Stroke-GAN Painter: Learning to Paint Artworks Using Stroke-Style Generative Adversarial Networks

Qian Wang
Macau University of Science and Technology
anrogim@outlook.com

Cai Guo
Macau University of Science and Technology
& Hanshan Normal University
c.guo@hstc.edu.cn

Hong-Ning Dai
Lingnan University
hndai@ieee.org

Ping Li
The Hong Kong Polytechnic University
p.li@polyu.edu.hk

Abstract

It is a challenging task to teach machines or computers to paint like human artists in a stroke-by-stroke fashion. Despite advances in stroke-based image rendering and deep learning-based image rendering, the existing painting methods have limitations: 1) lacking the flexibility to choose different art-style strokes, 2) losing content details of images, and 3) generating few art styles of paintings. In this paper, we propose a Stroke-Style Generative Adversarial Network (namely Stroke-GAN) to solve the first two limitations. Our Stroke-GAN learns styles of strokes from different stroke-style datasets and produces diverse stroke styles. We design three players in Stroke-GAN to generate pure-color strokes close to human artists' strokes, thereby improving the quality of painting details. Regarding the limitation 3, based on our Stroke-GAN, we then devise a neural network named Stroke-GAN Painter that generates different art styles of paintings. Experiments demonstrate that our artful painter generates different styles of paintings while well-preserving content details (such as human face texture and details of buildings) and retaining a higher similarity to the given images.

1. Introduction

Painting, as an important visual art, symbolizes human imagination and ingenuity. Human artists have used a variety of painting instruments to create their artworks with specific characteristic styles. However, it is time-consuming for people to master painting skills due to a huge amount of time being spent on learning, imitating and practising. The recent emerging computer-aided painting methods generate non-photorealistic images similar to paintings, thereby offering effective painting-assistants for human painting learners. But it is still a challenging task to teach machines to paint artworks referring to the given images like human artists. Different from directly generating a style-transfer image or photographic image [31, 9, 35, 37], machine painting is created by a machine or a computer in a stroke-by-stroke manner. The key to teaching a machine to mimic human artists lies in addressing the following three challenges: 1) painting on the canvas stroke by stroke according to a certain stroke order with a given reference image; 2) creating strokes containing the stroke texture similar to human artists' strokes; 3) preserving detailed contents of a given image to create a painting instead of reconstructing a photorealistic image.

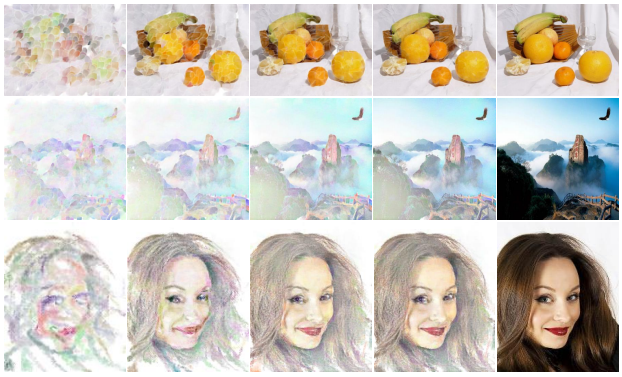

Figure 1. Learning-to-paint process of Stroke-GAN Painter. The last column shows the input reference images. Rows from top to bottom show oil paintings, watercolor paintings, pastel paintings, respectively. Here, N is the the number of painting times instead of the stroke number.

Some conventional methods include stroke-based rendering (SBR) methods [11, 13, 20] make contributions to stroke modeling. The quality of the stroke texture is good to mimic the human's strokes. But, these methods achieve the

semi-automatic painting process which needs users' substantial interventions. Meanwhile, this process is time-consuming and requires users' high painting skills. Moreover, these SBR models have *a limited number of painting styles*. Compared with conventional SBR models, learning-based methods have flexible frameworks which can adapt to diverse art styles. In addition, learning-based methods can create paintings without users' interventions. Recently, researchers typically design recurrent neural networks (RNN) [10, 36] and Reinforcement Learning (RL) models [7, 33, 16] to generate stroke-by-stroke artworks. However, this unified framework is *lacking flexibility in choosing different styles of strokes* and some art-style generated paintings (i.e., pastel-like paintings) are still *losing meticulous details*.

To address the above limitations of existing methods, we propose a new method leveraging advantages of both the conventional SBR methods and the learning-based methods. We first design a novel Stroke Generative Adversarial Network (Stroke-GAN) to learn different stroke styles from stroke-style datasets and generate diverse stroke styles with adjustable stroke shapes, size, transparency, and color. Based on Stroke-GAN, we then design a neural-network painter to learn to create different styles of paintings in a stroke-by-stroke manner, such as oil paintings, watercolor paintings, pastel paintings. We name the entire framework as *Stroke-GAN Painter*. The process of our Stroke-GAN Painter learning to generate a painting is a coarse-to-fine manner, as shown in Fig. 1. In particular, our Stroke-GAN Painter learns to paint from a novice to a veteran, i.e., the painting quality becomes better by repeating multiple learning-to-paint processes. The quality of the painting becomes better with the increased painting times. In contrast to existing methods, such as sketching [10, 28], doodling [7], Neural Painter (NP) [25], MDRL Painter (MDRLP) [16], our painter generates more diverse art styles of paintings with different types of strokes. Moreover, the images generated by our painter also well preserve key content details (such as face details of portraits) as shown in Fig. 1. The contributions of this paper are summarized as follows:

- We propose ***a three-player-game model***, namely Stroke-GAN, to generate art-style of strokes, which are fully adjustable in stroke shapes, size, transparency, and color, thereby greatly improving the stylization of generated paintings. We design two generators and one discriminator in Stroke-GAN, where the second generator learns to purify the stained-color strokes generated by the first generator. Consequently, the generated strokes have pure colors and close textures to human artists' strokes.

- We design a painter based on Stroke-GAN. We do not need to train Stroke-GAN Painter on any image datasets. Our Stroke-GAN Painter learns to create ***different art styles*** of paintings based on stylized strokes, such as oil paintings, watercolor paintings, and pastel paintings, in a unified framework. Our generated paintings well preserve content details of reference images while offering stylization diversity of paintings.

- Experimental results show that our painter generates diverse art styles of paintings covering various content types, such as portraits, animals, landscapes, and buildings. Paintings generated by our Stroke-GAN Painter well preserve content details such as eyes and teeth of portraits and meticulous details of buildings. User studies from participants covering various backgrounds, ages, and genders demonstrate that paintings generated by our Stroke-GAN Painter win the most votes with 76.9% of votes in pastel paintings between two compared methods and win 31.2% of votes in oil paintings among four compared methods in terms of fidelity and stylization.

2. Related work

We briefly survey the most related studies on machine paintings. We roughly classify related studies into conventional stroke-based rendering, learning-based rendering, and image-style transfer (IST).

2.1. Conventional stroke-based rendering

SBR methods mainly reconstruct images into non-photorealistic imagery with stroke-based models. Researchers adopt SBR methods to different types of artworks, e.g., paintings [11, 13, 20], pen-and-ink drawings [4, 32], and stippling drawings [2, 3]. Specifically, the work[11] introduces a semi-automatic painting method based on a greedy algorithm while this method needs substantial human interventions to control the stroke shapes and select the stroke location. The authors of [13] also propose a style design for their painting method by using spline-brush strokes to *render* the image though this method requires high painting skills of users. The work [20] proposes a method to segment an image into areas with similar levels of salience to control the strokes. However, most of these methods require substantial human interventions to choose key parameters, thereby not convenient for ordinary users. Moreover, SBR methods only generate a limited number of styles [14], consequently leading to inflexibility to diverse art styles.

2.2. Learning-based rendering

Recently, researchers have adopted learning-based methods to improve the painting effect compared with traditional SBR methods. SPIRAL [7, 24] develops an adversarially-trained deep reinforcement learning (DRL) agent which learns structures of images though it does not work well

on reconstructing details of human portraits. Moreover, Sketch-RNN [10] constructs stroke-based drawings via training on human-drawn image datasets to achieve excellent results of common objects though it only depicts simple-line paintings. StrokeNet [36] trains their agent to learn to paint based on a differentiable render and recurrent neural network (RNN) while it is poor when generalizing on color images. Further, computers are now able to generate more realistic oil paintings [16, 38, 22] and pastel-like paintings [25]. MDRLP [16] paints oil-painting-like pictures with a small number of strokes though it only mimics one style in a unified framework and loses brushstroke textures. Methods such as [38, 22] improve the stroke texture by redesigning their stroke renders. NP [25] has a similar design to ours since both dedicated to generating stroke by a GAN-based module. However, our approach differs from that method on several aspects. First, we design a three-player GAN model to generate adjustable strokes while NP only uses a normal GAN to generate fixed strokes. Second, our method can learn any stroke datasets while NP must use the stroke produced by a program namely MyPaint as the stroke dataset. Third, NP requires massive manually-labelled stroke dataset to generate one-by-one action-stroke. Their model requires the same strokes as the stroke dataset provided by MyPaint to ensure that the optimized strokes are those that the painting model needs. Our model has no data-labelling owing to the fact that our model can paint well by using the three-player Stroke-GAN to generate strokes similar to the stroke dataset.

