Wet-in-Wet Flow Effect Synthesis for Ink Diffusion

Ren-Jie Wang Chung-Ming Wang
Department of Computer Science and Engineering, Institute of Networks and Multimedia,
National Chung Hsing University.
E-mail: {rjwang, cmwang }@cs.nchu.edu.tw

Figure 1. Taiwan's Yushan (Mount Morrison) [1] (left) and its rendered images created by our new algorithm to simulate the wet-in-wet flow effect with controllable flow strength (middle and right). The image in the middle emphasizes the flow effect in the dark color (the mountain rocks) and the right image accentuates the effect in the light color (the clouds and sky).

ABSTRACT
Wet-in-wet flow effect is a famous phenomenon in Chinese ink paintings. In this paper, we propose a new two-stage algorithm to synthesize this renowned effect. Given a reference image, in the first stage, we render the reference image to produce an ink diffused image. This is achieved by using a new, physically-based color ink diffusion synthesis algorithm. In addition, this new algorithm explores a new and more sensitive Kubelka-Munk (SK-M) equation. As a result, the new algorithm produces visually pleasing results better than our previous work [1]. In the second stage, we present a controllable flow technique to render the wet-in-wet effect. In particular, we adopt the adaptive length line integral convolution to represent the global flow of the reference image. We then construct a controllable flow map by referring to the luminance and global flow of the reference image, and the desired weight coefficients, which are controlled by the user. Finally, we blend the controllable flow map with the available ink diffused image using the new SK-M model, producing the final ink diffused image with the wet-in-wet effect (see Figure 1). Our algorithm has four advantages: visually plausible, controllable, independent, and simple.

Keywords
Ink Diffusion, Wet-in-Wet Effect, Adaptive Line Integral Convolution, Kubelka-Munk (K-M) Equation

1. INTRODUCTION
Chinese ink painting simulation can be classified into two categories: non-image-based approach, which focuses on modeling and simulating an artist's hair brushes, and image-based approach, which renders a reference image with variant ink painting styles [1] (see a detailed survey in next section).

Ink diffusion is the most noticeable phenomenon in Chinese ink painting, which is caused by the microscopic capillary effect of absorbent paper [2][3][4][5]. Our previous work [1] proposed an image-based painterly rendering algorithm for automatically synthesizing an image with color ink diffusion. They suggested a mathematical model including a K-M equation with a physical base to achieve this goal. Their work indicates a success of simulating ink diffusion from an image-based approach. However, their results do not show a tone visually similar to the reference image. This is due to the fact that the K-M equation they employed is not sensitive enough to calculate the subtle change of the reflectance value for the mixed result.

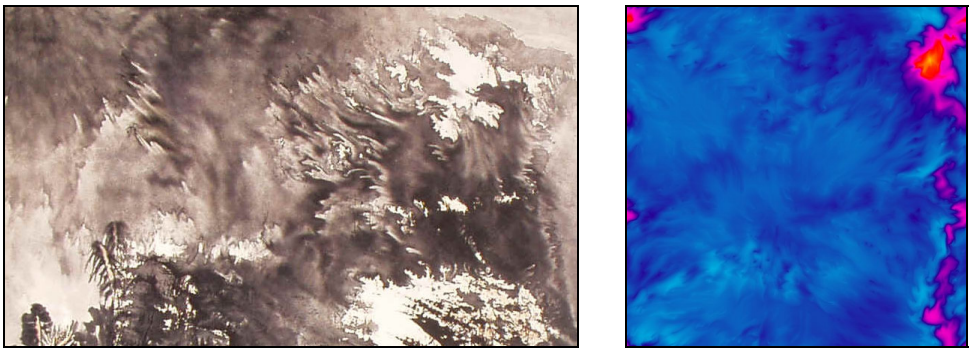

Figure 2. Famous ink effects--diffusion and wet-in-wet flow. Left picture is real Chinese ink painting (by Feng-Qi Tang). Right picture is a simulated result created by Chu et al's work [6].

Apart from ink diffusion, the wet-in-wet flow effect is another famous appearance in many Chinese ink painting works, as shown in Figure 2. This effect is due to the fact that the surface of the wet paper allows the colloidal ink to spread and follow the direction of water flow. Chu et al. [6] proposed a physically-based method for simulating ink dispersion in absorbent paper. Through a painting system, they demonstrate simulation results of various realistic effects with flow patterns. Their work indicates a success of wet-in-wet flow simulation from a non-image-based approach. The wet-in-wet flow effect had also been simulated in the work of computer-generated watercolor [7], and is called "fluid flow effect." Unfortunately, most image-based algorithms do not keep pace with the advances of wet-in-wet flow simulation. This has significantly limited the simulation applications of this important and renowned appearance. Therefore, the need to provide wet-in-wet flow simulation from the aspect of image-based approach is clear.

