Multimedia Tools and Applications manuscript No.
(will be inserted by the editor)

Lixing Dong · Shufang Lu · Xiaogang Jin

Real-Time Image-Based Chinese Ink Painting Rendering

Submitted October 1, 2011;*Revised* March 22, 2012

Abstract Chinese ink painting, also known as ink and wash painting, is a technically demanding art form. Creating Chinese ink paintings usually requires great skill, concentration, and years of training. This paper presents a novel real-time, automatic framework to convert images into Chinese ink painting style. Given an input image, we first construct its saliency map which captures the visual contents in perceptually salient regions. Next, the image is abstracted and its salient edges are calculated with the help of the saliency map. Then, the abstracted image is diffused by a non-physical ink diffusion process. After that, we combine the diffused image and the salient edges to obtain a composition image. Finally, the composition image is decolorized and texture advected to synthesize the resulting image with Chinese ink painting style. The whole pipeline is implemented on the GPU, enabling a real-time performance. We also propose some optional steps (foreground segmentation and image inversion) to improve the rendering quality. Experimental results show that our model is two to three orders of magnitude faster, while producing results comparable the ones obtained with the current image-based Chinese ink painting rendering method.

Keywords Chinese ink painting · Non-photorealistic rendering · Saliency map · Ink diffusion

1 Introduction

Chinese ink painting, also known by its Chinese name *shui-mo hua*, is an East Asian type of brush painting. It is a unique and fascinating form of art. Similar to East Asian calligraphy, only black ink is used. Chinese ink paintings are created by use of ink brush, ink stick, ink stone and *Xuan* paper. The created art works are usually stylized and the objects they depict are generally limited to plants, villages and landscapes. Ink brushes play an important role in Chinese ink painting; they help render a variety of strokes. This leads to lots of Chinese ink painting simulation work based on brushes [1][2]. Relatively, image-based approaches receive little attention.

Creating Chinese ink paintings usually requires great skill, concentration, and years of training. Few existing methods can convert photos into Chinese ink paintings automatically. In 2007, Wang et al. [3] proposed a physically-based Image-Based Color Ink Diffusion Rendering (IBCIDR) algorithm to synthesize an image with ink diffusion. It is also used to synthesize Chinese ink paintings with a visually pleasing appearance. This approach makes it possible to create images with Chinese ink painting style easily without using any strokes. However, it is quite slow. To render an image with 590×520, it usually needs several minutes to compute. This motivates us to develop a fast method to convert images into Chinese ink painting style. Nowadays, with the fast development of GPU and parallelized computing, more and more algorithms based on GPU have been proposed. The purpose of the paper is to develop a

L. Dong · S. Lu · X. Jin
State Key Lab of CAD & CG, Zhejiang University, Hangzhou, 310027, P. R. China
E-mail: cee.xingweimian@gmail.com, shufanglu@gmail.com, jin@cad.zju.edu.cn

(a) Input image (b) Our result

Fig. 1 An example of our real-time Chinese ink painting rendering algorithm.

GPU-based real-time rendering method which can convert an input image into a Chinese ink painting appearance automatically. One example of our image-based rendering approach is shown in Figure 1.

Our contribution is a novel framework to render an input image into Chinese ink painting style automatically. We employ the saliency map to get the regions which may attract the attention of human observers. A simple yet efficient method to simulate the ink diffusion process is proposed. We also propose a black-and-white enhancement method to make the results more conform to Chinese ink painting style. Different from previous approaches, the entire rendering process of our method is implemented on a Graphics Processing Unit (GPU), enabling a real-time feedback. Even casual users can use images to create Chinese ink paintings easily and automatically. Our approach is about 200 to 1000 times faster than Wang et al.'s image-based Chinese ink painting rendering algorithm [3].

2 Related Work

A lot of research has been carried out in simulating Chinese ink painting in the last decade. They can be roughly classified into two categories: non-image-based and image-based.

Most of the previous works belong to the first one. These methods mainly focus on the modeling of brushes or stokes and use them to design human-assisted painting systems. Methods in this category generally try to generate realistic strokes on canvas [4][5][1][6][2][7][8]. Some approaches employ texture mapping or specific nonphysical technologies to improve the rendering performance. Others emphasize on the realistic results and employ complex mathematical or physical models. Zhang et al. [7] presented a simple behavioral model of water and ink particles based on a 2D cellular automation computational model and it is applied to render Suibokuga-like 3D trees. Lee [8] proposed a particle method to render oriental black ink paintings with realistic diffusion effects. Kunii et al. [9] proposed a multidimensional diffusion model to improve the diffusion effect. Guo et al. [10] provided an approach of diffusion with fiber structure modeling. Chu et al. [2] raised an approach for simulating ink diffusion based on the lattice Boltzmann equation (LBE) method and they developed a digital painting system with various realistic ink dispersion effects. These methods can create good results, however, non-image based approaches have the limitation in that they usually need human assistance. That is, people need to draw themselves in order to get nice painting results, which may be difficult for users who do not have painting skills.

Image-based approach is the second category to create Chinese ink paintings. In the early stage, algorithms in this category were generally picture retouching algorithms. To simulate Chinese ink painting styles, these methods applied user-defined patterns or texture maps to the reference image [11][12]. Hand-made effects were imitated by utilizing some brush stroke texture primitivities. Later in 2007, Wang et al. [3] presented a novel approach for image-based color ink diffusion rendering (IBCIDR).

