A Multi-Layered Framework for color pastel painting

Heekyung Yang[1] and Kyungha Min[2]
[1] Dept. of Computer Science, Graduate School, Sangmyung Univ.,
Hongji-dong, Jongro-gu, Seoul, Korea
[e-mail: yhk775206@naver.com]
[2] Dept. of Computer Science, Sangmyung Univ.,
Hongji-dong, Jongro-gu, Seoul, Korea
[e-mail: rminkh@smu.ac.kr]
*Corresponding author: Kyungha Min

Received November 3, 2016; revised February 27, 2017; accepted March 14, 2017; published June 30, 2017

Abstract

We present a computerized framework for producing color pastel painting from the visual information extracted from a photograph. To express color pastel painting, we propose a multi-layered framework where each layer possesses pastel stroke patterns of different colors. The stroke patterns in the separate layers are merged by a rendering equation based on a participating media rendering scheme. To produce the stroke patterns in each layer, we review the physical properties of pastels and the mechanism of a convolution framework, which is the most widely used scheme to simulate stick-shaped media such as pencils. We devise the following computational models to extend the convolution framework to produce pastel strokes: a bold noise model, which mimics heavy and clustered deposition of pigment, and a thick convolution filter model, which produces various pastel stroke patterns. We also design a stochastic color coordination scheme to mimic pastel artists' color expression and to separate strokes in different layers. To demonstrate the soundness of approach, we conduct several experiments using the models and compare the results with existing works or real pastel paintings. We present the results for several pastel paintings to demonstrate the excellent performance of our framework.

Keywords: Non-photorealistic rendering, pastel media simulation, multi-layered approach

1. Introduction

Pastels have been used by many artists to depict shapes and to express colors in various styles since the sixteenth century. Impressionist painter Edgar Degas, for example, painted many outstanding pastel artworks including the famous Ballet Dancer series in 1878. The rich expression of color and the diverse depiction of shape that are possible with pastel make many people love pastel artworks. In the field of computer graphics, many researchers have examined various artistic media, including pencils, oil-paints, and watercolor paints, and have presented convincing methods for simulating their effects. However, our review of the literature in computer graphcs has not identified any serious research on pastels.

Through the survey of pastel artwork textbooks and the interview with pastel artists, we define that the most distinguished characteristic of pastel painting is the color effects expressed by stroke overlaying. Therefore, we design a multiple layered framework as shown in **Fig. 1**. In our framework, stroke patterns of different colors are produced on each layer, which is merged through our rendering equation to the final result. To produce stroke patterns, we extend the convolution-based framework for pencil drawing to pastel strokes. For this, we analyze pastel media and design computational models to mimic the physics of the media.

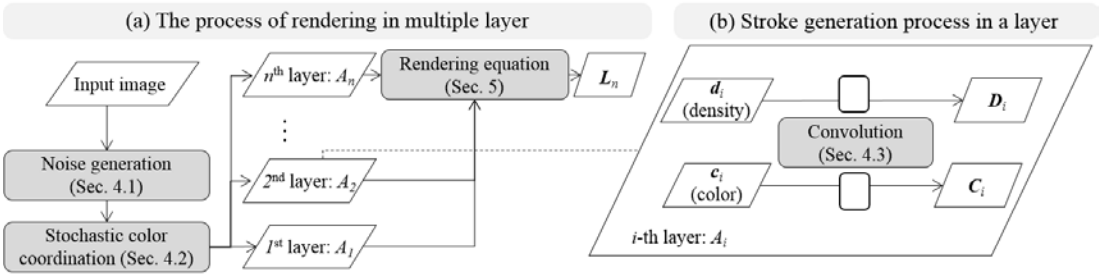

Fig. 1. The structure of our framework: (a) A multi-layer approach for producing a final result ($\boldsymbol{L}_n$), (b) The process for a single layer.

1.1 Pastel Media Analysis

By definition, pastels are stick-shaped artistic media, composed of powdered pigment and a binder. Pastels are distinguished from other stick-shaped media such as pencils in two points: their wide tips, which results in large areas of contact with paper surfaces (**Fig. 2** (a) (1)), and their relatively low proportion of binder (**Fig. 2** (a) (2)). The pigments that determine colors are similar for most artistic media, such as color pencils, pastels, crayons, and oil paints. The low proportion of binder in pastel makes pastel pigments easy to disperse. Furthermore, the tip of a pastel wears away faster than that of other stick-shaped media. Because of the easily dispersed pigment from the wide tip, pastel leaves a richer amount of pigment on a unit area of the surface compared to other stick-shaped media (**Fig. 2** (a) (3)).

The wide tip and rich amount of pigment produce painterly and visible stroke patterns, one of the most salient and distinguishable characteristics of pastel painting (**Fig. 2** (a) (4)). An individual pastel stroke is thicker and more opaque than the strokes produced by other stick-shaped media such as pencils. Pastel artists place these painterly strokes to depict the salient shape of objects in various styles. Many pastel artists place opaque strokes over the strokes that have already been drawn to express stroke overlaying effects, such as impasto,

scumbling and scripting (**Fig. 2** (a) (5)). By mixing or overlaying strokes, artists produce rich color expressions from a limited set of pastel colors. Furthermore, heavy deposition of low-viscosity pigment produces a smearing effect, wherein artists express smoothly varying tone or color in pastel paintings by rubbing the pigments deposited on the surface with their hands or soft tissues (**Fig. 2** (a) (6)).

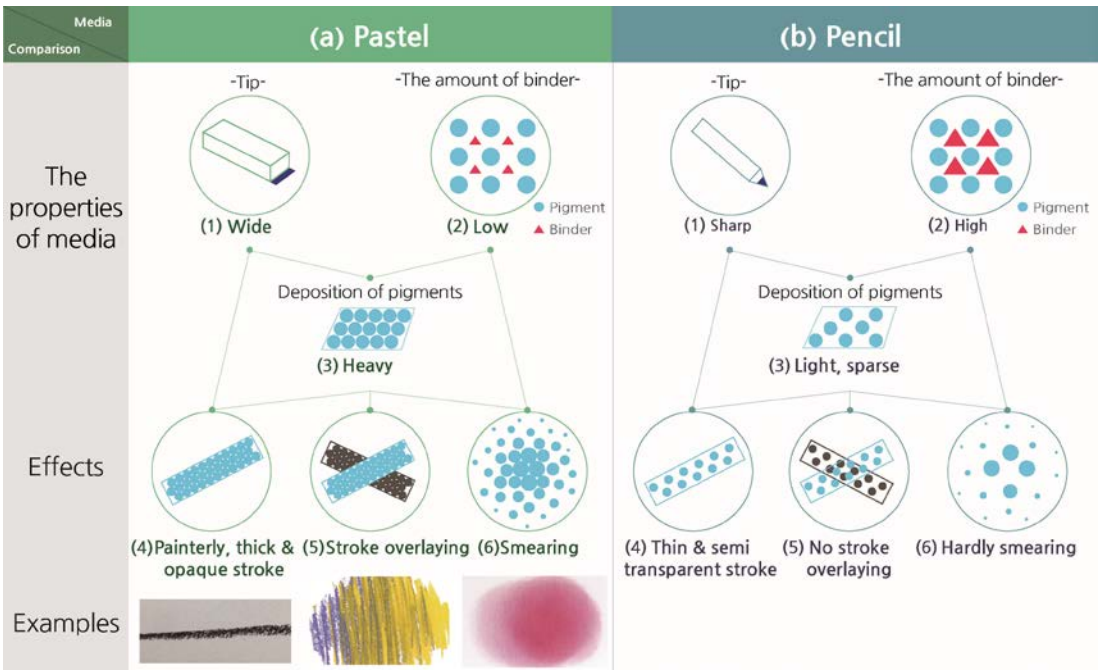

Fig. 2. Comparisons of pastel and pencil, both stick-shaped media in terms of their physical characteristics and effects. Note that the effects of the media are the products of the characteristics of the media. We illustrate the effects of pastel with samples of real pastel strokes.

1.2 Our Approach

To establish a convincing pastel simulation framework, we design a multi-layered framework, which is motivated from pastel artwork where artists places strokes over strokes to express various color effects. We also devise computational models to express thick and opaque stroke patterns on each layer. We employ a rendering scheme to visualize stroke patterns on a single layer and overlapped stroke patterns in multiple layers.

