V.Y.; Butusov, D.N.; Kopets, E.E. Brushstroke rendering algorithm for a painting robot. In Proceedings of the 2017 International Conference “Quality Management, Transport and Information Security, Information Technologies” (IT&QM&IS), Saint Petersburg, Russia, 24–30 September 2017; pp. 331–334.
9. Gerr, J. The Comic Artist’s Tools Suite: Centralized and Intuitive Non-Photorealistic Computer Graphics Renderings. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2022.
10. Mazzone, M.; Elgammal, A. Art, creativity, and the potential of artificial intelligence. *Arts* **2019**, *8*, 26. [\[CrossRef\]](#)
11. Guo, C.; Bai, T.; Lu, Y.; Lin, Y.; Xiong, G.; Wang, X.; Wang, F.Y. Skywork-daVinci: A novel CPSS-based painting support system. In Proceedings of the 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE), Hong Kong, China, 20–21 August 2020; pp. 673–678.
12. Jing, Y.; Yang, Y.; Feng, Z.; Ye, J.; Yu, Y.; Song, M. Neural style transfer: A review. *IEEE Trans. Vis. Comput. Graph.* **2019**, *26*, 3365–3385. [\[CrossRef\]](#)
13. Singh, A.; Jaiswal, V.; Joshi, G.; Sanjeeve, A.; Gite, S.; Kotecha, K. Neural style transfer: A critical review. *IEEE Access* **2021**, *9*, 131583–131613. [\[CrossRef\]](#)
14. Gatys, L.A.; Ecker, A.S.; Bethge, M. Image style transfer using convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2414–2423.
15. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [\[CrossRef\]](#)
16. Cheng, M.M.; Liu, X.C.; Wang, J.; Lu, S.P.; Lai, Y.K.; Rosin, P.L. Structure-preserving neural style transfer. *IEEE Trans. Image Process.* **2019**, *29*, 909–920. [\[CrossRef\]](#)
17. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2223–2232.
18. Isola, P.; Zhu, J.Y.; Zhou, T.; Efros, A.A. Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1125–1134.
19. Johnson, J.; Alahi, A.; Fei-Fei, L. Perceptual losses for real-time style transfer and super-resolution. In Proceedings of the Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Proceedings, Part II 14; Springer: Berlin/Heidelberg, Germany, 2016; pp. 694–711.
20. Ulyanov, D.; Lebedev, V.; Vedaldi, A.; Lempitsky, V. Texture networks: Feed-forward synthesis of textures and stylized images. *arXiv* **2016**, arXiv:1603.03417.
21. Vanderhaeghe, D.; Collomosse, J. Stroke based painterly rendering. In *Image and Video-Based Artistic Stylisation*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 3–21.
22. Haeberli, P. Paint by numbers: Abstract image representations. In Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques, Dallas, TX, USA, 6–10 August 1990; pp. 207–214.
23. Hertzmann, A. Painterly rendering with curved brush strokes of multiple sizes. In Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, Orlando, FL, USA, 19–24 July 1998; pp. 453–460.
24. Zeng, K.; Zhao, M.; Xiong, C.; Zhu, S.C. From image parsing to painterly rendering. *ACM Trans. Graph.* **2009**, *29*, 2.
25. Lu, J.; Barnes, C.; DiVerdi, S.; Finkelstein, A. Realbrush: Painting with examples of physical media. *ACM Trans. Graph. (TOG)* **2013**, *32*, 1–12. [\[CrossRef\]](#)
26. Lindemeier, T.; Metzner, J.; Pollak, L.; Deussen, O. Hardware-Based Non-Photorealistic Rendering Using a Painting Robot. *Comput. Graph. Forum* **2015**, *34*, 311–323. [\[CrossRef\]](#)
27. Beltramello, A.; Scalera, L.; Seriani, S.; Gallina, P. Artistic robotic painting using the palette knife technique. *Robotics* **2020**, *9*, 15. [\[CrossRef\]](#)

28. Guo, C.; Bai, T.; Wang, X.; Zhang, X.; Lu, Y.; Dai, X.; Wang, F.Y. ShadowPainter: Active learning enabled robotic painting through visual measurement and reproduction of the artistic creation process. *J. Intell. Robot. Syst.* **2022**, *105*, 61. [\[CrossRef\]](#)
29. Karimov, A.; Kopets, E.; Leonov, S.; Scalera, L.; Butusov, D. A Robot for Artistic Painting in Authentic Colors. *J. Intell. Robot. Syst.* **2023**, *107*, 34.
30. Fu, Y.; Yu, H.; Yeh, C.K.; Lee, T.Y.; Zhang, J.J. Fast accurate and automatic brushstroke extraction. *ACM Trans. Multimed. Comput. Commun. Appl. (TOMM)* **2021**, *17*, 1–24. [\[CrossRef\]](#)
31. Xie, N.; Zhao, T.; Tian, F.; Zhang, X.; Sugiyama, M. Stroke-based stylization learning and rendering with inverse reinforcement learning. In Proceedings of the 24th International Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015; pp. 2531–2537.
32. Nolte, F.; Melnik, A.; Ritter, H. Stroke-based Rendering: From Heuristics to Deep Learning. *arXiv* **2022**, arXiv:2302.00595.