2.3. Image style transfer

Image style transfer (IST) methods have become a trending topic in both research projects and industrial applications [8, 18, 6, 34, 1] though a few methods have been applied to stroke-based image rendering. PaintBot [17] based on a DRL network recreates the target image in a stroke-by-stroke manner while the painting style is only restricted to the style of the reference image. Moreover, Neural Painter [25] designs a method to generate style strokes to recreate the target image without style reference images in their model. However, this method requires a large manually-labelled stroke dataset and also lacks flexibility in choosing different styles of strokes. Moreover, the generated paintings lose some meticulous details, e.g., the details of humane face and textures of buildings.

In this paper, we propose a new learning-based method namely *Stroke-GAN Painter*. Our Stroke-GAN Painter integrates the advance of SBR methods with learning-based methods to generate diverse styles of paintings with high quality. Specifically, we design Stroke-GAN to generate different styles of strokes and Stroke Designer Modules to generate different style stroke datasets for our Stroke-GAN. Meanwhile, we design an Artist Module to optimize the stroke selection to "use" Stroke-GAN generating well-behaved strokes which are rendered onto the canvas to create high quality paintings.

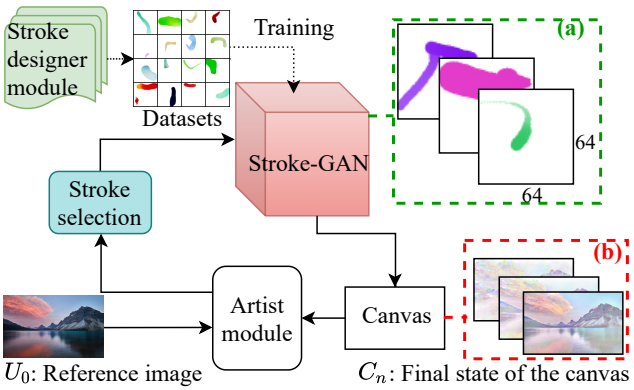

Figure 2. Network architecture of Stroke-GAN Painter mainly consists of Stroke-GAN, the Artist Module and Canvas. (a) shows style strokes generated by Stroke-GAN. (b) shows the states of the canvas during the learning-to-paint process.

3. Stroke-GAN painter

We propose a new painting model (namely Stroke-GAN painter) to achieve stroke-by-stroke painting for machines or computers. The goal of our Stroke-GAN painter is to paint ***diverse art styles*** of paintings in a unified framework. We mainly consider oil paintings, watercolor paintings, and pastel paintings on the canvas in this paper though extra painting styles can also be easily adopted to our framework. When given an image, our model can continually render style-strokes onto the canvas to create different art styles of paintings by choosing the style of strokes. Fig. 2 depicts our proposed Stroke-GAN painter consisting of Stroke Designer Module, Stroke-GAN, the Artist Module, and Canvas. The Stroke Designer Module provides different styles of stroke datasets for training Stroke-GAN, which generates style strokes (a). The Artist Module feeds in both U_0 and C_n to optimize Stroke Selection, which controls Stroke-GAN to generate well-behaved strokes. The states of the canvas during the learning-to-paint process are shown in (b). Stroke-GAN Painter learns to create paintings from a novice to a veteran and the painting quality is improved with more painting times. Section 3.1 presents Stroke Designer Module to generate different styles of stroke datasets for training Stroke-GAN. Section 3.2 presents Stroke-GAN, which feeds in values obtained from *stroke selection* to generate style strokes, which are then painted on Canvas C_n. The Artist Module feeds in both the reference image U_0 and Canvas C_n to optimize stroke selection, thereby finishing the painting process. Section 3.3 next describes the painting process and stroke selection optimization.

3.1. Stroke designer module

Stroke modeling. The art style of a painting can be affected by the stroke style. We artfully devise the Stroke Designer Module to generate stroke datasets for training Stroke-GAN. Inspired by previous studies [25, 16], our Stroke Designer Modules consider the following variables: $\mathbb{P}$ denotes the set of the control points of a Bézier curve, $\mathbb{S}$ denotes the set of the size of a geometric shape, $\mathbb{T}$ denotes a set to control the transparency of the stroke, and $\mathbb{V}$ denotes the set to control the stroke color.

- Stroke shape: We use different geometric shapes to represent a brush tip and Bézier curves to mimic the route of a brush. The points in $\mathbb{P} = \{(x_i, y_i)|i = 0, 1, 2, ..., d\}$ control Bézier curves, where d is the degree of Bézier curve. Bézier curves can be expressed explicitly as Eq. (1).
- Stroke size: We use $\mathbb{S} = \{s_0, s_1\}$ to define the size of a geometric shape to control the stroke size. The size of the brush tip varies with the values of $\mathbb{S}$.
- Stroke transparency: We use $\mathbb{T} = \{t_0, t_1\}$ to control the transparency of the stroke. The transparency of the stroke varies with the values of $\mathbb{T}$.
- Color: Primary colors denoted by sect $\mathbb{V} = \{r, g, b\}$ can determine the color of the stroke.

Stroke datasets. The Stroke Designer Module provides different stroke-style datasets for training Stroke-GAN. Each stroke dataset includes 200,000 stroke images, each with 64×64 resolution. We mainly adopt three most representative stroke styles (though we essentially generate more than five styles of strokes): 1) watercolor style, 2) oil-painting style, and 3) pastel-painting style. The pastel-painting style is provided by [25] while both oil-painting style and watercolor style are generated by the Stroke Designer Module. We use Bézier curves (BC) to simulate the route of the stroke and different circles to mimic the tip of a brush. Each stroke is made by 100 variant circles moving along BC.

$$\mathbf{B}(t) = \sum_{i=0}^{d} \binom{d}{i} (1-t)^{(d-i)} t^i \mathbf{P}_i, \quad t \in [0, 1], \quad (1)$$

where $\mathbf{P}_i$ denotes the control point with coordinates $(x_i, y_i) \in \mathbb{P}$.

3.2. Stroke-GAN

In order to improve the painting quality and the fidelity of the stroke, we design ***the three-player GAN model***, namely Stroke-GAN, to generate stylized strokes for the painting process. Stroke-GAN is the core component to endow our model with diverse art styles of paintings following a given image in a unified framework while preserving the high fidelity of details owing to the coloring module (i.e., the third player) in a GAN. Paintings mainly contain several elements, e.g., lines, textures, colors and so on [19]. Our Stroke-GAN is designed to generate strokes containing with these elements. Therefore, the stroke style has a major influence on the painting style. Moreover, our Stroke-GAN is an end-to-end training model so that our painter framework has the flexibility to change the art styles by choosing different trained Stroke-GAN models. We do not need to train the whole painter on any image datasets since the model is designed to learn to paint from a novice to a veteran.

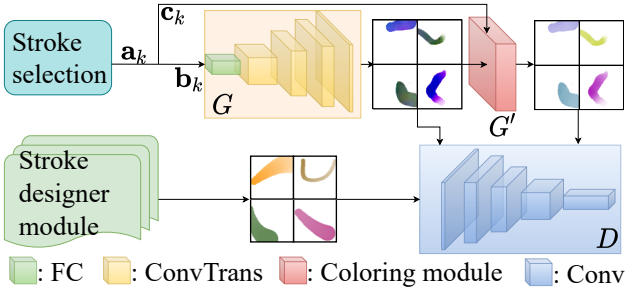

Figure 3. Structure of Stroke-GAN. The Stroke Designer Module provides stroke datasets. Stroke-GAN consists of a normal generator G, a coloring module G' and a discriminator D.

3.2.1 Motivation to design Stroke-GAN

We design our Stroke-GAN mainly based on Deep Convolutional Generative Adversarial Network (DCGAN) [26], which offers more stable training than conventional GANs. But the strokes generated by the normal DCGAN (i.e., two-player game) are stained with shade colors. It is difficult to mimic the painting process of human artists by using stained-colored strokes. To address this problem, we design a second generator (i.e., the third player) to re-color strokes with three color parameters to generate pure-colored strokes, which are better for painting (closer to the stroke painted by human artists).

3.2.2 Three-player-game of Stroke-GAN

In order to control stroke colors, we design the coloring module as the third player (i.e., the red cuboid) immediately following the normal generator (being essentially a convolutional neural network), as shown in Fig. 3. Stroke-GAN consists of a normal generator G, a coloring module G' and a discriminator D. After being fed with stroke selections, G learns to generate stroke images but stained-colored, and G' learns to generate pure-color strokes. The discriminator first determines whether a stroke generated by G is valid or not, and then determines the stroke image generated by G'. Both G and G' update during the adversary mode with D. In particular, G' learns to purify the stained-

color strokes generated by G according to $\mathbf{c}_k$. Since the randomness of DCGAN leads to the unexpected stroke color, we design the coloring module (G') to control the stroke color. Particularly, the stroke selection is controlled by $\mathbf{a}_k$ with 50 parameters (used as inputs of Stroke-GAN), where $\mathbf{a}_k = [\mathbf{b}_k, \mathbf{c}_k]$ ($k = 1, ..., K$ and K is the batch size as in Algorithm 1). The normal generator (G) feeds the variable $\mathbf{b}_k$ with 47 parameters and outputs the stroke image $G(\mathbf{b}_k)$ while $\mathbf{c}_k$ is used to control the stroke color $\{r, g, b\}$ fed into G'. Stroke-GAN improves the original DCGAN to purify the shade colors in strokes (as shown in Fig. 3). The coloring module feeds $\mathbf{c}_k$ with the stroke $G(\mathbf{b}_k)$ to learn to purify the stroke.