The framework is as follows. In each layer, properly integrating heavily and densely deposited pigment produces painterly and opaque pastel stroke patterns. To generate heavy and dense pigment deposition, we sample a pixel in an input image and capture a certain area centered at the sampled pixel. Then, we deposit noise of a similar color and density at the pixels covered by the area. Note that the color and density are stored at separate channels in the same layer. Our computational model, referred to as bold noise, defines the area in terms of information aspects such as position, shape and orientation. To express various stroke patterns using the bold noise model, we use a novel convolution filter that can produce a range of effects from salient stroke patterns to smooth smearing patterns. This filter is referred as a thick convolution filter, because it integrates noise not only on a linear kernel but also in an areal kernel. Pastel artists express various colors by overlaying strokes of different colors. They sometimes choose colors of different hues or saturations to express variations of their

target color. To mimic pastel artists' color expression, we devise a stochastic color coordination scheme where a set of candidate colors derived from a sampled color is produced and one of them is selected stochastically. Therefore, the same color sampled from different pixels can be expressed by the strokes of different colors, which is selected from the set of candidate colors. After selecting a color, we determine the layer where the stroke will be placed according to the value of the color. After convoluting the noise at each layer, we merge the stroke patterns at each layer by presenting a rendering scheme, which is designed based on a participating media rendering scheme [1].

1.3 Our Framework

The data structure of our pastel simulation framework is composed of the following variables (See **Fig. 1**). Since these variables are stored in each pixel of an image, the data structure for these variables are two-dimensional arrays:

- Stroke direction ($\boldsymbol{S}(\mathbf{x}) \in \Re$), The stroke direction at pixel $\mathbf{x}$;
- Density of noise ($d_i(\mathbf{x}) \in \Re$), The density of the noise at pixel $\mathbf{x}$ of the i-th layer;
- Color of noise ($\boldsymbol{c}_i(\mathbf{x}) \in \Re^3$), The color of the noise at pixel $\mathbf{x}$ of the i-th layer;
- Convoluted value for a density ($D_i(\mathbf{x}) \in \Re$), The convoluted value for the density at pixel $\mathbf{x}$ of the i-th layer;
- Convoluted value for a color ($\boldsymbol{C}_i(\mathbf{x}) \in \Re^3$), The convoluted value for the color at pixel $\mathbf{x}$ of the i-th layer;
- Value of stroke pattern ($L_i(\mathbf{x}) \in \Re^3$), The value of the stroke pattern at pixel $\mathbf{x}$.

At the initial stage, we compute the stroke direction field using edge tangent flow (ETF) [2], which is a weighted averaging approach that produces a smooth flow field from an image. We build n layers at which bold noise is generated and stored at d_i and $\boldsymbol{c}_i$, according to the scheme in Section 4.1 and 4.2. In the convolution stage, we apply the convolution suggested described in Section 4.3 and store the values at D_i and C_i. For the final pastel stroke patterns, we apply the rendering formula suggested in Section 5 to build L_n.

The contributions of this paper are summarized as follows:

1. A multi-layered framework to express color pastel painting.
2. A rendering equation that merges the strokes in different layers.
3. A convolution-based pastel stroke generation scheme from media analysis.

2. Related Work

Since pastel and pencil are a kind of graphite media composed of pigment and binder, the simulation techniques for the media have a common technical background. We classify the related research according to their background technology as follows: physical model that simulates the physics under the media, approximation model that presents a computational model to simulate the media, texture-based approach that applies stroke textures to mimic the media and example-based approach that employs texture transfer and neural networks.

2.1 Physical Model

Physical models for pencil, pastel and crayon have been developed in previous studies. Takagi et al. [3] observed the microstructure of a paper surface and designed a volumetric model for simulating the distribution of pigment over the surface. Using their model, they simulated colored pencil drawing and several effects, such as watering and erasing. Sousa and Buchanan [4] devised a simulation model that considers the effects of factors such as pencil hardness, pencil point size, the drawing paper and pencil paper interaction on the physics of pencil drawing. Their realistic pencil strokes reflect realistic consideration of pressure, a polygonal tip, pressure distribution, finger distance, pencil slanting and wrist and arm movement. In addition, they improved their work to simulate the action of blenders and erasers [5]. Murakami et al. [6] presented a physical model that simulates graphite media such as charcoal, crayon, and pencil. They constructed a height map representing the paper surface by scanning real paper texture. They expressed the stroke as a smooth curve of varying width. They then estimate the distribution of pigment over the paper surface along the stroke path. In their simulation of wax crayon medium, Rudolf et al. [7] expressed paper as a height field and crayon as a 2D mask. They designed a model that computes the amount of wax deposited on a paper surface. The deposited wax can be smeared or redeposited. They applied the Kubelka-Munk method [8] to render the crayon wax on the paper surface. Kubelka-Munk method is a widely used method based on subtractive model for representing pigments and their mixing.

2.2 Approximation Model

The most widely used approximation model for simulating artistic media is line integral convolution (LIC) [9], which was originally devised to visualize flow embedded in an image. Methods using LIC concentrate on simulating black and colored pencil drawing. Mao et al. [10] pioneered the application of LIC to pencil drawing. Their method distributes salt-and-pepper noise to reflect the tone of an image and applies LIC to produce stroke patterns. Yamamoto et al. [11] improved on this scheme by segmenting an image into several regions, each of which is then rendered using strokes in identical directions. Yamamoto et al. [12] extended the LIC scheme to simulate colored pencils. Their method requires users to select two dominant colors for each segmented region and to apply the LIC scheme to create two monochrome results. These results are combined into a colored pencil drawing using the Kubelka-Munk method. These schemes enable only uniform stroke directions on each segmented region. Yang and Min [13] combined LIC with edge tangent flow (ETF) to produce smooth monochrome pencil drawing effects. The strokes close to salient features of objects in the image follow the ETF directions, while the other strokes follow uniform stroke directions. This approach was improved to enable a stylized scheme [14] for producing colored pencil drawing effects. Three different categories of stroke directions are applied to the segmented image to produce realistic pencil stroke patterns. In addition, a scheme was presented that customizes the color to user-selected palettes. Kwon et al. [15] presented an LIC-based color pencil drawing scheme on three dimensional meshes. Their method generates noise on the triangles of the mesh and projects them into a 2D image space while preserving a consistent density. The noise in the 2D space is then integrated to produce pencil drawing effects. Hata et al. [16] presented a saliency map-based LIC scheme that emphasizes important regions of an image and eliminates insignificant ones. The saliency map is a computational model that predicts the area of visual focus within an image. A detail control algorithm using a multi

resolutional pyramid is used to adapt the rendering parameters that control the density, orientation and width of pencil strokes.

2.3 Texture-Based Approach

Texture-based approaches are used to produce maps of various tones by overlapping stroke textures captured from real strokes. The effects produced cover as wide range of styles, such as hatching, monochrome pencil, colored pencil, and pen-and-ink illustration. Matsui et al. [17] introduced a stroke-based algorithm with which strokes are placed along the offset curves of contours in an image. Lee et al. [18] rendered triangular meshes using a tonal map by generating various pencil textures and applying them in the direction of the principal curvature. Their scheme was advanced through the use of a graphics processing unit. Kim et al. [19] extended this scheme to generate line art illustrations of dynamic 3D objects with highly reflective surfaces. Lu et al. [20] presented a colored pencil drawing that combines line drawing and tonal texture. The line drawing was executed from the gradient extracted from an image, and the pencil texture was produced from the tonal map.

2.4 Example-Based Approach

Hertzmann et al. [21] presented a novel framework that applies desired effects to a target image automatically by processing a pair of images composed of an original image and a filtered image. The key idea of the framework is to extract the difference between the pair of input images using multi-scale autoregression-based texture synthesis. The extracted difference is applied to the target image to produce an effect, which is similar to the filtered image. This framework can be used to produce various effects, including blurring, embossing, texture transfer, and artistic filter. However, this scheme does not consider the structure embedded in an image, which sometimes produces undesired effects. Gatys et al. [22] presented an innovative approach to producing artistic images using a convolutional neural network. Their system employs a neural network model to analyze the content and style of arbitrary images and apply the style to an input image. Selim et al. [23] proposed a convolutional neural network in applying various artistic styles to head portraits. Unlike other example-based approaches such as that proposed by Hertzmann et al. [21], their approach requires only an input photograph and an example painting. The style of the example painting is transferred to the input photograph by imposing a set of spatial constraints that locally transfers the color distributions of the example painting using a convolutional neural network.