In this paper, we propose two approaches to conquer this problem. In particular, we present an IBCIDS algorithm, which employs a more sensitive K-M equation and a new overlapping equation to produce visually pleasing ink-diffused images. These two new equations enable our algorithm to produce a tone similar to the reference image. The resultant ink-diffused image is more visually pleasing than our previous work's result (see Figure 4). In regard to simulating the wet-in-wet flow effect, we introduce the concept of the flow map. In particular, we present a controllable wet-in-wet flow technique (CWFT), which is designed to be independent of the IBCIDS algorithm to ease the algorithm's complexity. In this technique, we apply the adaptive length line integral convolution (ALLIC) algorithm to depict the reference image's global flow. Then, we refer to the luminance of the reference image together with the desired weighted coefficient provided by the user to construct a controllable flow map. Finally, we utilize the more sensitive K-M equation we proposed to blend the flow map with the resultant diffused image simulated by our IBCIDS algorithm. The final rendered result contains both an effective color ink diffusion result and the wet-in-wet flow effect. We emphasize that we do not mimic Chinese ink painting styles. Rather, we present a new way to generate another illustrious color ink diffusion effect, the wet-in-wet flow phenomenon. We leave Chinese ink painting styles as a separate and interesting topic to explore in the future.

The overview of this paper is as follows. Section 2 takes a closer look at related research. Section 3 introduces the proposed algorithm. Section 4 shows the simulation results. Conclusions and future work are presented in section 5.

2. RELATED WORK

Research in the Chinese ink painting simulation can be classified into two categories: non-image-based and image-based [1]. The first category does not need a reference image, and focuses on modeling and simulating an artist's hair brushes, such as digital painting systems or illustration generators. Algorithms in this way generally try to produce realistic brush strokes on canvas [2][3][4][8][9][10][11][12][13]. Some of these approaches emphasize their real-time benefit, and they employ texture mapping or some specific but non-physical techniques. Lee [3] presented a practical technique to render oriental black ink paintings with realistic diffusion effects. His approach simulated a variety of paper types and black ink properties by specific algorithms. Others employ mathematical or physical models to obtain remarkable results. Kunii et al. [2][14] and Lee et al. [5] proposed a multidimensional diffusion model to simulate ink diffusion in absorbent paper. By physical analysis of real ink diffusion images, their model faithfully simulates the ink diffusion phenomenon. According to their approach, diffusion of diffuse

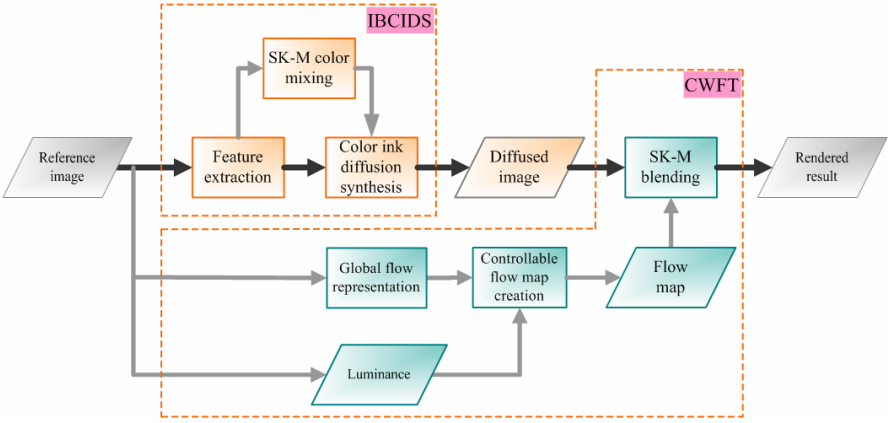

Figure 3. System architecture.

ink (colloidal liquid) on the surface of paper should be considered as two separate diffusion processes. In other words, diffusion of water in the paper and diffusion of the motion of solid particles in water should be treated as different (but interlinked) processes. Guo and Kunii [4] proposed an interactive painting system for generating high quality and artistic calligraphy characters and black ink paintings. Their system was a physically-based algorithm which simulated the dynamic diffusion of liquid ink into absorbent painting paper. However, these physically-based approaches merely focus on simulating the diffusion of black ink. Chu et al. [6] presented a lattice Boltzmann equation (LBE) method to simulate ink dispersion in absorbent paper for art creation purposes. They demonstrated a digital painting system with various realistic ink dispersion effects, including complex flow patterns observed in real artwork, and other special effects.