Since the stroke image just contains the background and the stroke, we can use *threshold segmentation* [5] to separate the stroke region and the background region. Let us denote the matrix of pixels in the stroke image by $\mathbf{P} = [p_1, ..., p_c]$, where c is the number of channels of the image. We denote the average of $\mathbf{P}$ by $\overline{\mathbf{P}}$. Since the pixel values of the background in the stroke image generated by G are unknown, we should train G' to find the threshold of the background pixels. We denote the threshold by γ. We use threshold segmentation to eliminate the background region from the stroke image, and the result is denoted by $\widetilde{\mathbf{P}}$, which is obtained as follows:

$$\widetilde{\mathbf{P}} = \gamma - \overline{\mathbf{P}}. \tag{2}$$

The values of the elements in $\widetilde{\mathbf{P}}$ are 0 or close to 0 in the background region. We use $\min(\cdot)$ and $\max(\cdot)$ to calculate the minimum and maximum values in $\widetilde{\mathbf{P}}$, respectively. We calculate the stroke region and denote the results by $\mathbf{S}_P$, which is obtained by

$$\mathbf{S}_P = (\widetilde{\mathbf{P}} - \min(\widetilde{\mathbf{P}}))/(\max(\widetilde{\mathbf{P}}) - \min(\widetilde{\mathbf{P}})). \tag{3}$$

In particular, the values of the elements in $\mathbf{S}_P$ close to 1 are the stroke pixels, and the values close to 0 represent the background pixels. We then use $\mathbf{c}_k$ to recolor the stroke and obtain the pure stroke image $\mathbf{P}_S$ as follows:

$$\mathbf{P}_S = [\mathbf{S}_P, \mathbf{S}_P, \mathbf{S}_P] \cdot \mathbf{c}_k. \tag{4}$$

The coloring module endues our painter with more creativity in painting, e.g., outputting paintings with different colors even from the same reference image owing to Stroke-GAN learning the color of the reference image directly by feeding in $\mathbf{c}_k$. This is the reason that Stroke-GAN can be trained without data-labelling. Even though Stroke-GAN generates strokes not the same as the dataset, the coloring module learning the color from the reference image can still ensure the rendered canvas close to the reference image. The design of the coloring module plays an important role to let a GAN generate realistic human strokes. Moreover, this design can also make Stroke-GAN easy to learn different styles of strokes. Therefore, a unified framework utilized Stroke-GAN can generate different styles of paintings.

Algorithm 1 Training Procedure for Stroke-GAN

Input: Number of training iterations T, batch size K
Output: Discriminator denoted by D, normal generator denoted by G, and the coloring module denoted by G'

1: **for** $t = 1, ..., T$ **do**
2: Sample examples $\mathbf{x} = \{\mathbf{x}_1, ..., \mathbf{x}_k\}$ from training dataset;
3: Sample normal random noise samples $\mathbf{a} = \{\mathbf{a}_1, ..., \mathbf{a}_k\}$, where $\mathbf{a}_k = [\mathbf{b}_k, \mathbf{c}_k]$;
4: Update D with the stochastic gradient:
$$\bigtriangledown_{\theta_d} \frac{1}{K} \sum_{k=1}^{K} \Big[- \log\Big(D(\mathbf{x}_k)\Big) - \log\Big(1 - D\big(G(\mathbf{b}_k)\big)\Big) - \log\Big(1 - D\big(G'(G(\mathbf{b}_k), \mathbf{c}_k)\big)\Big)\Big];$$
5: Update G with the stochastic gradient:
$$\bigtriangledown_{\theta_g} \frac{1}{K} \sum_{k=1}^{K} - \log\Big(D\big(G(\mathbf{b}_k)\big)\Big);$$
6: Update G' with the stochastic gradient:
$$\bigtriangledown_{\theta_g} \frac{1}{K} \sum_{k=1}^{K} - \log\Big(D\big(G'(G(\mathbf{b}_k), \mathbf{c}_k)\big)\Big);$$
7: **end for**

3.2.3 Training of Stroke-GAN

We train Stroke-GAN to get different stroke models to endow our painter with painting abilities of different stroke styles. Fig. 3 depicts the structure of Stroke-GAN. Stroke-GAN is trained with 128 images as a mini-batch for each stroke dataset containing 200,000 images. We use the whole dataset as the training set since the DCGAN model has no need to make the validation. When training Stroke-GAN, we directly feed the generator with the set of stroke parameters ($\mathbf{a}_k$) to generate a stroke image. The initial values of these parameters are random. During the training process, the generator learns to produce images close to the dataset by optimizing the values of these parameters. We use Adam optimizer to train the Stroke-GAN model, and the learning rate is 0.0002, the values of beta are 0.5 and 0.999. Each pair of a generated stoke image and a real stroke image is then fed into the discriminator. The discriminator next determines whether the pair of strokes is valid or not. If the generated stroke image is similar to the real stroke image, the pair is valid; invalid otherwise.

The training procedure for Stroke-GAN is given in Algorithm 1. The parameters of $\mathbf{a}_k$ in Algorithm 1 also mean the noise samples. We essentially train normal generator G and the coloring module G' and the discriminator D by back-propagating the loss so as to update the parameters θ_g, θ_c and θ_d, respectively. In particular, the discriminator D has the loss ℓ_d and the generator G has the loss ℓ_g. We use Binary Cross Entropy (BCE) denoted by $\ell(\mathbf{z}, \mathbf{y})$ to measure

the loss of $\mathbf{z}$ on conditional variable $\mathbf{y}$, the function is given as follows,

$$\ell(\mathbf{z},\mathbf{y}) = \frac{1}{K}\sum_{k=1}^{K}\big(-\mathbf{y_k}\cdot\log(\mathbf{z_k}) - (1-\mathbf{y_k})\cdot\log(1-\mathbf{z_k})\big). \tag{5}$$

When training the discriminator on real stroke images, $\mathbf{y} = 1$, according to Eq. (5), we get the loss of real strokes denoted by ℓ_{dr}, as follows:

$$\ell_{dr} = \ell\big(D(\mathbf{x}), 1\big). \tag{6}$$

When training the discriminator on fake stroke images generated by G, we then have $\mathbf{y} = 0$ according to Eq. (5), we get the loss of fake strokes denoted by ℓ_{da}, as follows:

$$\ell_{da} = \ell\big(D(G(\mathbf{b}_k)), 0\big). \tag{7}$$

When training the discriminator on fake stroke images generated by G', we then have $\mathbf{y} = 0$ according to Eq. (5), we get the loss of fake strokes denoted by ℓ_{db}, as follows:

$$\ell_{db} = \ell\Big(D\big(G'(G(\mathbf{b}_k), \mathbf{c}_k)\big), 0\Big). \tag{8}$$

We next have the entire loss of the discriminator denoted by $\ell_d = \ell_{dr} + \ell_{da} + \ell_{db}$ as follows:

$$\begin{aligned}\ell_d = \frac{1}{K}\sum_{k=1}^{K}\Big[&-\log\Big(D(\mathbf{x}_k)\Big) - \log\Big(1 - D\big(G(\mathbf{b}_k)\big)\Big) \\ &- \log\Big(1 - D\big(G'(G(\mathbf{b}_k), \mathbf{c}_k)\big)\Big)\Big].\end{aligned} \tag{9}$$

Similarly, the normal generator G has the loss as follows:

$$\ell_g = \frac{1}{K}\sum_{k=1}^{K} -\log\Big(D\big(G(\mathbf{b}_k)\big)\Big). \tag{10}$$

And the loss function of the coloring module is designed as follows:

$$\ell_c = \frac{1}{K}\sum_{k=1}^{K} -\log\Big(D\big(G'(G(\mathbf{b}_k), \mathbf{c}_k)\big)\Big), \tag{11}$$

where the strokes in $G(\mathbf{b}_k)$ are visually stained. The coloring module learns to compute the content of the stroke region in $G(\mathbf{b}_k)$ and re-color the stroke by $\mathbf{c}_k$.

As shown in Fig. 4(a), our Stroke-GAN converges on different stroke datasets. Fig. 4(b) compares sample strokes generated by Stroke-GAN with the coloring module and that without the coloring module in contrast to stroke datasets. Images with red borders in the second column and the third column denote the strokes with mixed colors and pure colors generated by Stroke-GAN W/O CM and W CM, respectively. The stained-color strokes generated by Stroke-GAN W/O CM cannot mimic human artists' strokes. The pure-color strokes generated by Stroke-GAN W CM are close to human artists' strokes, thereby being beneficial to improve the quality of the content details. Although both of Stroke-GAN W/O CM and Stroke-GAN W CM can generate strokes similar to the given strokes, Stroke-GAN without the coloring module generates parti-colored strokes in contrast to Stroke-GAN with the coloring module generating pure-colored strokes, which are better for painting (closer to human artists). Our Stroke-GAN can learn any styles of strokes with the given certain dataset. In this paper, we adopt three stoke datasets: watercolor strokes, oil-painting strokes, and pastel strokes to generate style strokes. We save the models trained on watercolor-stroke dataset, oil-painting-stroke dataset, pastel-stroke dataset as Style1, Style2, and Style3, respectively. We then choose the corresponding Stroke-GAN model (namely "Style+NO.") for a certain art style.