3. Background Theory of Convolution Framework for Stick-Shaped Media Simulation Introduction

Stick-shaped media such as pencils, charcoals, pastels, and crayons, leave pigments on paper surfaces along stroke directions as the result of abrasion between the tip of the media and the paper surface. The pigments from a narrow tip form long thin marks on the surface, which are considered the stroke pattern. A convolution-based framework mimics the drawing process of stick-shaped media in the following three steps. First, stroke directions are estimated at every pixel in an image. Second, noise, which plays the role of an individual pigment, is embedded at each pixel. The value of the noise reflects the intensity or color of the pixel at which the noise is embedded. Third, a linear convolution filter is applied and the noise values lying on the kernel of the filter are integrated to produce the value of the stroke pattern at the pixel. The result of the convolution of the noise resembles the stroke patterns in the

following reason. Because the integration ranges of the incident pixels lying in the same convolution direction differ by only two pixels, their convolution results have very similar values (See A and B in **Fig. 3** (a)). In contrast, the incident pixels not lying in the same direction have different convolution results, because their integration ranges are quite different (See A and C in **Fig. 3** (a)). Therefore, the convolution results resemble long thin stroke patterns.

Even though the conventional convolution schemes are very effective in filling a region with long thin stroke patterns, they have several limitations with respect to their ability to produce pastel strokes. They cannot control the thickness and opacity of the stroke patterns. Individual stroke marks of various shapes are not expressed either. Achievement of a smoothly varying tone or color pattern is also beyond the abilities of conventional convolution schemes.

In thick and painterly strokes, not only the incident pixels in the stroke direction but also those orthogonal to the stroke direction have similar values. Our bold noise model, which deposits noise with values similar to those of the incident pixels within its area, is effective in mimicking thick and painterly strokes. Using the bold noise model, the incident pixels in the direction orthogonal to the stroke direction and the pixels in the stroke direction have similar convolutional values (See A, B and C in **Fig. 3** (b)).

The linear stroke patterns produced by the convolution-based approach resemble the real stroke patterns drawn by stick-shaped artistic media. Unfortunately, convolution-based approaches do not mimic the smearing technique well, because smearing marks are far from linear, exhibiting isotropic gradations in tone and color. To mimic smearing marks, the noise deposited over a wide area should be integrated. A wider integration area further smoothens the integration values of the incident pixels. For this purpose, we devise a thick convolution filter whose kernel is a rectangle (See **Fig. 3** (c)). In our thick filter, the incident pixels have very similar integration areas, which guarantees smooth gradations of tone and color.

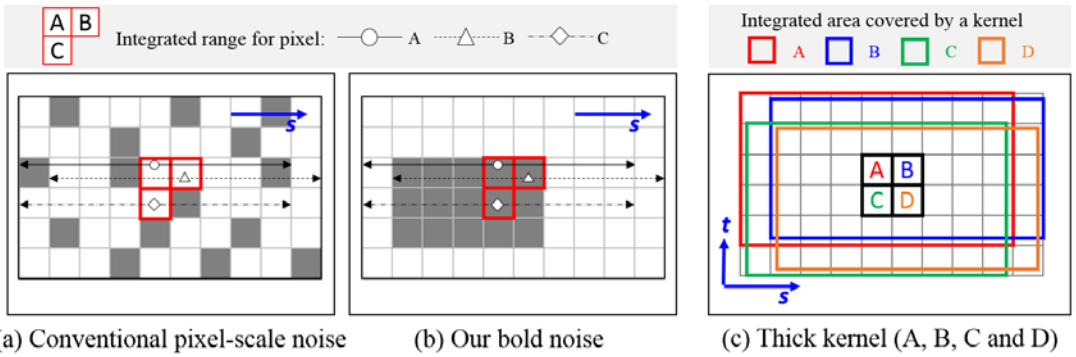

Fig. 3. The background theories: (a) The theory behind the conventional convolution scheme according to which incident pixels along the integration filter have similar values, whereas incident pixels orthogonal to the filter have quite different values; (b) The theory behind our bold noise model with which our scheme produces similar values for the incident pixels; and (c) The theory behind our thick convolution filter that results in incident pixels of similar tone.

4. Pastel Stroke Generation

Producing pastel strokes in a single layer consists of the following steps: generating heavy and clustered deposition of noise and applying a proper convolution to the noise. In generating noise, we assign proper colors to noises and deposit them in different layers.

4.1 Bold Noise

4.1.1 Definition

We define bold noise as a rectangle aligned in a specific direction. We store noise color and density at the pixels underlying the rectangle. The noise shape is defined by the following properties: the center position (**x**), length (l), width (w), orientation (α), color (**C**) and density (δ).

e (1)

These parameters are illustrated in **Fig. 4**.

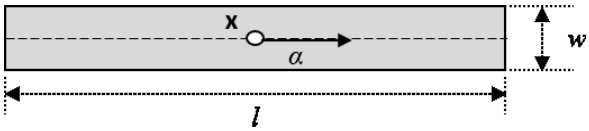

Fig. 4. The definition of a bold noise.

4.1.2 Generation

We sample the center position (**x**) of a bold noise using the dart throwing algorithm, where any prefix of generated noises preserve Poisson distribution. An efficient implementation of the algorithm requires a distance map of the generated noise sequence. The width (w) and length (l) of the noise are assigned by the user. The orientation (α) of the noise follows the stroke direction with some perturbation. We assign the color (**C**) of noise by sampling the color of the center position according to the scheme in Section 4.2. The density (δ) of the noise at the pixel is assigned to be proportional to the area overlapped by the noise.

4.1.3 Results and Comparison

Using our bold noise model, we generated a simple image with different tones and compared them with an image generated using the conventional pixel-scale noise (See **Fig. 5**). This image has thick and painterly stroke patterns that fill the region. For the purpose of this comparison, we generated noise of different densities and compared the results. We also produced an image re-rendered from an input photograph (See **Fig. 6**). By controlling the length and width of the bold noise, we were able to produce pastel stroke patterns with different styles. As we increased the width of the noise, the stroke pattern became thicker and bolder. For the sake of efficiency of the comparison, we used monochrome color for the noise.

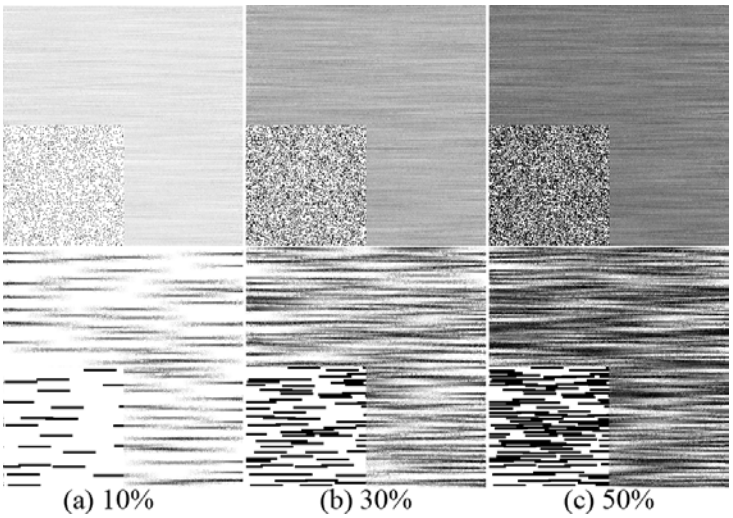

Fig. 5. The generation of bold noise of different densities: (upper row) conventional pixel-scale noise and (lower row) bold noise. The lower row shows painterly and thick stroke patterns with the increased clustering of noise.

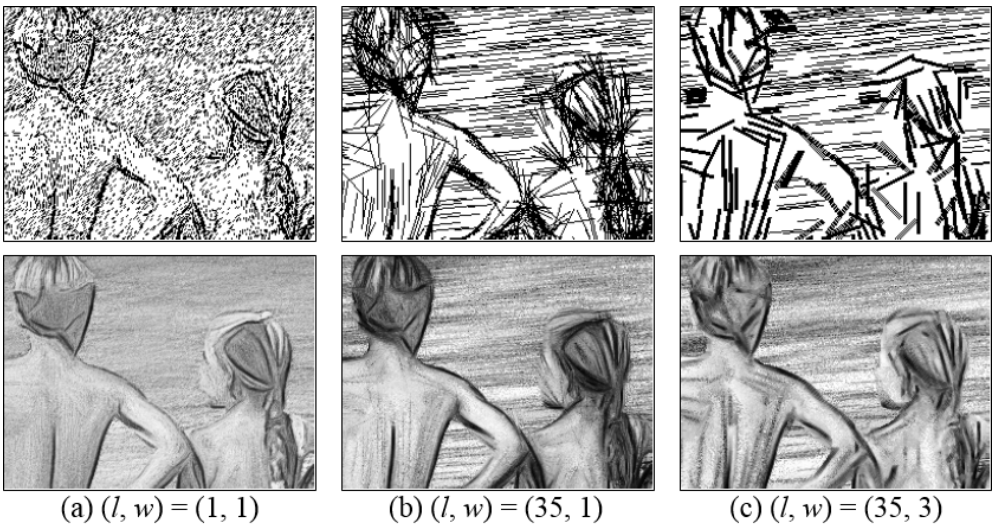

Fig. 6. Comparison of bold noise: (a) Point noise and its result, (b) Bold noise with (l, w) = (35, 1) and its result, (c) Bold noise with (l, w) = (35, 3) and its result.