The second category is image-based. Picture retouching systems belong to this category. Algorithms in this way generally apply user-defined patterns or texture maps to a reference image in order to render some Chinese ink painting styles [15][16][17]. Yu et al. [16] described a two-stage framework for synthesizing Chinese landscape painting. They mainly employed some brush stroke texture primitivities (BSTP) to mimic the hand-made effects. Farbiz et al. [17] described a four steps algorithm to automatically generate an Asian ink painting result from photographs. Their method allows the output image objects not to be in the same position as they are in the input image, and they try to convey the artists' inner feelings just through using a few simple strokes on paper. In this category, few researchers have proposed a color ink diffused method based on physical foundations. Some commercial packages produce ink diffusion effects with blur filters, but those packages are not physically-based approaches. Our previous work [1] proposed an image-based painterly rendering algorithm for automatically synthesizing an image with color ink diffusion. We suggested a mathematical model with a physical base to simulate the phenomenon of color colloidal ink diffusing into absorbent paper, and proposed a non-stroke-based algorithm that allows an input color image to be automatically converted to the color diffused ink style with a visually pleasing appearance. However, our previous work has two defects. First, ink diffusion is one of wet-in-wet effects. Artists in Chinese ink paintings always demonstrate more wet-in-wet effects in their works, such as wet-in-wet flow. Secondly, our results do not have quite similar tone to the reference images.

3. Wet-in-Wet Flow Effect Synthesis for Ink Diffusion

In this paper, we propose a novel, image-based algorithm to support not only the color ink diffusion effect, but also the wet-in-wet flow effect. Our system consists of two independent parts: the image-based color ink diffusion synthesis (IBCIDS) and the controllable wet-in-wet flow technique (CWFT) (illustrated in Figure 3). The IBCIDS algorithm simulates color ink diffusion first. Based on this result, the CWFT generates wet-in-wet flow effect, producing the final rendered result. We will describe each part in the following sections.

3.1 Image-based color ink diffusion synthesis

The color ink used in Chinese ink painting is a colloidal liquid which mainly consists of water and pigment particles. We employ a physically-based color ink diffusion model which was originally proposed by Kunii et al. [2] and extended by our previous work [1] to simulate color ink diffusing on absorbent paper. But,

we emphasize that, our IBCIDS is better than our previous work [1] because we employ a more sensitive K-M equation to mix pigment color, and use a better equation to overlap deposit layer with diffusion layer. Thus, our rendered result has a similar tone to the reference image.

Our IBCIDS consists of two layers to simulate the process of diffusion: diffusion layer and deposit layer. The diffusion layer is used to abstract non-features of the reference image, and the deposit layer reinforces the key features. When a drop of colloidal ink falls on the surface of absorbent paper, the water will spread through the paper structure until it is absorbed. At the same time, the pigment particles are carried by the flow of water. If the accumulated quantity of pigment is greater than some threshold (here we assume that threshold Q_{th} is equal to 1), the pigment particles will be deposited on the deposit layer; otherwise they will be diffused on the diffusion layer. The rendered color is generated by a new overlapping equation, which overlaps the diffusion and deposit layers as follows:

$$R_{i,j} = \left(\frac{P_{i,j}+G_{i,j}}{d_{i,j}+P_{i,j}+G_{i,j}} C_{i,j} + \frac{d_{i,j}}{d_{i,j}+P_{i,j}+G_{i,j}} \max(C_{i,j}, D_{i,j})\right) \quad (1)$$

Here $R_{i,j}$ denotes the rendered color of pixel (i, j); $D_{i,j}$ denotes the color of the deposit layer; $C_{i,j}$ denotes the color of the diffusion layer. $d_{i,j}$ denotes the quantity of the deposit layer; $P_{i,j}$ is the quantity of the pigment; $G_{i,j}$ denotes the quantity of the water. Our new overlapping equation is more effective than our previous work since we take more factors into account, such as water flow and deposit quantity.

When artists paint, they often delineate some features and make others more abstract. Our previous work [1] suggested a two-phase feature extraction: the luminance division and color segmentation to simplify the input information; and the block variation to extract key characteristics. However, they use a fixed threshold to decide the key characteristics. It is not always suitable for all input reference images. Here, we assign a statistic value $\overline{VB}$ -- the mean of $VB_{i,j}$, to the threshold to make the key characteristics more reasonable because every image has its own $\overline{VB}$ .Our block variation equation is described as follows:

$$V_{i,j} = \frac{1}{N^2} \sum_{i-\frac{n}{2}}^{i+\frac{n}{2}} \sum_{j-\frac{n}{2}}^{j+\frac{n}{2}} (\overline{L'} - L'_{i,j})^2,$$

$$VB_{i,j} = \frac{V_{i,j} - VB_m}{VB_M - VB_m},$$

$$\overline{VB} = \frac{\sum VB_{i,j}}{M \times N}, \quad (2)$$

where $L'_{i,j}$ is luminance of reference image on a pixel (i, j) ; and $\overline{L'}$ denotes mean. $VB_{i,j}$ denotes the block variation of luminance on a pixel (i, j) in the reference image. n is block size. Here we use a block size with 7 pixel width, but users can decide other suitable block sizes. VB_M denotes the maximum of block variation, and VB_m denotes the minimum. $VB_{i,j}$ has a value between 0 and 1. $\overline{VB}$ is the mean of $VB_{i,j}$. The scale of reference image is $M \times N$. By assigning $\overline{VB}$ to the threshold, block variation can generate adequate key features rather than gradient magnitude.