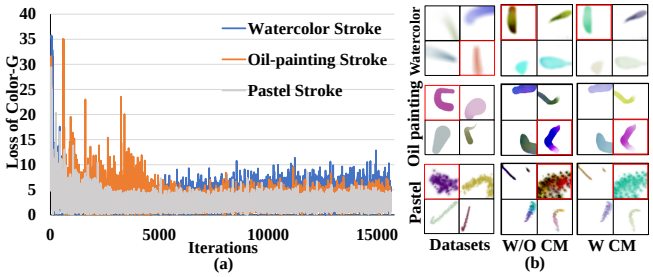

Figure 4. The training process of Color-G. (a) plots the loss of Color-G component with iterations on three stroke datasets; (b) shows stroke samples generated by two comparative methods: Stroke-GAN with coloring module (W CM) and without coloring module (W/O CM).

In summary, our Stroke-GAN has the following *merits: 1) our Stroke-GAN can recolor the stroke by the design of the coloring module, thereby improving the artistic creativity of the painter. For example, the color of the painting can be recreated close to but not the same as the input reference image; 2) our Stroke-GAN is flexible to learn any style strokes as long as the stroke dataset is available owing to the completeness and independence of Stroke-GAN; 3) our Stroke-GAN enables end-to-end training so that it can be easily applied in the painting model to change art styles by choosing different Stroke-GAN models.*

3.3. Artist Module

We endow our painter with the capability in painting diverse art styles. Besides using diverse stroke-styles generated by Stroke-GAN, we also consider the feature-extracting network (FEN) to extract contents of reference images. After being processed by FEN, the original reference image may lose some contents but retaining the core information

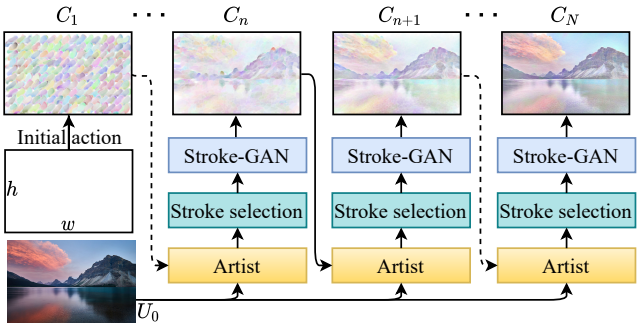

Figure 5. The coarse-to-fine learning-to-paint process. The reference image is denoted by U_0, C_n denotes a certain state of the canvas, and N is the painting times. The width and the height of the canvas are denoted by w and h, respectively.

of the image. We design the Artist Module with FEN and an optimization algorithm. We use FEN in our Artist Module to process the features of the reference image and the canvas. We use the optimization algorithm to "pick" well-behaved strokes for rendering the canvas.

3.3.1 Painting Process

We design our painter to mimic the painting process of human artists with a given style. This painting process is conducted in a coarse-to-fine manner, in which our painter learns to paint from scratch to a fine-grained painting after multiple times of painting. Our Stroke-GAN generates a sequence of strokes at one time and the Artist Module optimizes the stroke selection in Section 3.3.2 to "render" these strokes on the canvas. The painting process is shown in Fig. 5. The painting model consists of the Artist Module, stroke selection, canvas, and Stroke-GAN (generating strokes). The Artist Module optimizes the stroke selection and the Stroke-GAN generates strokes used to continually paint. Stroke-GAN Painter learns to paint from a novice to a veteran. We observe that the painting quality becomes better by repeating multiple learning-to-paint processes. The painting quality becomes better with the increased n. The reference image denoted by U_0. A certain state of the canvas and the set of stroke selections are denoted by C_n and A_n, respectively. The height h and the width w of the canvas are automatically configured according to the aspect ratio of the reference image.

We model our painting process as a Stroke-State-Optimizing-Process with a state set $\mathcal{S}$, a stroke selection $\mathcal{A}$ and a mapping f: $\mathcal{S} \rightarrow \mathcal{A}$. We denote $\mathcal{S} = \{C_n | n = 1, 2, ..., N\}$, $\mathcal{A} = \{A_n | n = 1, 2, ..., N\}$, where N is the number of the painting times, also the number of the iteration of the painting model. We denote the total number of the strokes needed to complete the painting by K. Since each A_n has K elements, we then choose $A_n = \{\mathbf{a}_k | k = 1, 2, ..., K\}$. One $\mathbf{a}_k$ in the stroke selection is used to pick one stroke (generated by Stroke-GAN). For each iteration, the stroke selection outputs a set of (the number is K) $\mathbf{a}_k$ to let Stroke-GAN generate K strokes. One painting process is finished when K strokes all rendered onto the canvas, i.e., one C_n is done.

A sequence of strokes generated by Stroke-GAN are essentially images, each with 64×64 pixels. Thus, the canvas is divided into $h \times w$ grids for rendering convenience, and the size of each grid is also 64×64 pixels. We render the sequence of strokes into the canvas grid-by-grid. In each grid, the content at the stroke position in the new stroke image replaces that at the corresponding position. Stroke-GAN produces K strokes at a time, where K equals the number of strokes in each grid multiplied by $h \times w$ grids. The strokes are sequentially rendered in the grid-by-grid order. Stroke-GAN runs once in one painting process. The mapping f: $\mathcal{S} \rightarrow \mathcal{A}$ adapts the transition function $C_{n+1} = f(C_n, A_{n+1})$. The stroke selection first outputs an initial set A_1; each element in A_1 denotes an effect for rendering a stroke at a certain position of the canvas. The artist module then optimizes the stroke selection via the stroke-selection-optimizing algorithm and generates a new set of elements A_n, which are used to render strokes on the canvas to get C_n. We continue the coarse-to-fine process from C_n to C_{n+1}, where C_{n+1} denotes a fine-grained painting (with optimized strokes). Continuing the above process, we finally obtain the well-done painting C_N.

3.3.2 Stroke selection optimization

We design an Artist Module to optimize the stroke selection. The Artist Module consists of FEN and the optimization algorithm. During the painting process, the Artist Module first feeds in both the reference image and the painted canvas to compute the distance between them, then optimizes the stroke selection. Therefore, the stroke selection *picks* well-behaved strokes generated by Stroke-GAN. In other words, the state of the canvas C_{n+1} is better than C_n. This procedure continues until one painting process completes and a painting C_N is generated after N painting process. The learning-to-paint process works in a coarse-to-fine manner.

It is crucial to optimize the stroke selection. The Artist Module first extracts the features from U_0 and C_n and then compute the distance between them by the ℓ_1-distance. We denote the extracted features from the reference image U_0 and those from the painted canvas C_n by $\mathcal{F}(U_0) = \{I_j | j = 1, 2, ..., M\}$ and $\mathcal{F}(C_n) = \{c_j | j = 0, 1, 2, ..., M\}$, respectively, where M is the number of features extracted by the neural Artist Module. In particular, I_j, c_j denote the features of U_0 and those of a certain state of canvas C_n, respectively. We calculate the ℓ_1-distance loss function denoted by

$\mathcal{L}(U_0, C_n)$ as follows

$$\mathcal{L}(U_0, C_n) = \frac{1}{M} \sum_{j=1}^{M} |I_j - c_j|. \tag{12}$$

The ℓ_1-distance loss function essentially computes the distance between the features of reference image U_0 and those of canvas C_n. The stroke-selection-optimization algorithm optimizes the stroke selection by tuning the values of the parameters in each $\mathbf{a}_k$ according to the ℓ_1-distance loss function. The function $f(C_n, A_{n+1})$ is done by the backpropagation algorithm for $\mathcal{L}(U_0, C_n)$. Every state of the canvas C_n is composed of K strokes, and every stroke is determined by $\mathbf{a}_k$. Thus, the values of the parameters in each $\mathbf{a}_k$ can be optimized by the backpropagation for $\mathcal{L}(U_0, C_n)$.

3.4. Style reconstruction

As explained in Section 3.2, Stroke-GAN is the core component to generate the style. The Artist Module also contributes to the stylization of the whole painting. In particular, we use FEN in the Artist Module to extract features of the reference image as well as the painted canvas. This process brings some content loss of the original image but mimicking paintings close to the original image. This step can make the generated painting different but similar to the reference image, i.e., the recreation of the art mimesis or "realism". We reconstruct the painting style by using Stroke-GAN and FEN in the Artist Module. In particular, Stroke-GAN endows the painter with diverse styles of strokes and FEN in the Artist Module creates the art style.