4.1.4 Experiment

To demonstrate that our bold noise generates clustered pigments, we conducted the following experiment. First, we filled 50pixels using pixel-scaled point noise (**Fig. 7**. (a)) and bold noise (**Fig. 7**. (b)). Second, we counted the number of noise-embedded pixels on a linear kernel whose length is n, because the convolution-based scheme integrates noise using a linear kernel. By applying the kernel at every pixels and by counting the noise under the kernel, we were able to draw the results shown in **Fig. 7**. In the case of (a), the amount of noise in the kernel follows a normal distribution whose average is $n/2$. However, in (b), the amount of noise in the kernel is distributed almost uniformly over the range of ($0 \sim n-1$). Based on the

results of this experiment, we concluded that our bold noise model effectively produces clustered pigment deposition on a paper surface.

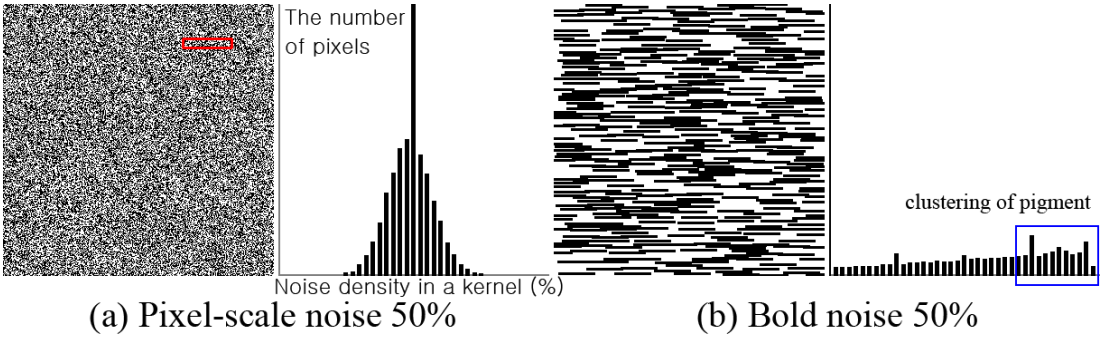

Fig. 7. Comparison of the distributions of pigments in two different types of noise (a linear kernel of n = 42 was used).

4.2 Stochastic Color Coordination

To support pastel color effects via stroke overlaying, we express the color of a stroke by varying the sample color at the center position of a noise. We cannot find a logical explanation about the variation of color through the survey of pastel artwork textbooks and the interview with pastel artists, since even pastel artists cannot explain their reason of variation. Pastel master Degas, for example, prefers the colors of higher tone than the original colors, but sometimes do not (See **Fig. 9**). Colors of higher tone produce more vivid and interesting expressions than those of darker tone. Some textbooks use the term "rhythmic expression" for the vivid and interesting colors. Some researcher pioneered the variation of a color by mixing or blending different colors. In oil-painting research, Lin et al. [24] express stroke color by combining brush strokes of warm colors and cold colors from predefined stroke dictionary. In pencil research, Yang et al. [14] approximate a color by mixing two colors sampled from Faber-Castel color set. Unfortunately, Lin et al.'s work requires well-defined color mixing stroke dictionary and Yang et al.'s work does not support color overlaying schemes.

4.2.1 Color Variation

We present a stochastic color coordination scheme that varies colors from the sampled color without predefined dataset. A color component C'_i of a color $\mathbf{C} = (C_1, C_2, C_3)$ is reallocated as C_i as follows:

$$C'_i = \begin{cases} C_i, \ if\ \gamma > C_i \\ 1, if\ otherwise \end{cases} \tag{2}$$

where γ is a random number in (0 ~ 1). The expectation of C'_i is $P(\gamma > C_i)\times C_i + P(\gamma < C_i)\times 1 = (1-C_i)\times C_i + C_i \times 1 = 2\,C_i - C^2_i > C_i$, which means that C'_i has higher tone than C_i.

Applying this scheme for each channel of a color, an interval (0, 1) is divided into four sub-intervals. If we assume $r > g > b$, then, the sub-intervals become $(0, b)$, (b, g), (g, r) and $(r, 1)$. In each sub-interval, the noise generated at a pixel whose color $\mathbf{C} = (r, g, b)$ has the following four colors:

$$C'_i = \begin{cases} (1,1,1),\ if\ \gamma \in (0,b) \\ (1,1,b),\ if\ \gamma \in (b,g) \\ (1,g,b),\ if\ \gamma \in (g,r) \\ (r,g,b),\ if\ \gamma \in (r,1) \end{cases} \tag{3}$$

The set of candidate colors is composed of these four colors, and random number γ determines which one to select. The color of noise can be the original color (r, g, b) or extraneous colors (1, 1, b) and (1, g, b) or white (1, 1, 1). Example color sets from sampled colors are illustrated in **Fig. 8**. It is interesting to note that in **Fig. 8** (a), (1.0, 0.48, 0.1), which is very close to red, is generated. The extraneous colors have higher tones than the original color. By reversing the order in the above equation, colors of darker tones can be generated. After selecting a color from the candidate set, we determine which layer to deposit the noise of the selected color. We consider the saturation of the color.

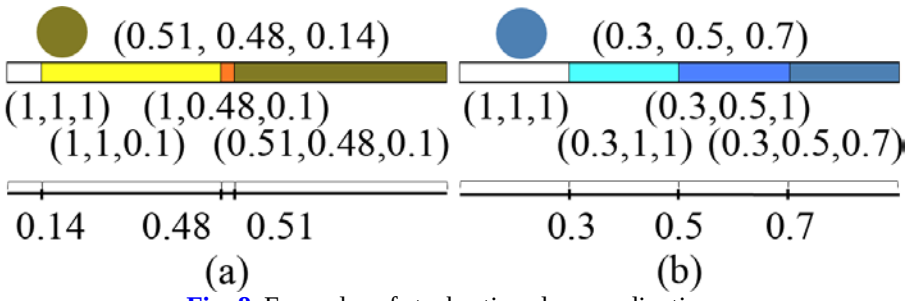

Fig. 8. Examples of stochastic color coordination:
Four candidate colors are generated from a sample color.

4.2.2 Comparison

Fig. 9 shows four different results. **Fig. 9** (a) shows the result without stochastic color coordination scheme. We used only sampled colors for noise color. **Fig. 9** (b) ~ (d) show the results using stochastic color coordination scheme. In **Fig. 9** (b), noise is deposited in a single layer, and in **Fig. 9** (c) and (d), noise is deposited in two layers according to their saturations. Among them, the noise whose color has higher saturations is deposited in the upper layer in the result in (c), while the noise of lower saturations in the upper layer in (d).

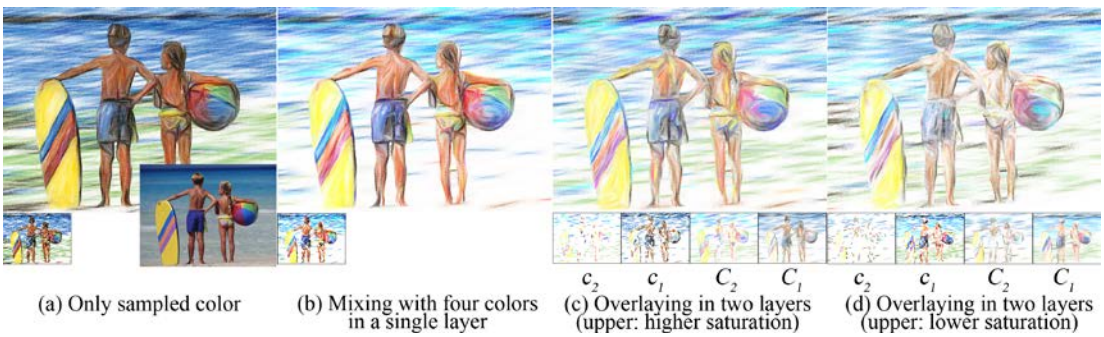

Fig. 9. Different results from different colors and layers.