The following algorithm can help to decide the quantity of pigment and the quantity of water by the block variation $VB_{i,j}$.

PaintOnPaper (i, j)

```
1  if (VB_ij ≥ VB ){          //key characteristic, thick ink
2     assign very little water
3     assign maximum pigment quantity}
4  else {                     //user-defined ink
5     assign user-defined water
6     assign user-defined pigment quantity}
7  if (pigment quantity pq_ij ≥ Q_th){
8    d_ij←pq_ij  }            //deposit to deposit layer
```

Line 1, we use the mean of block variation $\overline{VB}$, a statistic value, to decide where the key characteristics are and assign adequate quantity to water and pigment. We render the scene from light pigment to dark pigment in animation. If the user-defined pigment quantity is greater than $0.5Q_{th}$ and less than Q_{th}, not only the key characteristics but also the remarkable edges will deposit on deposit layer. Those pigments located on diffusion layer will diffuse to pale. The others located on deposit layer still keep their color.

Figure 4 presents diffused images generated by our IBCIDS (middle column) and the rendered results simulated by our previous work [1] (right column). We can see that our results have similar tones to the reference images rather than our previous work.

3.2 Controllable wet-in-wet flow technique

The CWFT is independent of the IBCIDS to take advantage of facilitating the complexity of the whole simulation system. In our algorithm, we employ an adaptive length line integral convolution (ALLIC) to represent the global flow of the reference image. Then, we refer the luminance of the reference image and the weight coefficient given by the user to construct a controllable flow map which is used to decide the resultant flow strength. The ALLIC, a technique converted from line integral convolution (LIC) vector field, is employed to portray the reference image's global flow. The LIC method, originally developed for imaging vector field in scientific visualization, has the potential to produce images with directional characteristics

Figure 4. A diffused image generated by our IBCIDS algorithm (middle column) and the rendered result simulated by our previous work [1] (right column). The pictures on the left column are reference images

[18][19]. However, the LIC vector field which has the same length and sharp visualization is not suitable, if directly used to enhance wet-in-wet effect. Thus, we adapt the LIC equation to an adaptive length line integral convolution (ALLIC) as follows:

$$I(x_0) = \int_{s_0-l(x_0)}^{s_0+l(x_0)} k(l(x_0)-s_0)\,T(\sigma(l(x_0)), \overline{L'})ds$$

$$l(x_0) = l_M - (l_M - l_m) \times L'_{x_0}, \qquad (3)$$

where k denotes a normalized, one dimensional filter kernel with the length magnitude of $2l(x_0)$. $T()$ is our noise function. We modify the original noise function with an extra frequency parameter. In our experiment, if frequency is the mean of input image's luminance $\overline{L'}$, we can obtain an optimal wet-in-wet flow effect in the final step.

Given a flow line $\sigma(l(x_0))$, parameterized by the arc-length $l(x_0)$, the equation shown as above calculates the intensity of an output pixel $I(x_0)$ located at $x_0 = \sigma(l(x_0))$, by the convolution integral [18]. Here, we give a length function $l(x_0)$ to control the length. l_M and l_m are the user-defined maximum length and minimum length respectively. L'_{x_0} is the luminance of pixel x_0. Figure 5 shows the difference between ALLIC vector field and LIC vector field. ALLIC vector field has variant lengths on different luminance. Our maximum length l_M is 10 (i.e. maximum length is 20) and minimum length lm is 2 (i.e. minimum length is 4).

The ALLIC vector field is very sharp. Thus, we use anisotropic diffusion to smooth it but still keep features clear. Anisotropic diffusion [20] is a well-known

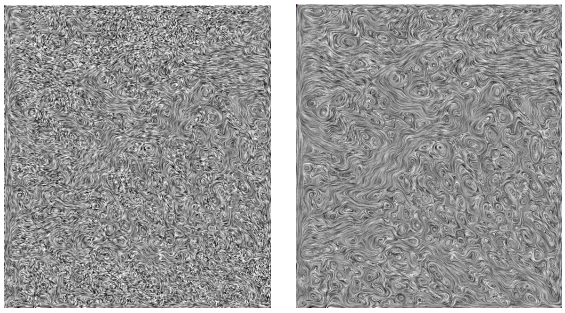

Figure 5. ALLIC vector field with an adaptive length L_M=10 and L_m=2 (left) and LIC vector field with a fixed length L=10 (right).