33. Kotovenko, D.; Wright, M.; Heimbrecht, A.; Ommer, B. Rethinking style transfer: From pixels to parameterized brushstrokes. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 12196–12205.
34. Mandryk, R.L.; Mould, D.; Li, H. Evaluation of emotional response to non-photorealistic images. In Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Non-Photorealistic Animation and Rendering, Vancouver, BC, Canada, 5–7 August 2011; pp. 7–16.
35. Mould, D.; Mandryk, R.L.; Li, H. Emotional response and visual attention to non-photorealistic images. *Comput. Graph.* **2012**, *36*, 658–672. [\[CrossRef\]](#)
36. Santella, A.; DeCarlo, D. Visual interest and NPR: An evaluation and manifesto. In Proceedings of the 3rd International Symposium on Non-Photorealistic Animation and Rendering, Annecy, France, 7–9 June 2004; pp. 71–150.
37. Mould, D. Authorial subjective evaluation of non-photorealistic images. In Proceedings of the Workshop on Non-Photorealistic Animation and Rendering, Vancouver, BC, Canada, 8–10 August 2014; pp. 49–56.
38. Hong, K.; Jeon, S.; Yang, H.; Fu, J.; Byun, H. Domain-aware universal style transfer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 14609–14617.
39. Deng, Y.; Tang, F.; Dong, W.; Ma, C.; Pan, X.; Wang, L.; Xu, C. Stytr2: Image style transfer with transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 11326–11336.
40. Maciejewski, R.; Isenberg, T.; Andrews, W.M.; Ebert, D.S.; Sousa, M.C.; Chen, W. Measuring stipple aesthetics in hand-drawn and computer-generated images. *IEEE Comput. Graph. Appl.* **2008**, *28*, 62–74. [\[CrossRef\]](#)
41. Dowson, D.; Landau, B. The Fréchet distance between multivariate normal distributions. *J. Multivar. Anal.* **1982**, *12*, 450–455. [\[CrossRef\]](#)
42. Lucic, M.; Kurach, K.; Michalski, M.; Gelly, S.; Bousquet, O. Are gans created equal? A large-scale study. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 698–707.
43. Wright, M.; Ommer, B. Artfid: Quantitative evaluation of neural style transfer. In Proceedings of the Pattern Recognition: 44th DAGM German Conference, DAGM GCPR 2022, Konstanz, Germany, 27–30 September 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 560–576.
44. Wang, Z.; Zhao, L.; Chen, H.; Zuo, Z.; Li, A.; Xing, W.; Lu, D. Evaluate and improve the quality of neural style transfer. *Comput. Vis. Image Underst.* **2021**, *207*, 103203. [\[CrossRef\]](#)
45. Gatys, L.; Ecker, A.S.; Bethge, M. Texture synthesis using convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 262–270.
46. Karimov, A.I.; Kopets, E.E.; Rybin, V.G.; Leonov, S.V.; Voroshilova, A.I.; Butusov, D.N. Advanced tone rendition technique for a painting robot. *Robot. Auton. Syst.* **2019**, *115*, 17–27. [\[CrossRef\]](#)
47. Lamberti, F.; Sanna, A.; Paravati, G. Computer-assisted analysis of painting brushstrokes: Digital image processing for unsupervised extraction of visible features from van Gogh’s works. *EURASIP J. Image Video Process.* **2014**, *2014*, 53. [\[CrossRef\]](#)
48. Li, J.; Yao, L.; Hendriks, E.; Wang, J.Z. Rhythmic brushstrokes distinguish van Gogh from his contemporaries: Findings via automated brushstroke extraction. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *34*, 1159–1176.
49. Bakurov, I.; Ross, B.J. Non-photorealistic rendering with cartesian genetic programming using graphics processing units. In Proceedings of the Computational Intelligence in Music, Sound, Art and Design: 7th International Conference, EvoMUSART 2018, Parma, Italy, 4–6 April 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 34–49.
50. Collomosse, J.P. Supervised genetic search for parameter selection in painterly rendering. In Proceedings of the Applications of Evolutionary Computing: EvoWorkshops 2006: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoINTERACTION, EvoMUSART, and EvoSTOC, Budapest, Hungary, 10–12 April 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 599–610.
51. Collomosse, J.P. Evolutionary search for the artistic rendering of photographs. In *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 39–62.
52. Kang, H.W.; Chakraborty, U.K.; Chui, C.K.; He, W. Multi-scale stroke-based rendering by evolutionary algorithm. In Proceedings of the International Workshop on Frontiers of Evolutionary Algorithms, JCIS, Salt Lake City, UT, USA, 21–26 July 2005; pp. 546–549.

53. Ross, B.J.; Ralph, W.; Zong, H. Evolutionary image synthesis using a model of aesthetics. In Proceedings of the 2006 IEEE International Conference on Evolutionary Computation, Vancouver, BC, Canada, 16–21 July 2006; pp. 1087–1094.
54. Putri, T.; Mukundan, R.; Neshatian, K. Artistic style characterization and brush stroke modelling for non-photorealistic rendering. In Proceedings of the 2017 International Conference on Image and Vision Computing New Zealand (IVCNZ), Christchurch, New Zealand, 4–6 December 2017; pp. 1–7.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.