3.4.1 Art style

Since [25] indicates that the content objective preserves only the high-level features while the parameterization can fill out the details, we also only take the high-level features as inputs. We adopt two most representative deep neural networks: GoogleNet [29] or Residual nets (ResNets) [12] for the digital Artist Module. The design of using FEN to process the original reference image not directly using the original image endues our painter with the artistic creativity while retaining a high similarity to the reference image. We focus on the realism of the oil painting so that we use ResNet to build FEN in the Artist Module. ResNets have a high accuracy on information extracting [12] so that ResNets can keep a high fidelity of the extracted features. Therefore, we use ResNet to mimic the realism of the oil painting. On the other hand, features extracted by GoogleNet are relatively sparse so that it may offer more creative space for our painter, e.g., mimesis and diversity. Therefore, we use GoogleNet to for watercolor and pastel paintings (integrated with the style strokes).

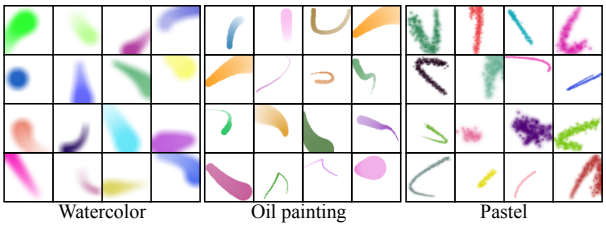

Figure 6. Samples of three kinds of stroke styles. Note that, the oil-painting strokes have sharp contours and volatile routes. When stacking multiple strokes on the canvas, the oil-painting texture can be recognised easily via the sharp contours and volatile routes.

3.4.2 Stroke style

Different strokes can bring different styles of artwork even though the different type strokes are used by the same human artist. We design three kinds of strokes to endow our painter with more creativity. Fig. 6 shows different stroke styles: the watercolor strokes, oil-painting strokes, pastel strokes from left to right. The watercolor strokes have smooth and soft contours and the routes of the brushes are simple and pure. In contrast, oil-painting strokes have sharp contours and volatile routes. Moreover, the pastel strokes look being accumulated with many uneven points (mimicking granular textures of pastel paintings). These different styles of strokes render the canvas so as to show different styles of the paintings. After utilizing different FENs to process the reference image to integrate with different types of strokes, we can obtain different styles of paintings.

4. Experimental results

We evaluate our painters with several experiments. We first introduce the implementation. Then, we evaluate three styles of paintings generated by our painter and compare the performance of the proposed Stroke-GAN painter with state-of-the-art (SOTA) methods. Finally, we conduct ablation studies to compare how our Stroke-GAN painter generates different styles of paintings by fine-tuning the design.

4.1. Implementation

Our experiments were conducted on a workstation with an i7-7700k CPU and an NVIDIA Titan RTX GPU. We evaluate our painter on three image datasets: CelebA [23], ImageNet [27], and real-world photos. These images cover various types of contents including portraits, animals, landscapes, and buildings. All the images used in experiments are labelled as "Img No.". We use Style1 (Stroke-GAN model) and the Artist Module composed of GoogleNet to generate watercolor-stroke paintings, Style2 and the Artist Module composed of ResNet to generate oil-painting-stroke paintings, Style3 and the Artist Module composed of GoogleNet to generate pastel-stroke paintings.

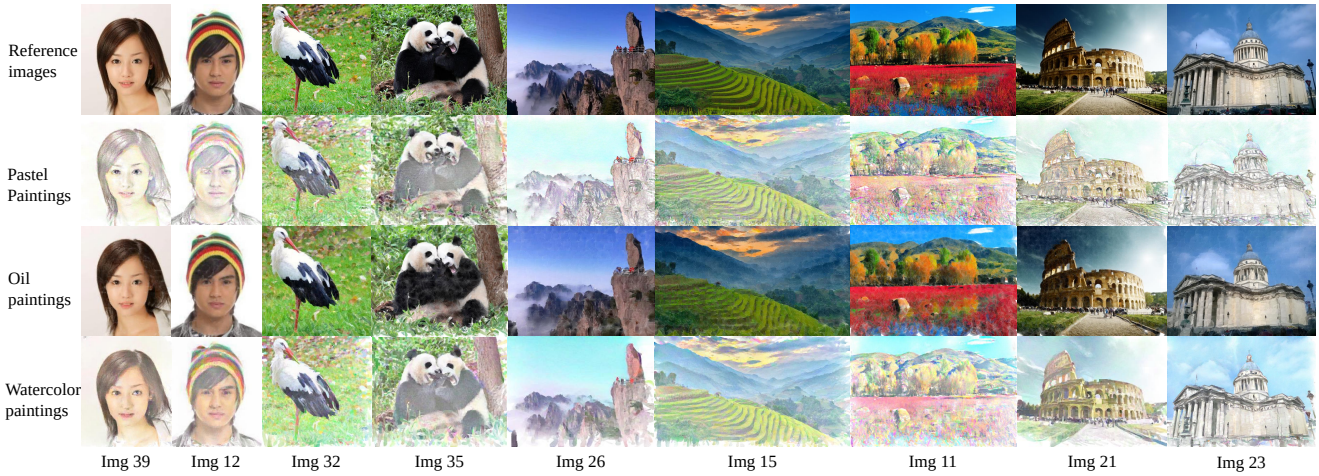

Figure 7. Paintings generated by our method from CelebA [23], ImageNet [27], and real-world images, covering various types of contents, such as portraits, animals, landscapes, and buildings. The top-row images are the reference images while our painter generates the rest three rows of images: the pastel-stroke paintings, oil-painting-stroke paintings, and watercolor-stroke paintings.

4.2. Comparison of stroke styles

We compare paintings generated by different styles of strokes on CelebA, ImageNet, and real-world photos. As shown in Fig. 7, the top-row images are the reference images while our painter generates the rest three rows of images: the pastel-stroke paintings, oil-painting-stroke paintings, and watercolor-stroke paintings. We also label images with "Img No." for the experiment, where images labelled with Img 39 and Img 12 are taken from CelebA [23], Img 32 and Img 35 are taken from ImageNet [27], the rest of them are real-world images. Fig. 7 illustrates that all the generated paintings exhibit different styles in contrast to the reference images. In particular, the pastel-stroke paintings in the second row preserve enough textures and lines while losing some color features. The oil-painting-stroke paintings in the third row well preserve textures, lines, and color features, consequently capturing meticulous details to reference images. Meanwhile, we observe stroke textures from oil paintings comparing with reference images, demonstrating the oil-painting stylization. The watercolor-stroke paintings in the fourth row exhibit a stylization between pastel paintings and oil paintings though this painting style is good at expressing watercolor styles for scenic and building images (see Img 26, 15, 11, 21, and 23 of the bottom row in Fig. 7).

Fig. 8 plots the ℓ_1-distance testing results between the generated images and the reference images. Specifically, Fig. 8(a) plots the ℓ_1-distance versus painting times with consideration of three stroke styles: oil-painting stroke, watercolor stroke, and pastel stroke. We observe that all the three styles nearly converge after 300 painting times though the oil-painting stroke style converges faster than other two styles. Differently, the pastel-stroke style paintings converge slower than watercolor and oil paintings since

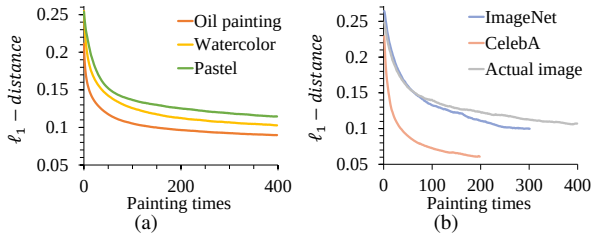

Figure 8. The ℓ_1-distance between the generated images and reference images. (a) shows the ℓ_1-distance of different stroke styles on real-world photos; (b) is ℓ_1-distance of different datasets of the watercolor-stroke style.

pastel-stroke paintings lose more content details. Fig. 8(b) compares three types of image datasets with the same watercolor-stroke style. We observe that it took 200 painting times to recreate the images of CelebA, 300 painting times to generate images of ImageNet, and 400 painting times to obtain images of real-world photos. Because the portrait images of CelebA are relatively easier to learn for our painter than those of ImageNet and real-world photos due to fewer features.