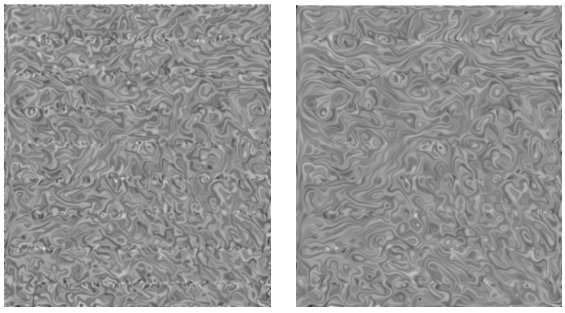

Figure 6. A smooth ALLIC vector field with L_M=10 and L_m=2 (left) and a smooth LIC vector field with L=10 (right).

smooth filter in digital image processing community. We employ the anisotropic diffusion filter provided by a commercial package, the Adobe Photoshop, to produce a smooth ALLIC vector field. Figure 6 shows the difference between smooth ALLIC vector field and smooth LIC vector field. Smooth ALLIC vector field has variant lengths distributed on variant luminance but the smooth LIC does not.

Then, the flow map $\varphi_{i,j}$ of pixel (i, j) can be generated by the following equation.

$$\varphi_{i,j} = 1 - \alpha \bullet il_{i,j} \bullet \widetilde{I}_{i,j} \qquad (4)$$

where $\widetilde{I}_{i,j}$ denotes the smooth ALLIC; $il_{i,j}$ is the luminance on the reference image (i, j); and α is weight coefficient (here default α=0.5~1.0). Figure 7 illustrates different flow maps generated by the smooth ALLIC vector field with variant α value. The flow map is controllable. We can use different flow map to emphasize dark color or light color. We also can choose variant α value to control flow strength.

3.3 Rendering algorithm

Our system consists of two independent parts: IBCIDS and CWFT. The IBCIDS generates a color ink-diffused

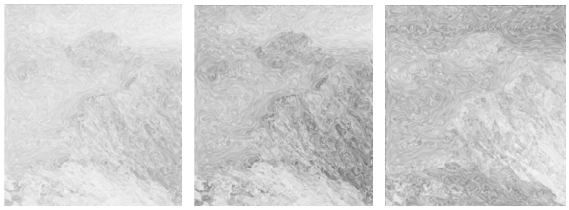

Figure 7. Different flow maps created by smooth ALLIC vector field with variant α value: with α=0.6 (left), with α=1.0 (middle), and with α=0.8 reverse (right).

image from light pigment to dark pigment in animation. The CWFT blends the ink diffusion with wet-in-wet flow effect in interaction. The rendering algorithm is described as follows:

RenderAlgorithm()
//$il_{i,j}$: luminance on a reference image (i, j)
//$iC_{i,j}$: color on a reference image (i, j)
//$ol_{i,j}$: luminance on rendering canvas (i, j)
//$oC_{i,j}$: color on rendering canvas (i, j)
/*IBCIDS*/
1 while (water not dry) {
2 for (all light layer to dark layer k) {
3 for (all $il_{i,j}$ in layer k) {
4 if ($il_{i,j} - ol_{i,j} \geq O_T$) { //overlap threshold O_T
5 PaintOnPaper(i, j) //water, pigment
6 $oC_{i,j} \leftarrow$SK-M($iC_{i,j}$, $pq_{i,j}$, $oC_{i,j}$, $P_{i,j}$)}}} // color
7 ColorInkDiffusionSynthesis(G, P, oC)}
/*CWFT*/
8 for (all i, j) {
9 $oC_{i,j} \leftarrow$SK-M(1-$oC_{i,j}$, $\varphi_{i,j}$, 0, 0) } //SK-M blending

In the IBCIDS phase, we render the diffused image on rendering canvas from light pigment to dark pigment in animation, and every pigment layer has an interval of time (here we assume 6 iterations of diffused computation) to make diffusion processing. And, the ink diffusion will continue computing until water is dry. In the above algorithm, line 4 decides whether the pixel of the reference image is painted on the rendering canvas or not. Line 6 is the SK-M color mixing. When a pixel with color $iC_{i,j}$, and pigment quantity $pq_{i,j}$, is painted to the rendering canvas with color $oC_{i,j}$ and pigment quantity $P_{i,j}$, its resultant color should take account of the color on the rendering canvas. Our sensitive K-M equation is employed to approximate the result of mixing the color. In the CWFT phase, line 9 is the SK-M blending. The SK-M equation plays a very important role in this phase. Thus, we introduce our new and sensitive K-M equation in next section.