4.3 Thick Convolution Filter

4.3.1 Definition

Our thick convolution filter is defined as follows:

$$C(x) = \frac{1}{V}\int_{-\tau}^{\tau} G_{\sigma_1}(\theta_i - \theta)\int_{-S}^{S}\int_{-T}^{T} G_{\sigma_2}(|I(x) - I(y)| \cdot N_y)dtdsd\theta \tag{4}$$

where $I(\mathbf{x})$ and $I(\mathbf{y})$ denotes the intensity at pixel $\mathbf{x}$ and $\mathbf{y}$, respectively. $N_\mathbf{y}$ denotes the color ($\mathbf{c}_i(\mathbf{y})$) or density ($d_i(\mathbf{y})$) at $\mathbf{y}$. Note that the direction along ($-S$, S) is orthogonal to the direction along ($-T$, T). Our thick convolution filter integrates noise in three stages (See **Fig. 10**). The first-stage integration, $\int dt$, integrates the noise embedded in the pixels orthogonal to the axis of the kernel at each sampled point (s) along the axis of the kernel. The second-stage integration $\int ds$ integrates the integrated values stored at s along the axis. Finally, we vary the integration direction in ($-\tau$, τ) and perform the third integration $\int d\theta$ to express the perturbed stroke patterns. Note that our filter is a bilateral filter, which is expressed by two Gaussian filters $G_{\theta 1}$ and $G_{\theta 2}$. A smaller θ_1 increases the rough stroke patterns, and a larger θ_2 increases the blending of color at the boundary. n is a normalization term. We apply this filter to each d_i and c_i and produce D_i and $\mathbf{C}_i$, respectively.

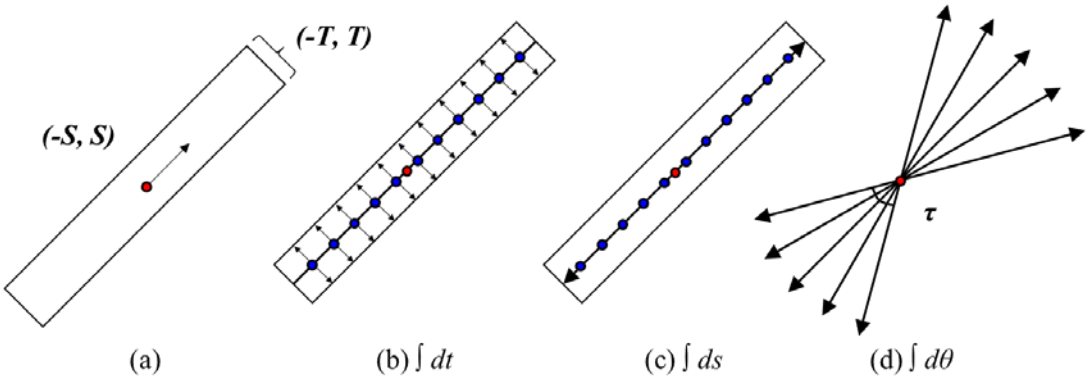

Fig. 10. Convolution steps: (a) Integration range (b) Integration along T (c) Integration along S (d) Integration along τ.

4.3.2 Results and Comparison

We present two results obtained using our thick convolution filter. One result is a hatching pattern produced with increased T and angle (τ). As **Fig. 11**, τ affects the perturbation of the stroke patterns, and T influences the smoothness of the strokes. With (τ, T) = (5, 30), we achieved very smooth results, similar to smearing. The other result mimics the real pastel smearing effect. **Fig. 12** (c) shows a real pastel smearing image captured from a textbook. To mimic this image, we generated noise of a similar color, as shown in **Fig. 12** (a), and we used our thick filter to produce the result shown in **Fig. 12** (b). A comparison of **Fig. 12** (b) and (c) demonstrates that our scheme successfully mimic pastel smearing effects.

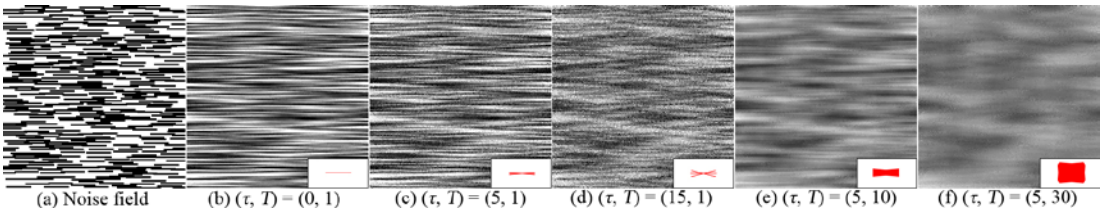

Fig. 11. Comparison of results for different values of τ and T.

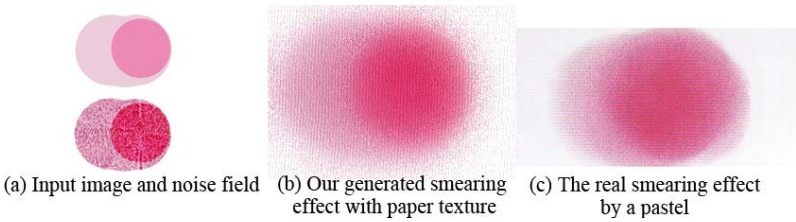

Fig. 12. Comparison with real pastel image.

4.3.3 Experiments

Fig. 13 shows that the results of an experiment conducted to produce smearing effects and, compare them to real pastel smearing effects. **Fig. 13** (a) shows the real pastel smearing pattern, and **Fig. 13** (b) shows our result. **Fig. 13** (c) and (d) show the pastel strokes simulated by the conventional thin filter. The graphs in the lower row show the measured smoothness of the results. At each pixel of the results, we applied a 5×5 mask and computed the maximum and minimum among the 25 pixels. The absolute values of the differences between the maximum and minimum are plotted on the x-axis of the graph and are distributed within the range of (0 ~ 255). The number of pixels with identical differences is plotted on the y-axis. The difference becomes zero for an image of identical tone, and becomes close to zero for an image of smooth tone. The results obtained with our thick filter (**Fig. 13** (b)) are very similar to the real pastel smearing pattern in terms of their smoothness distribution, shown in **Fig. 13** (a), and the results from the thin filter, shown in **Fig. 13** (c) and **Fig. 13** (d) have quite different smoothness distributions.

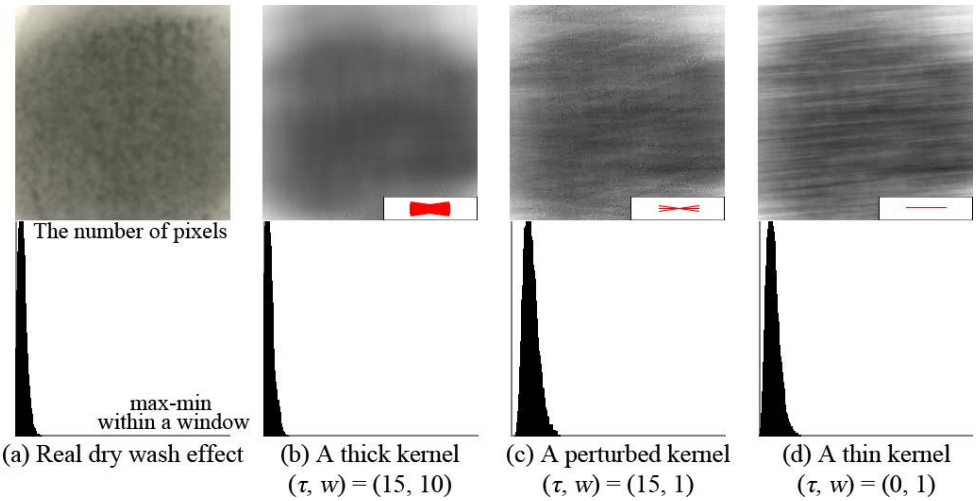

Fig. 13. Experiment and analysis: Upper layer shows stroke patterns (captured from real image, smeared pattern from our thick filter, linear stroke patterns from conventional filters) and lower layer shows their distributions.

5. Rendering Pastel Strokes

Pastel artists mix or blend stroke patterns to present rich color expression. The strokes painted in a single layer are mixed to produce strokes of new colors and the strokes painted in different layers are overlapped and blended to present strokes of transparently blended colors. Sometimes artists separate the strokes using a fixative, which is a chemical liquid that stabilizes or preserves pastel paintings. Our rendering scheme should support both mixing and blending.