4.3. Comparison with prior methods

We evaluate our painter by comparing it with the state-of-the-art learning-based methods, including Neural Painter (NP) [25], MDRL Painter (MDLRP) [16], SNP [38] and PaintTF [22] that outperform other learning methods and traditional SBR methods. Meanwhile, we consider two representative art styles of our model: the pastel-stroke painting and oil-stroke painting for comparison. This is because our model can generate diverse art styles of paintings. In particular, we adopt NP to generate the pastel-stroke paintings to compare with the paste-stroke paintings generated

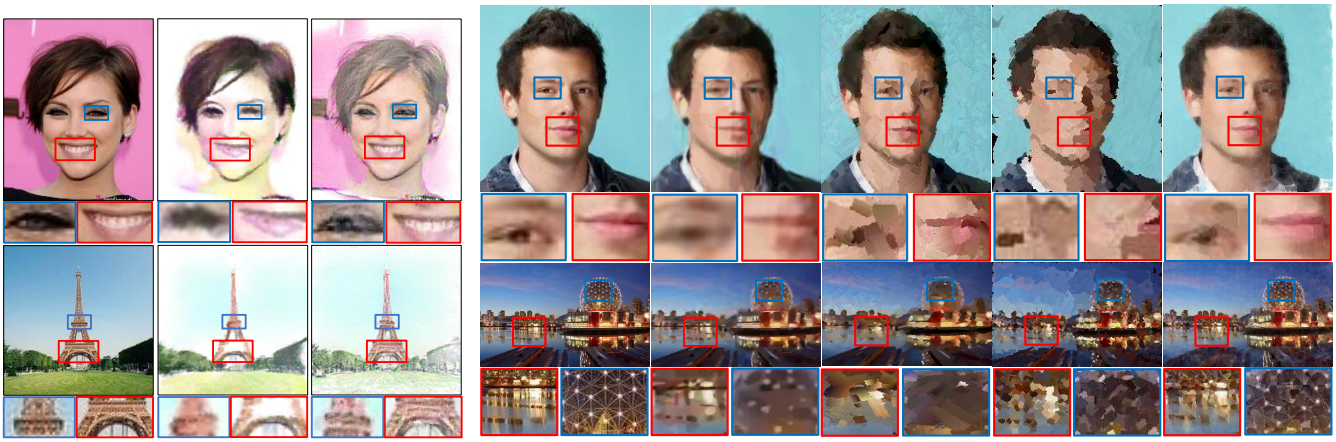

Figure 9. Comparison with prior methods. (a) Pastel-stroke paintings generated by our painter and NP [25]. (b) Oil-painting-stroke paintings generated by our painter, MDRLP [16], SNP [38] and PaintTF [22].

by our painter. Similarly, we employ MDRLP, SNP and PaintTF to generate oil paintings for comparison with our painter. In order to obtain the best paintings generated by the compared methods (mentioned in their papers), we use the pre-trained models provided by the authors and choose the parameters to be the default values given by the authors.

4.3.1 Qualitative comparison

Fig. 9 compares visual effect of paintings generated by our painter, NP [25], MDRLP [16] SNP [38] and PaintTF [22]. Fig. 9(a) compares pastel paintings generated by NP and our painter. We observe that our painter generates images with more details and textures than NP. For example, we cannot identify the face texture and teeth of the woman's portrait in the image generated by NP while the image generated by our painter well preserves those details, thereby looking more vivid. Compared with the stroke generating, NP [25] only generates fixed strokes, while our Stroke-GAN generates adjustable strokes thanks to the coloring module. Therefore, strokes are tuned according to the input image to retain more details.

Fig. 9(b) compares oil paintings generated by MDRLP, SNP, PaintTF and our painter. We observe that images generated by our painter have less content loss than those generated by MDRLP, SNP and PaintTF. It is quite obvious when comparing the zoomed blocks, our painter well preserves the details of the man's eyes and mouth and textures of the building. Our model uses the independent Stroke-GAN to generate strokes with diverse shapes and variant sizes so as to depict detailed contents. However, brushstrokes used in SNP and PaintTF have few variants on the stroke shape owing to the stroke directly generated by their entire models. In particular, their models have only two shapes of strokes despite variant stroke size and angles. On the other hand, the stroke-texture representation is different among all compared methods. MDRLP presents the stroke textures while losing some contents since they have no special process of the stroke to mimic the stroke textures. Meanwhile, more strokes and painting steps can make the result be the same as the input photo instead of a painting. SNP and PaintTF present stroke textures by adding a stroke-texture mask after the stroke is generated. In other words, the stroke contains no textures when it is generated and the textures have no affect on optimizing the stroke. Our Stroke-GAN painter renders the stroke texture by generating a sharp contour that mimics the thick edge of oil paints in a stroke. Therefore, the painting results present irregular-line textures instead of the brush textures.

4.3.2 Quantitative comparison

To further evaluate the quality of paintings generated by our Stroke-GAN painter and the state-of-the-art methods, we conduct a two-step user study inspired by [15, 30]: User Study I and User Study II. For fair comparison, we conduct blind trials, in which neither participants knew the methods generated comparison paintings nor the authors of this paper. User Study I focuses on the favor of people to the art-style paintings generated by our Stroke-GAN painter and the state-of-the-art methods. User Study II focuses on the fidelity of the content details and the recognizability of stroke textures of the paintings generated by our Stroke-GAN painter and the state-of-the-art methods.

User Study I. User Study I is conducted by two questionnaires, the first one is to compare the pastel style artworks, and the second one is to compare the oil-painting style artworks. Since User Study I is designed to evaluate the favor

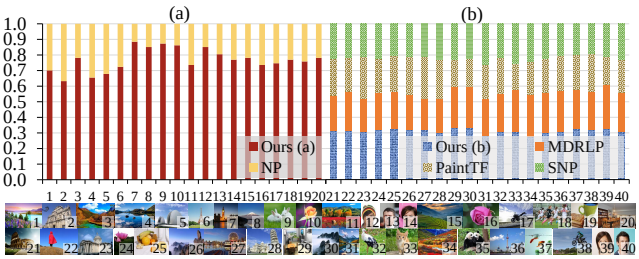

Figure 10. User Study I. The vertical axis and horizontal axis denote the percentage of users' votes and image pairs, respectively. (a) Results on pastel-stroke paintings generated by our painter and NP [25]. (b) Results on oil paintings generated by our painter, MDRLP [16], SNP [38] and PaintTF [22].

of people to artworks generated by different methods, we do not emphasize the background of users in the comparison, although the users are from artistic and non-artistic backgrounds. In the first questionnaire, we arbitrarily choose 20 images from CelebA [23] (3 images), ImageNet [27] (6 images), and real-world photos (11 photos); those images also cover various types of contents including landscapes, buildings, animals, and portraits. The first group of participants are chosen from various background (10% in the art field), age groups (17-50), and genders where there are 44 females and 43 males. User Study I is conducted to evaluate the effect of the pastel-stroke paintings generated by our painter and NP [25].

For each reference image (numbered from 1 to 20 in Fig. 10), we obtain a pair of images painted by our painter and NP. Consequently, there are 20 pairs of images for the user study. We mainly evaluate the *favor* and the *stylization* of generated images. Thus, we ask participants to choose which image represents a pastel-stroke painting and is also favoured in each pair of images. Fig. 10(a) depicts the results. We observe that more participants choose the images generated by our painter in all the 20 pairs of images, i.e., we obtain at least 63.22% and 76.9% (on average) of all the votes, implying that images generated by our painter are closer to pastel paintings. Similarly, we conduct the second questionnaire to evaluate the oil-painting effect with comparison of our painter, MDRLP [16], SNP [38] and PaintTF [22]. The second group of participants are also chosen from diverse background, ages, and genders (i.e., 40 females and 32 males). We also select 20 images from CelebA, ImageNet, and real-world images to cover different content types (see the numbered images from 21 to 40 in the bottom row of Fig. 10). We also ask participants to choose which image is closer to an oil painting and more popular in each pair of images. We have a similar observation from Fig. 10(b) that most participants (31.2% among four methods) choose images generated by our painter for oil paintings among the compared methods.

User Study II. We further conduct User Study II to compare the paintings generated by our method and the state-of-the-art methods in terms of the content details and stroke textures. We also conduct two user-study questionnaires (Likert scale [21]) for pastel painting style and oil painting style, respectively. The participants are divided into two groups: users with artistic background and users without artistic background. All the participants are chosen from various age groups (17-50) and genders where there are 20 females and 5 males for each user-study questionnaire. We compare the average score (μ), variance (σ), the 95% confidence interval of paintings generated by each method. Tab. 1 and Tab. 2 show the results of the group without artistic background, and the results of the group with artistic background, respectively.

In Tab. 1 (without artistic background), the content details of paintings generated by MDRLP [16], PaintTF [22] and SNP [38] gain low scores, i.e., lower than 3. One reason lies in the fact that their paintings lose too many details, and the stroke textures generated by MDRLP [16] are also difficult to be recognized for most of the users. In contrast, the paintings generated by our Stroke-GAN painter have positive scores on both content details and stroke textures. Similarly, in Tab. 2 (with artistic background), our method gains a higher evaluation score on both these two items than other methods. Comparing Tab. 1 and Tab. 2 together, users without artistic background give higher evaluations than users with artistic background on most of the compared methods. On the other hand, both users with and users without artistic background have higher evaluation on our paintings than other methods. In particular, the average score denoted by μ of the content details reaches 3.940 and the upper bound is 4.060 (in 95% confidence interval) as shown in Tab. 1. In Tab. 2, the average score of the stroke texture reaches 3.956 with the upper bound 4.141. Interestingly, for the representation of stroke textures, users without artistic background give higher score (2.816) for MDRLP [16] than users with artistic background (2.583), and the highest score given by users without artistic background is 3.468 (our method). However, there are opposite scores given by users with artistic background. The scores given for SNP [38] and PaintTF [22] are higher than our method although our score is 3.604 (close to these two methods).

In summary, both pastel-stroke paintings and oil-painting-stroke painting of our method contain more detailed contents than the compared methods. For non-artistic users, the stroke textures are not well presented by most methods, and for artistic users, PaintTF [22], SNP [38] and our methods (both pastel and oil-painting) can present stroke textures well.

User Study III. In order to evaluate the aesthetic of the art-style results, we further conduct User Study III to evaluate the items of color tone and aesthetic beauty of the painting. We also conduct two user-study questionnaires (Likert

Table 1. User Study II. Scores of paintings generated by our methods and SOTA methods about content details and stroke textures without artistic background.