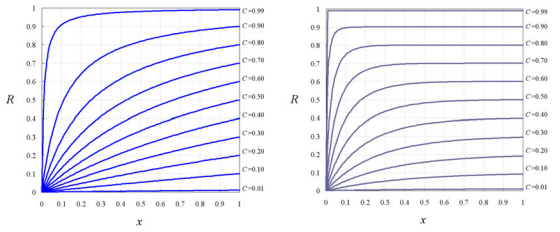

Figure 8. The relation of the input color C, thickness x, and the output reflectance R using the SK-M equation we propose (left) and the original equation in our previous work [1] (right).

3.4 Sensitive K-M equation

When using the K-M theory for a typical application to mix Color-1 with Color-2, we need to determine the corresponding scattering coefficient S and the corresponding absorption coefficient A for each color. These coefficients are usually measured with spectral measurements from a layer with the known thickness x. Our previous work [1] has suggested a simplified method to approximate S and A. Here, we propose a more sensitive K-M (SK-M) equation inspired from Curtis' work [7].

We first assume the paper's colors are nearly white. Thus, we set the color *white* to a safe value (here assumed to be 0.9999998) and compute the coefficients (S, a, and b) for the RGB color value C by the following equations:

$$S = \frac{1}{b} \bullet \coth^{-1}\left(\frac{b^2 - (a - white)(a - 1)}{b(1 - white)}\right)$$

$$a = \frac{1}{2}\left(white + \frac{C - white + 1}{white}\right), \quad b = \sqrt{a^2 + 1}\,. \qquad (5)$$

Note that for Color-1 and Color-2, we need to determine its corresponding coefficients (S, a, and b). In our system, we compute the coefficients (S, a, and b) for color $iC_{i,j}$ and $oC_{i,j}$ using the above equations. Then, we compute the reflectance R_1 and transmittance T_1 for color $iC_{i,j}$ by setting the thickness coefficient x to $pq_{i,j}$, as shown in Equation 6. This was Kubelka's optical compositing equation [7]. Similarly, we compute the R_2 and T_2 of the color $oC_{i,j}$ by setting the thickness coefficient x to $P_{i,j}$. Then, we can determine the overall reflectance R and transmittance T.

$$R = \sinh bSx / c,$$

$$T = b / c \text{ , where } c = a \sinh bSx + b \cosh bSx,$$

$$R = R_1 + \frac{T_1^2 R_2}{1 - R_1 R_2}\ ,\ T = \frac{T_1 T_2}{1 - R_1 R_2}\,. \qquad (6)$$

Here, we directly use the composite value R to express the result of color mixing. Figure 8 compares

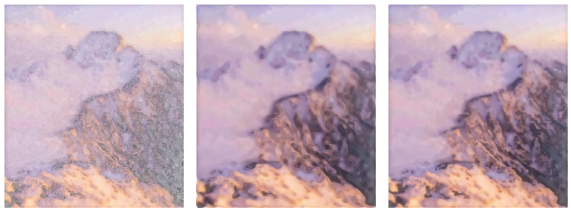

Figure 9. Some resultant images created by other approaches (diffused image is Figure 4, the first row, middle picture): using alpha blending on the diffused image (left); directly applying LIC algorithm on the diffused image (middle); employing Adobe Photoshop's anisotropic diffusion filter on the diffused image (right).

the SK-M equation we propose with the original one in our previous work [1]. Clearly, our equation demonstrates that the responding reflectance R (left vertical axis) for different input colors C (right vertical axis) is sensitive with respect to various thicknesses x (the horizontal axis). The comparison shows that that our new equation can generate more sensitive reflectance R than our previous work [1]. If the real paper color is not white, we can compose the finally rendered result with the paper's color by the above K-M equation. The reflectance R which is calculated by K-M equation is a non-linear curve. This is why K-M theory is more suitable than linear interpolation method for pigment-based rendering simulation. Because conventional pigments used in Chinese ink painting are different from those in watercolor, we leave the spectral measurements to pigments used in Chinese ink painting as a separate topic to explore in the future.

In the above rendering algorithm, line 9 uses a flow map φ to enhance wet-in-wet flow effect by K-M equation. Before being combined with flow map, the diffused result contains paper texture and diffusion effect, shown as Figure 4 in the first row, middle picture. After combining the diffused result with flow map (Figure 7, middle picture), we can obviously see that the color ink diffused within paper fibers and wet-in-wet flow effect is visually pleasing (shown as Figure 1, middle). If we blend the diffused image (Figure 4, the first row, left picture) with a reverse flow map (shown in Figure 7, right), the rendered result (shown in Figure 1, right) will accentuate its wet-in-wet flow effect on the light color. The K-M blending produces a non-linear and visual pleasing result. This is why K-M theory is also more suitable than linear interpolation method for blending the diffused image with flow map in our algorithm.

4. RESULTS

We implemented our system using C++ programming language and OpenGL on a personal computer. It contains a 3.0 GHz CPU, 1.0 GB RAM, and a video card with Nvidia GeForce 6600 GPU and 128MB video RAM.