Yamamoto et al. [12] and Lee et al. [18] overlapped uniformly hatched monochrome stroke patterns to present mixed hatching patterns, and Yamamoto et al. [11] presented a scheme that mixes two user-selected representative colors of a region. However, overlapping pencil strokes of arbitrary colors to present various stroke overlaying effects has not been studied yet. In oil painting research, alpha blending [25] or ray tracing [26] is used for rendering overlapped color strokes.

To devise our multilayered approach, we study physical properties of pastel media and observe pastel drawing techniques. Various blending or mixing color strokes is one of the most fascinating characteristic of pastel drawing. Our scheme mimics pastel drawing technique by allowing blending or mixing color strokes in various styles. To implement stroke blending, we separate the color and the density of a noise and design a rendering scheme for pastel strokes based on participating media rendering scheme.

In devising a rendering equation, we assume that the incoming radiance only reflects or penetrates on the pigment deposited from pastel media. Other reactions such as refraction are not considered. We also assume that D_i, the density of the stroke patterns at i-th layer determines the reflection or penetration. From these assumptions, we propose the following basic strategies for building a rendering equation.

(1) The incoming radiance at a pixel $\mathbf{x}$ either reflects or penetrates. The ratio of reflection to penetration corresponds to $D_i(\mathbf{x})$, the convoluted density at x in the i-th layer.
(2) The penetrating radiance collides at the next layer and either reflects or penetrates.
(3) The radiance reflected from the lower layer also collides with the upper layer and either penetrates or reflects.
(4) The total sum of the reflected radiance from all layers becomes the final result of our rendering equation.

5.1 Rendering Strokes in a Single Layer: Mixing

The color from a stroke pattern at a pixel $\mathbf{x}$ in a single layer is determined by the radiance reflected at the layer. Because the reflected radiance corresponds to $D_i(\mathbf{x})$, the color is estimated as $D_i(\mathbf{x}) \cdot C_i(\mathbf{x})$, the color of the pigment at $\mathbf{x}$.

5.2 Rendering Strokes in Multiple Layers: Blending

We apply a similar approach for rendering stroke patterns in multiple layers. In **Fig. 14**, we illustrate the process of building rendering equations for a stroke pattern for three layers. The rendering equation summarizes the contributions from the three layers to determine the stroke pattern. Among the incoming radiance, the reflected radiance at A_3, the top layer, is estimated as D_3 and the penetrating radiance as $1-D_3$. Therefore, A_3 contributes $D_3 \cdot C_3$ to the stroke pattern. Among the penetrating radiance of A_1, A_2, the next layer, reflects D_2 and penetrates $1-D_2$. Therefore, the total amount of reflected radiance at A_2 layer is $(1-D_3) \cdot D_2$, and the total

amount of penetration is $(1-D_3)\cdot(1-D_2)$. Because the reflected radiance at A_2 penetrates A_1 again, the contribution of A_2 to stroke pattern becomes $(1-D_3)^2 \cdot D_2 \cdot C_2$. Similarly, D_1 of the penetrating radiance at A_2 reflect at A_1, the bottom layer, and $(1-D_1)$ of them penetrate at A_1. Therefore, the reflected radiance from A_1 becomes $(1-D_3) \cdot (1-D_2) \cdot D_1$. After penetrating A_2 and A_3 again, the contribution of this radiance to stroke pattern is $(1-D_3)^2 \cdot (1-D_2)^2 \cdot D_1 \cdot C_1$. The complete stroke pattern is estimated as $D_3 \cdot C_3 + (1-D_3)^2 \cdot D_2 \cdot C_2 + (1-D_3)^2 \cdot (1-D_2)^2 \cdot D_1 \cdot C_1$. After considering these equations, we build the following recursive rendering equation for stroke patterns.

$$L_n = D_n \cdot C_n + (1-D_n)^2 \cdot L_{n-1}, where\, L_1 = D_1 \cdot C_1. \tag{5}$$

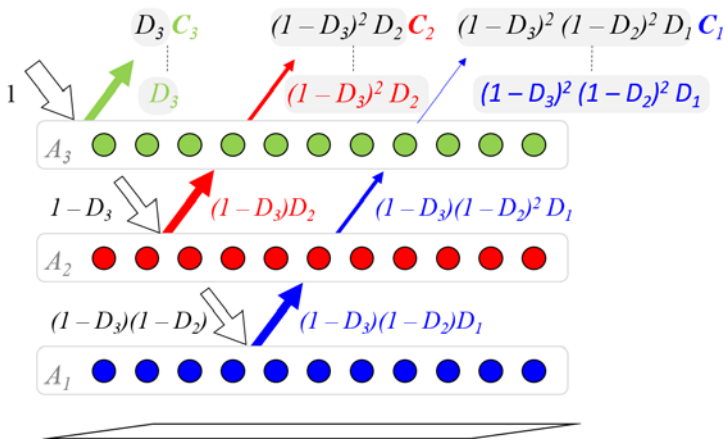

Fig. 14. Rendering strokes in 3 layers.

5.3 Results and Comparison

We tested our rendering equation on both mixing strokes in a single layer and overlaying strokes in multiple layers. As illustrated in **Fig. 15**, the red and yellow stroke patterns produce orange strokes for mixed strokes in a single layer and transparently blended strokes for overlapping strokes in multiple layers. We also implemented an alpha blending-based stroke overlaying scheme and compared it with ours. As illustrated in **Fig. 16**, the alpha blended stroke overlaying produced rather mixed color strokes. Our rendering equation, however, produces more pastel like stroke overlaying effects according to the stroke patterns.

Our scheme can support more than three layers for pastel painting. Practically, however, using more than three is not observed in most professional pastel artworks. We apply two-layer implementation, which reveals how the result depends on overlaying strokes of different densities.

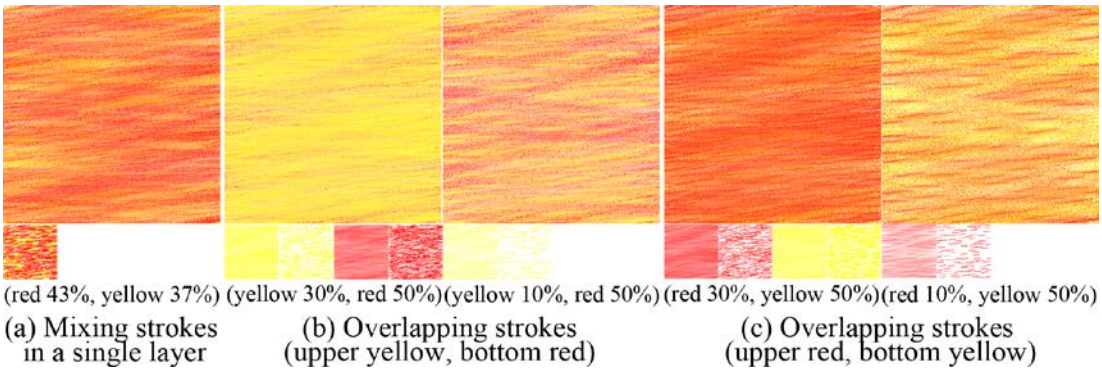

Fig. 15. Multiple layering methods: mixing and overlapping.

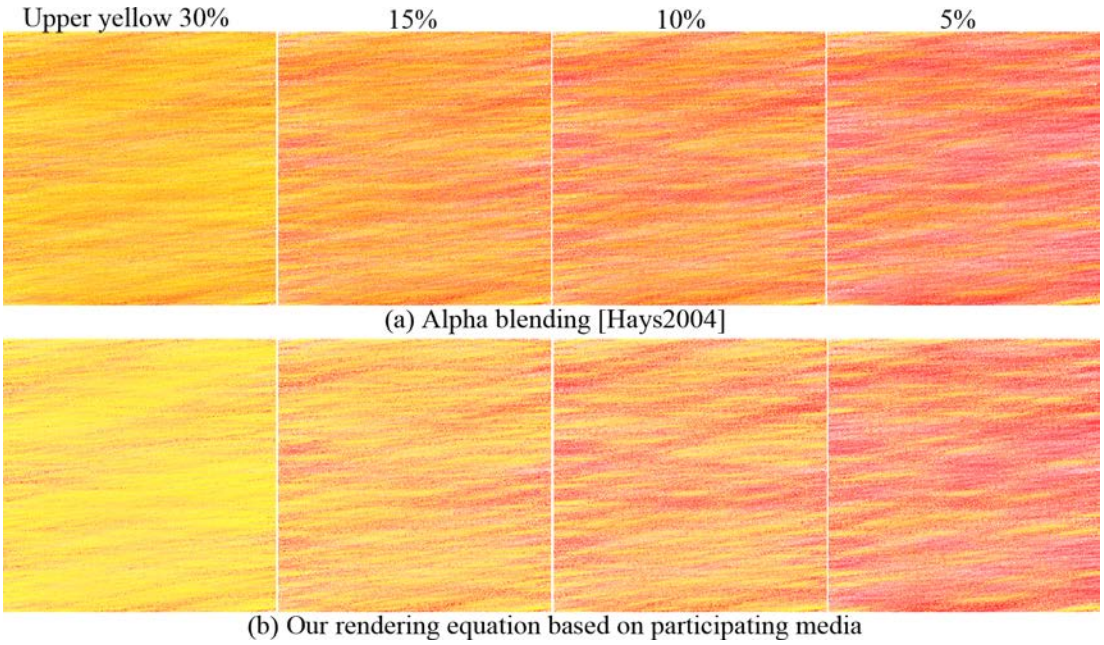

Fig. 16. Overlaying methods: Alpha blending vs. ours.