Items	Methods	μ	σ	95% confidence interval	
				Lower Bound	Upper Bound
Content	NP [25]	3.626	0.181	3.530	3.723
	Ours	**3.940**	0.227	3.820	**4.060**
	MDRLP [16]	2.839	0.308	2.683	2.996
	PaintTF [22]	2.471	0.386	2.274	2.668
	SNP [38]	2.374	0.298	2.153	2.595
	Ours	3.361	0.214	3.251	3.611
Stroke	NP [25]	3.605	0.265	3.465	3.746
	Ours	3.722	0.258	3.585	3.859
	MDRLP [16]	2.816	0.286	2.670	2.962
	PaintTF [22]	2.582	0.368	2.394	2.769
	SNP [38]	2.576	0.294	2.358	2.794
	Ours	3.468	0.201	3.366	3.571

Table 2. User Study II. Scores of paintings generated by our methods and SOTA methods about content details and stroke textures with artistic background.

Items	Methods	μ	σ	95% confidence interval	
				Lower Bound	Upper Bound
Contents	NP [25]	3.446	0.225	3.258	3.635
	Ours	3.775	0.229	3.584	3.967
	MDRLP [16]	2.717	0.446	2.459	2.974
	PaintTF [22]	2.397	0.559	2.074	2.719
	SNP [38]	2.237	0.567	1.909	2.564
	Ours	3.803	0.313	3.623	3.984
Strokes	NP [25]	3.594	0.286	3.355	3.833
	Ours	**3.956**	0.221	3.772	**4.141**
	MDRLP [16]	2.583	0.398	2.353	2.813
	PaintTF [22]	3.668	0.214	3.544	3.791
	SNP [38]	3.682	0.153	3.594	3.770
	Ours	3.604	0.239	3.466	3.742

Table 3. User Study III. Scores of paintings generated by our methods and SOTA methods about artistic sense on color tone and beauty without artistic background.

Items	Methods	μ	σ	95% confidence interval	
				Lower Bound	Upper Bound
Color tone	NP [25]	**3.703**	0.316	3.535	**3.870**
	Ours	3.688	0.306	3.526	3.851
	MDRLP [16]	3.211	0.274	3.071	3.350
	PaintTF [22]	2.934	0.363	2.749	3.119
	SNP [38]	2.845	0.346	2.588	3.101
	Ours	3.708	0.190	3.611	3.805
Beauty	NP [25]	3.782	0.386	3.577	3.986
	Ours	3.833	0.293	3.678	3.988
	MDRLP [16]	2.963	0.327	2.796	3.130
	PaintTF [22]	2.537	0.380	2.343	2.731
	SNP [38]	2.484	0.356	2.220	2.748
	Ours	3.508	0.188	3.412	3.604

Table 4. User Study III. Scores of paintings generated by our methods and SOTA methods about artistic sense on color tone and beauty with artistic background.

Items	Methods	μ	σ	95% confidence interval	
				Lower Bound	Upper Bound
Color tone	NP [25]	3.857	0.243	3.654	4.060
	Ours	3.913	0.250	3.704	4.121
	MDRLP [16]	3.823	0.238	3.686	3.961
	PaintTF [22]	3.607	0.290	3.439	3.774
	SNP [38]	3.743	0.237	3.606	3.880
	Ours	**4.217**	0.357	4.010	4.423
Beauty	NP [25]	3.707	0.346	3.418	3.996
	Ours	3.753	0.218	3.571	3.935
	MDRLP [16]	3.550	0.327	3.361	3.739
	PaintTF [22]	3.267	0.555	2.946	3.587
	SNP [38]	3.470	0.454	3.208	3.732
	Ours	**3.937**	0.400	3.706	4.167

scale [21]) for pastel painting style and oil painting style, respectively. The compared input images are the same as images used in User Study II. The participants are divided into two groups: users with artistic background (15) and users without artistic background (19). All the participants are chosen from various age groups (21-40) and genders where there are 18 females and 16 males for each user-study questionnaire. We compare the average score (μ), variance (σ), the 95% confidence interval of paintings generated by each method. Tab. 3 and Tab. 4 show the results of the group without artistic background, and the results of the group with artistic background, respectively.

In Tab. 3, scores of color tone and aesthetic beauty given by users without artistic background for pastel paintings (NP [25] and our method) are higher than 3. On the other hand, oil-paintings obtain lower scores. Especially, paintings generated by PaintTF [22] and SNP [38] gain much lower scores than our method and MDRLP [16] on the item of aesthetic beauty. Considering the scores of the content item in Tab. 1 (given by users without artistic background) are also lower than 3, the paintings generated by PaintTF [22] and SNP [38] lose too many detailed contents so that users without artistic background give low scores on the beauty item. Although users without artistic background did not give high scores, the ranks of the compared methods about the aesthetic beauty item and the color-tone item keep the same.

On the other hand, users with artistic background give higher scores than users without artistic background. In Tab. 4, paintings generated by all the compared methods obtain scores higher than 3. Especially, our paintings gain 4.217 on the item of color tone and 3.937 on the item of aesthetic beauty. Comparing Tab. 4 with Tab. 3, our method, MDRLP [16], PaintTF [22] and SNP [38] achieve much higher scores than NP [25]. In other words, the evaluation of pastel-style paintings generated by NP has little difference between users with artistic background and users without artistic background. However, the difference between these two kinds of users is obvious when evaluating the oil-painting style paintings. For example, on the item of aesthetic beauty, users without artistic background give higher scores for paintings of PaintTF than SNP while users with artistic background give lower scores. But all users give consistent evaluation. In particular, on both color tone and aesthetic beauty, our method is better than others while PaintTF [22] and SNP [38] rank the last.

Comparing Tab. 1, Tab. 2, Tab. 3 and Tab. 4 together, our method gains the highest comprehensive scores among the state-of-the-art methods. Meanwhile, NP performs well on both content and color-tone items; MDRLP performs well on the color-tone item; PaintTF and SNP perform well on the stroke-texture item.

4.4. Ablation study

In this section, we investigate how our painter generates different styles of paintings with different FENs and strokes.

Alternative Feature-Extracting Networks. Recall that we choose GoogleNet as the FEN for watercolor and pastel-stroke images and ResNet as the FEN for oil-painting images. We exploit a new FEN with a combination of GoogleNet and ResNet, namely (G+R) to generate paintings. We denote the ℓ_1-distance of features extracted by GoogleNet and the ℓ_1-distance of features extracted by ResNet by $\mathcal{L}_G(U_0, C_n)$ and $\mathcal{L}_R(U_0, C_n)$, respectively, according to Eq. (12). In particular, the loss function of G+R network can be written as

$$\mathcal{L}(U_0, C_n) = 0.5\mathcal{L}_G(U_0, C_n) + 0.5\mathcal{L}_R(U_0, C_n). \quad (13)$$

We only make the backpropagation algorithm for the final $\mathcal{L}(U_0, C_n)$.

Fig. 11 depicts the results of three types of FENs with four different styles of paintings denoted by strokes Style1, Style2, Style3 and Style4. Note that Style4 is a new stroke designed with a hollow circle and Cubic Bézier curve and Style4 stroke dataset is generated by a similar method to Style1. We observe from Fig. 11 that the G+R network can generate a new style of paintings similar to both pastel-stroke and oil-painting images. Moreover, we also find that an FEN essentially determines the *art style* and a stroke determines the *painting style*. The various combinations of them can generate a diversity of paintings.

Number of Strokes. We next investigate the impact of the number of strokes. In particular, the canvas is divided into $h \times w$ grids. We then generate images with various number of strokes in each grid. Fig. 12 depicts the paintings generated by different numbers of strokes with the same FEN. The small image denotes the reference image while the big images are generated by 2, 5, 8 and 10 strokes bundled in one grid. We observe that the image generated by 2 strokes looks colorful and artfully creative though it also loses many content details. The increment of strokes (e.g., 8 strokes or 10 strokes) leads to an exquisite image, which looks much closer to the reference image than the image generated by small number of strokes (e.g., 2 strokes or 5 strokes). In summary, the adjustable number of strokes provides the users with an alternative choice.

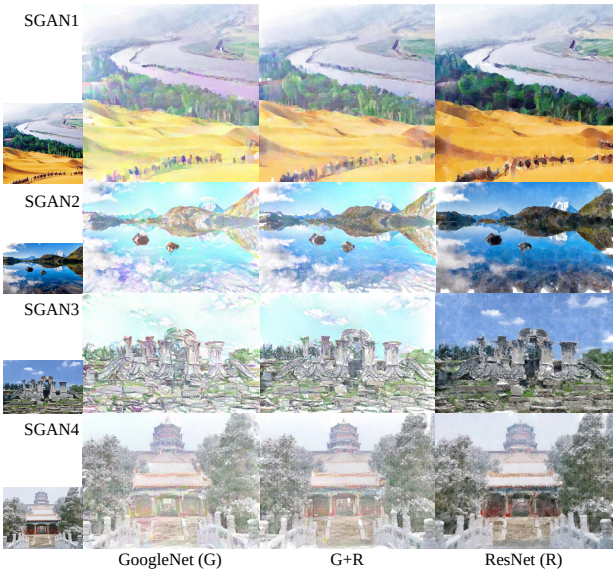

Figure 11. Paintings generated by different feature-extracting networks (FENs) and different styles of strokes. Each row consists of one stroke style of images generated by GoogleNet (G), GoogleNet+ResNet (G+R), ResNet (R) from the 2nd to the 4th column, respectively. The reference image is also placed in the 1st column in each group.