Given a diffused image rendered by our algorithm (Figure 4, the first row, middle picture), Figure 9 shows three images rendered by other approaches for comparison, including the alpha blending, directly applying the LIC technique only, and employing commercial Adobe Photoshop's anisotropic diffusion filter. By comparing with Figure 1 (middle and right pictures), our new algorithm renders an image demonstrating visually pleasing appearance with diffused ink and wet-in-wet flow style. The results imply that to generate wet-in-wet flow effect, it is inappropriate to simply apply the alpha blending, the LIC algorithm or the anisotropic diffusion filter on a diffused image. The results also show the power of the flow map and the SK-M-blending process in our method.

Figure 10 presents results generated by different flow maps to demonstrate that our wet-in-wet flow effect is controllable. We can directly use flow map to emphasize dark color. In contrast, we can also reverse the flow map to emphasize the light color. As a result, a user can fully control the flow strength by changing the weight coefficient α.

Figure 11 compares the black ink result created by our previous work [1] (middle picture) with ours (right picture). Observing from these images, we verify that our algorithm renders more visible ink diffusion and variant wet-in-wet flow effects, producing visually plausible appearance.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we present a novel algorithm to synthesize both ink diffusion with better visual effect and wet-in-wet flow effect. We conclude that our algorithm has four advantages. First, the quality of the resultant images we rendered is more visually plausible than our previous work's result. This is contributed from using our new rendering algorithm, the more sensitive K-M equation, our controllable flow technique, and the new overlapping equation. Secondly, our algorithm is controllable, allowing a user to manipulate the flow strength via variant flow maps when simulating the wet-in-wet flow effect. Thirdly, the algorithm is independent. This feature makes it possible to ingeniously weave the controllable wet-in-wet flow technique (CWFT) not only to our ink diffusion algorithm but also to other ink diffusion approaches. Thus, the integrated color ink diffusion algorithm takes both advantages to have significant features, including feature-

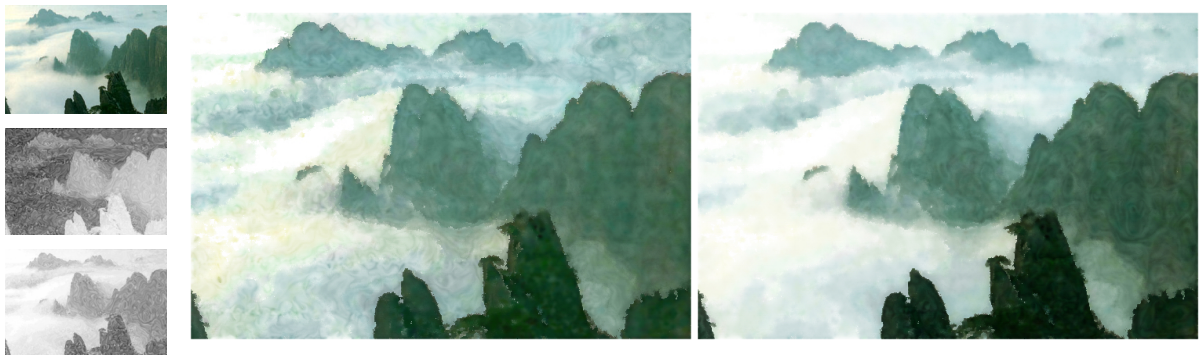

Figure 10. Two images demonstrating controllable flow strength in simulating the wet-in-wet flow effect. The left top picture is the reference image [21]. The middle image emphasizes the light color (clouds) using a reverse flow map (left middle) with α=1.0. The right image focuses on the dark color (mountains) using the flow map with α=1.0 (left bottom).

Figure 11. A black ink result simulated by our algorithm. The picture on the left top is the reference image [1] and the one in the left bottom is the reverse flow map (α=0.9). The middle image is our previous work's result and the right one is our result. Clearly, our result demonstrates more visible ink diffusion style and the wet-in-wet flow effect in the cloud of the sky.

based, non-SBR, physically-based, and controllable wet-in-wet flow effect. Finally, it is simple. Without any strokes, a color image can be automatically converted to a visually pleasing appearance with both diffused ink and wet-in-wet flow style.

Although a physically-based approach has the benefit of realistic simulation, it, in the mean time, has a problem with computation. Our first future task, therefore, is to speed up the computation. In addition, artists usually use a limited number of conventional pigments in Chinese ink painting. Yet, a real image may contain more than a million colors. Our next task is to synthesize an even more realistic result by employing the conventional pigments used in Chinese ink painting. This would make the color appearance shown in our result look more like Chinese ink painting in real world.