6. Implementation and Results

We implemented our algorithm in a personal computer with Pentium Core i7 processor and 32 GB of main memory. We compare our result with pencil drawing presented in [14] in **Fig. 17**. We used stochastic color coordination to generate noise color and a single layer approach. We also applied thick filter in skin to present smooth color. We also generated thicker bold noise to emphasize the wrinkles on the cloth. As a result, we have several distinguished points from the pencil drawing. The hair includes some red strokes and the wall includes some green strokes, which present vivid color effects. The skin on the face and the arm is expressed with

very smoothly varying color. Finally, the wrinkles on the cloth are expressed with much salient strokes. Therefore, we conclude that our approach presents pastel effects, which cannot be expressed by pencil drawing works such as [14].

In **Fig. 18**, we present another comparison with the LIC-based pencil drawing method by Lu et al. [20]. Their result shows a pencil drawing style of fine stroke patterns with sharp and thin edges. On the contrary, our result shows a style of painterly and bold stroke patterns. In **Fig. 19**, another distinguished point of ours to Lu et al.'s [20] is that our system can express charcoal drawings whose stroke patterns are also thick and painterly. Sousa et al.'s work [4] has a limitation in expressing fine features such as hair. Our system can produce both fine strokes that express detailed features such as hair and bold strokes that mimics charcoal strokes.

(a) Input image (b) [14] (c) Ours

Fig. 17. Comparison with colored pencil drawings for portrait.

(a) Input image (b) [20] (c) Ours

Fig. 18. Comparison with colored pencil drawings for a building.

(a) Input image (b) [4] (c) [20] (d) Ours

Fig. 19. Comparison with monochrome pencil drawings for portrait.

We present our results on landscapes in **Fig. 20**, other results in **Fig. 21** and their original images in **Fig.** 22. The computation time and parameters are in **Table 1**. Additional images and their parameters are suggested in the accompanying material. Furthermore, we request an artist to draw an original image in **Fig. 9** with color pastel and charcoal pencil (See **Fig. 23**). In comparison, we find that our results mimic the thick and bold stroke pattern, but they have some limitations in expressing artist's intuition like deformation of shape.

Our approach has several limitations. It depends on an image segmentation, which heavily depends on users' input. Another limitation is the difficulty in determining parameters such as the number of layers and the deposition of colors on each layer.

6.1 User Study

In NPR, the most frequently used evaluation scheme is the comparison with the results of the related works. Furthermore, we employ a focus group interview, by hiring 10 pastel artists including professional artists, graduate students and undergraduate students mastering fine arts. Our survey includes the following questions in 5-point metric.

(i) Express your professionality of pastel artwork
(ii) Express how our results resemble real pastel artworks
(iii) Write good points of our results
(iv) Write bad points of our results

We present 35 results composed of seven still lives including animals, thirteen portraits, eight landscapes, and seven pastel-mimicking drawings.

The interviewees record 4.5 in average for the first question and 4.3 in average for the second. In detail, each category acquires 4.4, 4.4, 4.1 and 4.2. The most frequent good point they wrote is that our system mimics pastel strokes of wide range from thin and fine to thick and bold. Other good points are the expression of smooth features and randomized color expression.

Therefore, we conclude that our system produces results similar to pastel artworks, but it still requires improvements. The interviewees pointed out that our system fails to capture deep artistic styles such as random stroke patterns. They also pointed out that our system cannot present some deformation of the objects.

Fig. 20. Our landscape results: Tiny textures on trees and flowers are expressed convincingly using stroke overlaying.

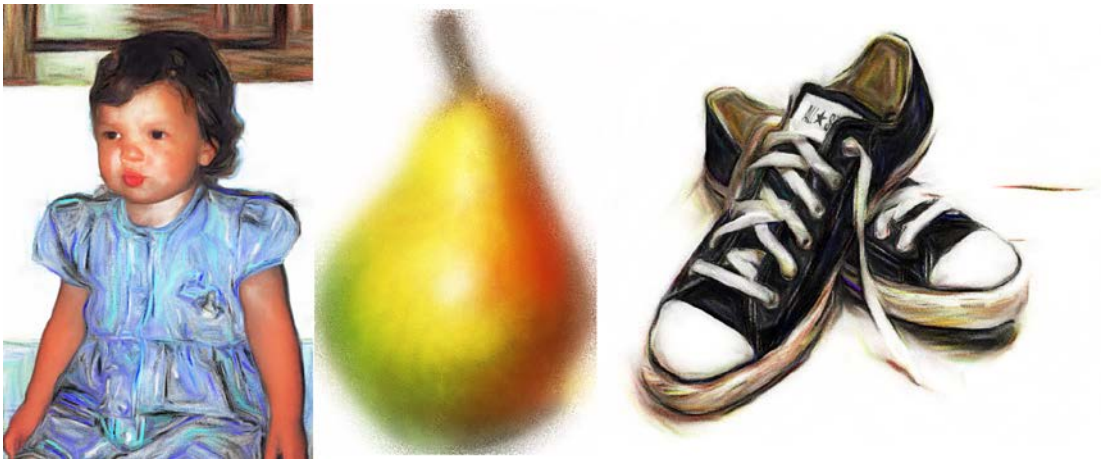

Fig. 21. Our results including a portrait and still-lifes.

Fig. 22. Original images.

Table 1. Parameters used in our results and computation time.

Figures		*l*	*w*	*S*	*T*	*τ*	*Time(sec)*
Fig. 20. Cherry (1024 x 678)	Building	35	3	31	1	15	5.04
	Botanical (*overlaying*)	9	7	20	1	40	
	Water (*smearing*)	15	2	43	5	60	
Fig. 20. Tree (800 x 644)	Leaves	15	5	20	1	40	16.23
	Sky	15	2	43	1	60	
	Trunk	35	3	43	1	15	
	Grass	7	3	20	1	40	
Fig. 21. Casy (512 x 752)	Skin (*smearing*)	2	1	43	8	35	9.58
	Hair (*smearing*)	10	2	43	8	35	
	Cloth (*overlaying*)	35	3	43	1	15	
	BG	25	4	43	1	15	
Fig. 21. Pear (468 x 736)	All (*smearing*)	10	1	50	10	60	1.52
Fig. 21. Shoes (832 x 700)	Label	40	5	43	1	15	2.09
	Others	5	3	43	1	15	
Fig. 23. Beach (800 x 600)	FG	100	5	43	1	60	2.09
	BG	100	5	43	1	15	