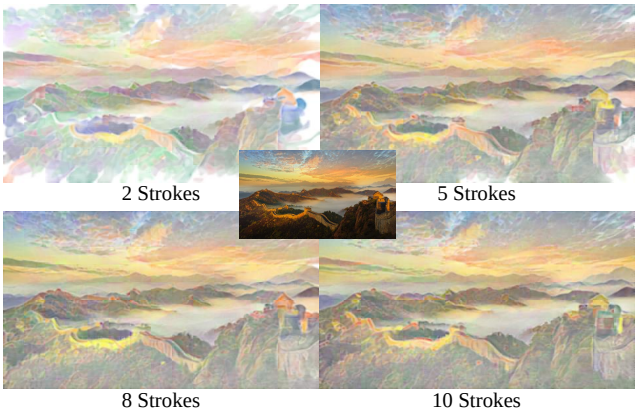

Figure 12. Paintings generated with different numbers of strokes in one grid. The small image in the center is the reference image while images generated by 2, 5, 8, 10 strokes are placed in a clockwise direction.

5. Conclusion and future work

In this paper, we present a stroke-based image rendering approach to mimic the human painting process and generate different styles of paintings. In particular, we design Stroke-GAN for the painter to generate different styles of strokes. Meanwhile, we model the painting process as an stoke-state-optimization process, which can be optimized by deep convolutional neural networks. Our artful painter can generate different styles of paintings in a coarse-to-fine fashion like human painters. The user studies of the images

generated by our painter and the state-of-the-art learning-based methods demonstrate that our painter won the most votes for the closeness to the pastel paintings and oil paintings. Moreover, the images generated by our painter also preserve more content details than existing methods.

A deep learning algorithm and Stroke-GAN are proposed to decompose the reference image into finite grids for a sequence strokes rendering to achieve the stroke-by-stroke effect. We design Stroke-GAN to generate style strokes by learning stroke datasets. Our Stroke-GAN can learn any style strokes endowing the painting agent some creativity and flexibility. Though our generated paintings are not so well as the masters' artworks, we make an important step for learning-based AI painting with creative and flexible art styles. Meanwhile, there are much worthy work to improve the quality of the artworks, and some other art styles not mentioned in this paper are waiting to uncover their mystery veil. In the future, we can combine the advantages of conventional stroke-based methods with the learning-based method to improve the painting quality. On the other hand, we will develop new style transfer methods in a stroke-by-stroke manner to rich the art styles of AI paintings.

References

[1] W. Chu and Y. Wu. Image style classification based on learnt deep correlation features. *IEEE Transactions on Multimedia*, 20(9):2491–2502, 2018. 3

[2] O. Deussen, S. Hiller, C. V. Overveld, and T. Strothotte. Floating points: A method for computing stipple drawings. *Computer Graphics Forum*, 19, 2000. 2

[3] O. Deussen and T. Isenberg. Halftoning and stippling. In *Image and Video-Based Artistic Stylisation*, volume 42, pages 45–61, 2013. 2

[4] O. Deussen and T. Strothotte. Computer-generated pen-and-ink illustration of trees. In *ACM SIGGRAPH*, page 1318, 2000. 2

[5] D. L. Donoho. De-noising by soft-thresholding. *IEEE Transactions on Information Theory*, 41(3):613–627, 1995. 5

[6] T. Dutta, A. Singh, and S. Biswas. StyleGuide: Zero-shot sketch-based image retrieval using style-guided image generation. *IEEE Transactions on Multimedia*, pages 1–10, 2020. 3

[7] Y. Ganin, T. Kulkarni, I. Babuschkin, S. M. A. Eslami, and O. Vinyals. Synthesizing programs for images using reinforced adversarial learning. In J. Dy and A. Krause, editors, *International Conference on Machine Learning*, volume 80, pages 1666–1675, 2018. 2

[8] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *ArXiv*, abs/1508.06576, 2015. 3

[9] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, LAS VEGAS, June 2016. 1

[10] D. Ha and D. Eck. A neural representation of sketch drawings. In *International Conference on Learning Representations*, pages 1–16, 2018. 2, 3

[11] P. Haeberli. Paint by numbers: Abstract image representations. *ACM SIGGRAPH Computer Graphics*, 24(4):207214, 1990. 1, 2

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 8

[13] A. Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In *ACM SIGGRAPH*, page 453460, 1998. 1, 2

[14] A. Hertzmann. A survey of stroke-based rendering. *IEEE Computer Graphics and Applications*, 23:70–81, 2003. 2

[15] H.-Z. Huang, S.-H. Zhang, R. R. Martin, and S.-M. Hu. Learning natural colors for image recoloring. *Computer Graphics Forum*, 33(7):299–308, 2014. 10

[16] Z. Huang, W. Heng, and S. Zhou. Learning to paint with model-based deep reinforcement learning. In *IEEE International Conference on Computer Vision*, pages 8708–8717, 2019. 2, 3, 4, 9, 10, 11, 12

[17] B. Jia, C. Fang, J. Brandt, B. Kim, and D. Manocha. PaintBot: A reinforcement learning approach for natural media painting. *ArXiv*, abs/1904.02201, 2019. 3

[18] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song. Neural style transfer: A review. *IEEE Transactions on Visualization and Computer Graphics*, 26(11):3365–3385, 2020. 3

[19] R. Justin. *Elements of Art: Interpreting Meaning Through the Language of Visual Cues*. PhD thesis, Stony Brook University, 2018. Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2020-12-08. 4

[20] H. Lee, S. Seo, S. Ryoo, K. Ahn, and K. Yoon. A multi-level depiction method for painterly rendering based on visual perception cue. *Multimedia Tools and Applications*, 64(2):277–292, 2013. 1, 2

[21] T. M. Liddell and J. K. Kruschke. Analyzing ordinal data with metric models: What could possibly go wrong? *Journal of Experimental Social Psychology*, 79:328–348, 2018. 11, 12

[22] S. Liu, T. Lin, D. He, F. Li, R. Deng, X. Li, E. Ding, and H. Wang. Paint transformer: Feed forward neural painting with stroke prediction. In *IEEE International Conference on Computer Vision*, 2021. 3, 9, 10, 11, 12

[23] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *IEEE International Conference on Computer Vision*, pages 3730–3738, 2015. 8, 9, 11

[24] J. F. J. Mellor, E. Park, Y. Ganin, I. Babuschkin, T. Kulkarni, D. Rosenbaum, A. Ballard, T. Weber, O. Vinyals, and S. Eslami. Unsupervised doodling and painting with improved SPIRAL. In *Neural Information Processing Systems Workshops*, 2019. 2

[25] R. Nakano. Neural painters: A learned differentiable constraint for generating brushstroke paintings. In *Neural Information Processing Systems Workshops*, 2019. 2, 3, 4, 8, 9, 10, 11, 12

[26] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, pages 1–16, 2016. 4

[27] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and F.-F. Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015. 8, 9, 11

[28] J. Song, K. Pang, Y.-Z. Song, T. Xiang, and T. M. Hospedales. Learning to sketch with shortcut cycle consistency. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 801–810, 2018. 2

[29] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, June 2015. 8

[30] Z. Tong, X. Chen, B. Ni, and X. Wang. Sketch generation with drawing process guided by vector flow and grayscale. In *AAAI Conference on Artificial Intelligence*, volume 53, pages 609–616, May 2021. 10

[31] L. Wang, Z. Wang, X. Yang, S.-M. Hu, and J. Zhang. Photographic style transfer. *The Visual Computer*, 36(2):317331, 2020. 1

[32] B. Wilson and K. Ma. Rendering complexity in computer-generated pen-and-ink illustrations. In *International Symposium on Non-Photorealistic Animation and Rendering*, page 129137, 2004. 2

[33] N. Xie, H. Hachiya, and M. Sugiyama. Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting. *IEICE Transactions on Information and Systems*, E96.D(5):1134–1144, 2013. 2

[34] M. Xu, H. Su, Y. Li, X. Li, J. Liao, J. Niu, P. Lv, and B. Zhou. Stylized aesthetic QR code. *IEEE Transactions on Multimedia*, 21(8):1960–1970, 2019. 3

[35] Y. Zhao, B. Deng, J. Huang, H. Lu, and X.-S. Hua. Stylized adversarial autoencoder for image generation. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, page 244251, New York, NY, USA, 2017. Association for Computing Machinery. 1

[36] N. Zheng, Y. Jiang, and D. jiang Huang. StrokeNet: A neural painting environment. In *International Conference on Learning Representations*, 2019. 2, 3

[37] W.-Y. Zhou, G.-W. Yang, and S.-M. Hu. Jittor-gan: A fast-training generative adversarial network model zoo based on jittor. *Computational Visual Media*, 7(1):153–157, 2021. 1

[38] Z. Zou, T. Shi, S. Qiu, Y. Yuan, and Z. Shi. Stylized neural painting. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 15689–15698, Virtual, June 2021. 3, 9, 10, 11, 12