6. ACKNOWLEDGMENTS

This research was supported by the National Science Council (NSC) Taiwan under the grant numbers NSC 95-2221-E-005-046, NSC 94-2213-E-005-022, NSC 93-2213-E-005-018, and NSC 92-2213-E-005-021. The reference image in Figure 1 is photographed by Mr. Ming-Nan Chien. We thank Ms. Irma Chen for her help in improving the clarity of this paper.

7. REFERENCES

[1] C. M. Wang and R. J. Wang, "Image-based color ink diffusion rendering," *IEEE Transactions on Visualization and Computer Graphics*, Vol. 13, No. 2, pp. 235-246, 2007.

[2] T. L. Kunii, G. V. Nosovskij, and T. A. Hayashi, "Diffusion model for computer animation of dif-

fuse ink painting," *Proceedings of Computer Animation '95*, pp. 98-102, 1995.

[3] J. Lee, "Diffusion rendering of black ink paintings using new paper and ink models," *Computers & Graphics*, Vol. 25, pp. 295-308, 2001.

[4] Q. L. Guo and T. L. Kunii, "Nijimi rendering algorithm for creating quality black ink paintings," *Proceedings of the Computer Graphics international (CGI'03)*, pp. 152-159, 2003.

[5] S. Lee, H. Y. Lee, I. F. Lee, and C. Y. Tseng, "Ink diffusion in water," *European Journal of Physics*, Vol. 25, No. 2, pp. 331-336, 2004.

[6] N. S. H. Chu and C. L. Tai, "MoXi: real-time ink dispersion in absorbent paper," *ACM Transactions on Graphics (SIGGRAPH 2005 issue)*, Vol. 24, No. 3, pp. 504-511, 2005.

[7] C. J. Curtis, S. E. Anderson, J. E. Seims, K. W. Fleischery, and D. H. Salesin, "Computer-generated watercolor," *Proceedings of 24th Annual Conference on Computer Graphics & Interactive Techniques*, ACM SIGGRAPH, ACM Press, pp. 421-430, 1997.

[8] H. T. F. Wong and H. H. S. Ip, "Virtual brush: a model-based synthesis of Chinese calligraphy," *Computer & Graphics*, Vol. 24, pp. 99-113, 2000.

[9] C. W. Hsu, D. L. Way, and Z. C. Shih, "The synthesis of rock textures in Chinese landscape painting," *Computer Graphics Forum*, Vol. 20, No. 3, pp. C123-C131, 2001.

[10] C. Chan and E. Akleman, "Two methods for creating Chinese painting," *Proceedings of 10th Pacific Conference on Computer Graphics and Applications (PG'02)*, pp. 403-412, 2002.

[11] J. S. Yeh, , T. Lien, and Y. M. Ouhyoung, "On the effects of haptic display in brush and ink simulation for Chinese painting and calligraphy," *Proceedings of 10th Pacific Conference on Computer Graphics and Applications (PG'02)*, pp. 439-441, 2002.

[12] S. H. Chu and C. L. Tai, "Real-time painting with an expressive virtual Chinese brush," *IEEE Computer Graphics and Applications*, Vol. 24, No. 5, pp. 76-85, 2004.

[13] D. L. Way, W. J. Lin, and Z. C. Shih, "Computer-generated Chinese color ink paintings," *Journal of the Chinese Institute of Engineers*, Vol. 29, No. 6, pp. 1041-1050, 2006.

[14] T. L. Kunii and G. V. Nosovskij, "Two-dimensional diffusion model for diffusion ink painting," *International Journal of Shape Modeling*, Vol. 7, No. 1, pp. 45-58, 2001.

[15] S. Z. Wen, Z. C. Shih, and H. Y. Chiu, "The synthesis of Chinese ink painting," *Proceedings of National Computing Symposium '99 (NCS'99)*, Taiwan, pp. 461-468, 1999.

[16] J. H. Yu, G. M. Luo, and Q. S. Peng, "Image-based synthesis of Chinese landscape painting," *Journal of Computer Science and Technology*, Vol. 18, No. 1, pp. 22-28, 2003.

[17] F. Farbiz, A. D. Cheok, and P. Lincoln, "Automatic Asian art: computers converting photos to Asian paintings using humanistic fuzzy logic rules," *Proceedings of the SIGGRAPH 2003, Sketches & Applications*, p. 1, 2003.

[18] D. H. Stalling and C. Hege, "Fast and resolution independent line integral convolution," *Proceedings of the ACM SIGGRAPH '95 Conference on Computer Graphics*, pp. 249-256, 1995.

[19] C. M. Wang and J. S. Lee, "Using ILIC algorithm for an impressionist effect and stylized virtual environments," *International Journal of Visual Languages and Computing*, Vol. 14, No. 3, pp. 255-274, 2003.

[20] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 7, pp. 629-639, 1990.

[21] A. Hertzmann, "Painterly rendering with curved brush strokes of multiple sizes," *Proceedings of ACM SIGGRAPH 98*, pp. 453-460, 1998.