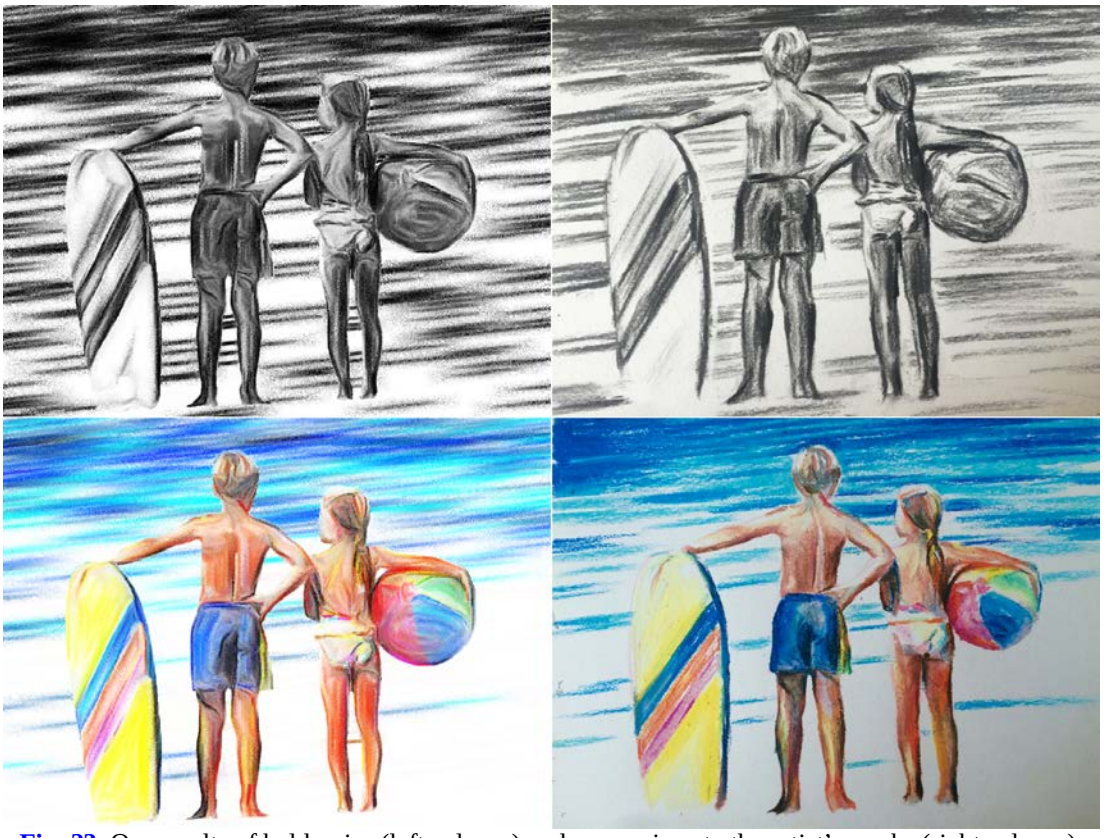

Fig. 23. Our results of bold noise (left column) and comparison to the artist's works (right column).

7. Conclusion and Future Work

In this paper, we presented a framework for producing pastel painting based on analysis of pastel media. To demonstrate the soundness of our framework, we conducted several experiments and comparisons. We also produced several visually pleasing pastel artworks in portrait, landscape, and still life using the framework.

The application of machine learning algorithms to nonphotorealistic rendering is the subject of considerable current research interest. Our efficient approach to mimicking pastel painting can complement this research. We intend to improve the framework proposed in this paper by mimicking various pastel artwork techniques and tailoring the expertise embedded in our framework.

Acknowledgement

This research was supported by a research grant from Sangmyung Univ. in 2014.

References

[1] H. Jensen and P. Christensen, “Efficient simulation of light transport in scenes with participating media using photon maps,” in *Proc. of Siggraph* 98, pp. 311–320, 1998. Article (CrossRef Link)

[2] H. Kang, S. Lee and C. Chui, “Flow-based image abstraction,” *IEEE TVCG* 15, 1, pp. 62–76, 2009. Article (CrossRef Link)

[3] S. Takagi, I. Fujishiro and M. Nakajimam, “Volumetric modeling of colored pencil drawing,” in *Proc. of Pacific Graphics* 99, pp. 250–258, 1999. Article (CrossRef Link)

[4] M. C. Sousa and J. Buchanan, “Computer-generated graphite pencil rendering of 3d polygonal models,” in *Proc. of Eurographics* 99, pp. 195–207, 1999. Article (CrossRef Link)

[5] M. C. Sousa and J. Buchanan, “Observational model of blenders and erasers in computer-generated pencil rendering,” in *Proc. of Grapics Interface* 99, pp. 157–166, 1999. Article (CrossRef Link)

[6] K. Murakami, R. Tsuruno and E. Genda, “Multiple illuminated paper textures for drawing strokes,” in *Proc. of Computer Graphics International* 05, pp. 156–161, 2005. Article (CrossRef Link)

[7] D. Rudolf, D. Mould and E. Neufeld, “A bidirectional deposition model of wax crayons,” *Comp. Graph. Forum* vol. 24, no. 1, pp. 27–39, 2005. Article (CrossRef Link)

[8] C. S. Haase and G. W. Meyer, "Modeling pigmented materials for realistic image synthesis," *ACM TOG*, vol. 11, no.4, pp. 305-335, 1992. Article (CrossRef Link)

[9] B. Cabral and C. Leedom, “Imaging vector field using line integral convolution,” in *Proc. of Siggraph* 93, pp. 263–270, 1993. Article (CrossRef Link)

[10] X. Mao, Y. Nagasaka and A. Imamiya, “Automatic generation of pencil drawing using lic,” in *Proc. of ACM Siggraph 2002 Conference Abstractions and Applications*, 149, 2002. Article (CrossRef Link)

[11] S. Yamamoto, X. Mao and A. Imamiya, “Colored pencil filter with custom colors,” in *Proc. of Pacific Graphics* 04, pp. 329–338, 2004. Article (CrossRef Link)

[12] S. Yamamoto, X. Mao and A. Imamiya, “Enhanced lic pencil filter,” in *Proc. of the International Conference on Computer Graphics, Imaging and Visualization* 04, pp. 251–256, 2004. Article (CrossRef Link)

[13] H. Yang and K. Min, “Feature-guided convolution for pencil rendering,” *KSII Trans. on Internet and Information Systems*, vol. 5, no. 7, pp. 1311–1328, 2011. Article (CrossRef Link)

[14] H. Yang, Y. Kwon and K. Min, “A Stylized Approach for Pencil Drawing from Photographs,” *Computer Graphics Forum* 31(4), 1471-1480, 2012. Article (CrossRef Link)

[15] Y. Kwon, H. Yang and K. Min, "Pencil rendering on 3d meshes using convolution," *Comp. & Graph*. vol. 36, no. 8, pp. 930–944, 2012. Article (CrossRef Link)
[16] M. Hata, M. Toyoura and X. Mao, "Automatic generation of accentuated pencil drawing with saliency map and lic," *Vis. Comp*. vol. 28, no. 6-8, pp. 657–668, 2012. Article (CrossRef Link)
[17] H. Matsui, H. Johan and T. Nishita, "Creating colored pencil images by drawing strokes based on boundaries of regions," in *Proc. of Computer Graphics International* 05, pp. 148–155, 2005. Article (CrossRef Link)
[18] H. Lee, S. Kwon and S. Lee, "Real-time pencil rendering," in *Proc. of NPAR* 06, vol. 4, no. 8, pp. 37–45, 2006. Article (CrossRef Link)
[19] Y. Kim, J. Yu, H. Yu and S. Lee, "Line-art illustration of dynamic and specular surfaces," *ACM ToG*, vol. 27, no. 5, 156, 2008. Article (CrossRef Link)
[20] C. Lu, L. Xu and J. Jia, "Combining sketch and tone for pencil drawing production," in *Proc. of NPAR* 2012, pp. 65–73, 2012. Article (CrossRef Link)
[21] A. Hertzmann, C. Jacobs, N. Oliver and B. Curless, "Image analogies," in *Proc. of Siggraph* 01, pp. 327–340, 2001. Article (CrossRef Link)
[22] L. Gatys, A. Ecker and M. Bethge, "Image style transfer using convolutional neural networks," in *Proc. of CVPR* 16, pp. 2414–2423, 2016. Article (CrossRef Link)
[23] A. Selim, M. Elgharib and L. Doyle, "Painting style transfer for head portraits using convolutional neural networks," *ACM ToG* vol. 35, no. 4, 129, 2016. Article (CrossRef Link)
[24] L. Lin, K. Zeng, H. Lv, Y. Wang, Y. Xu and S. Zhu, "Painterly animation using video semantics and feature correspondence," in *Proc. of NPAR* 10, pp. 73–80, 2010. Article (CrossRef Link)
[25] J. Hays and I. Essa, "Image and video-based painterly animation," in *Proc. of NPAR* 04, pp. 113–120, 2004. Article (CrossRef Link)
[26] Z. Chen, B. Kim, D. Ito, H. Wang, "Wetbrush: Gpu-based 3d painting simulation at the bristle level," *ACM ToG*, vol. 34, no. 6, 200, 2015. Article (CrossRef Link)

Heekyung Yang received her B.S. degree and M.S. degree from Sangmyung University, Seoul, Korea, in 2010. She is currently a Ph.D. student in the same college. Her research interests are computer graphics, especially NPR (non-photorealistic rendering). She is also interested in image processing, 3D-mesh processing, volume rendering and medical rendering.

Kyungha Min received his MS in Computer Science from KAIST in 1992. He received his BS and Ph.D in Computer Science and Engineering from POSTECH in 1994 and 2000, respectively. His main research interests are computer graphics and image processing.