This approach adopted a physical model to simulate ink diffusion and Chinese ink paintings. Although it can produce good results, it has the limitation in that it is slow to render. As reported in their paper, it took 259 seconds to synthesize an image with size 590×520 pixels on a personal computer with an AMD Athlon XP 2000+ 1.7GHz CPU, 512MB RAM, and a video card with TNT2 Model64 AGP bus and 32MB video RAM.

Other related works are watercolor painterly rendering and saliency maps. Watercolor is one of the art forms similar to Chinese ink painting, in which the paints are made of pigments suspended in a water soluble vehicle. As in Chinese ink painting simulation, it consists of non-image-based methods such as works of Small [18] and Curtis et al. [13], and image-based methods such as works of Hays et al. [14] and Bousseau et al. [15]. The saliency map is a map that represents visual saliency of a corresponding visual scene, and it is usually employed to extract important regions from an image for selective rendering. It uses features like color, luminance and contrast to measure the importance value. Itti et al. [16] first proposed the classic saliency model. They computed a saliency map by combining information from center-surround mechanisms. Lee et al. [17] introduced the idea of mesh saliency as a measure of regional importance for meshes. Saliency map can be used to guide the abstraction of images and videos [19].

3 Rendering Algorithms

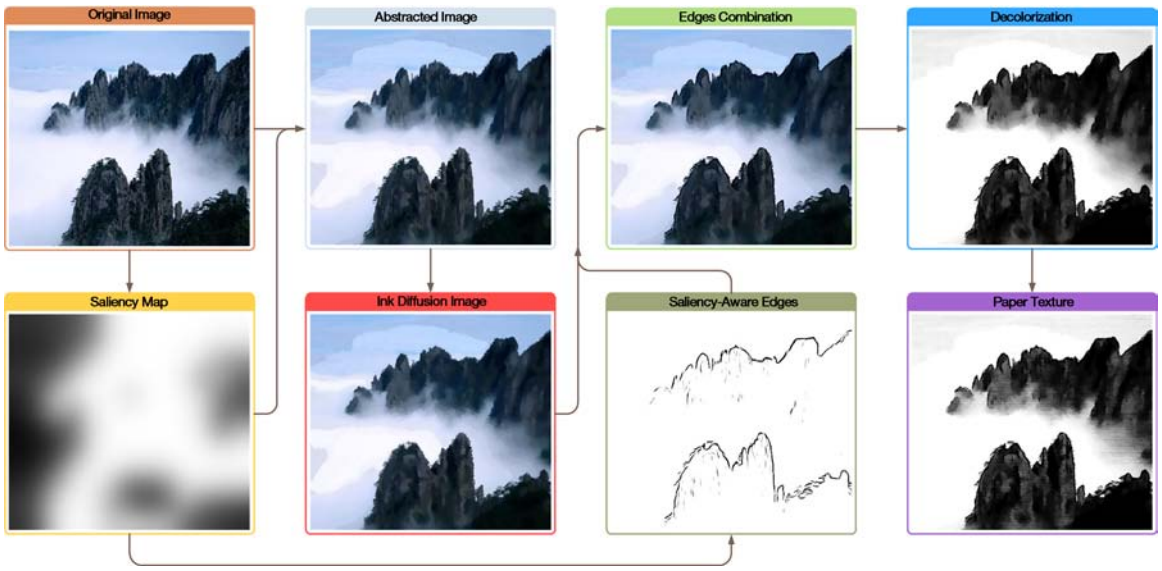


Fig. 2 The workflow of our image-based Chinese ink painting rendering algorithm.

3.1 Workflow Overview

We aim to develop a system that can automatically convert an input image into Chinese ink painting appearance. The workflow of our model is illustrated in Figure 2. We first calculate the saliency map of the input image on the GPU. With the help of the saliency map (or the composite map, which will be discussed in the optional processing), we obtain the abstracted image and saliency-aware edges. Ink diffusion is then performed on the abstracted image. After combining the ink diffused image with the edges, we decolorize the combined image and perform the pigment granulation. In this way, a result image with Chinese ink painting style is created. Detailed algorithms are described below.

Fig. 3 Image before (left) and after (right) abstraction.

3.2 Saliency Map

Freehand Chinese ink painting is an important painting style in traditional Chinese paintings. It is characterized by simple and bold strokes intended to represent the exaggerated likenesses of the objects. In such paintings, artists emphasize the main body of the scene and weaken the details. On the other hand, photographs usually contain rich details because they are realistic. To retain the important part and neglect the unimportant part, we introduce the saliency map to perform the selective rendering. The saliency map also helps remove unwanted lines and dots generated in the edge detection process.

Saliency map is commonly used in the computer vision community. A saliency map tries to describe elements of a visual scene which is likely to attract the attention of human observers. Given an input image, the map encodes for saliency at every location in the visual input. Researchers have developed several saliency models [16] [20] [21]. Among them, Itti' model is based on the theory of human attention. It computes many low-level features and then combines them into a master saliency map. Theoretically, it is more likely to present the objects that the painters will depict. In addition, compared to other visual saliency models, Itti's model is more appropriate to parallelize. Therefore, we employ Itti' saliency model in our implementation.

3.3 Image Abstraction

Abstraction is a distinctive characteristic in freehand Chinese ink paintings where details are generally ignored. In image processing, image abstraction can be used to simplify regions of low contrast while enhancing high contrast regions. Therefore, we incorporate the image abstraction as a step in our framework as it can significantly reduce details in the input image. We use saliency-aware image abstraction algorithm [19] to abstract images. Figure 3 shows a sample input and its abstraction result.

3.4 Ink Diffusion

Traditional Chinese ink paintings are drawn on *Xuan* paper. *Xuan* paper is a kind of absorbent paper. Ink diffusion will arise when people draw on it. Thus, ink diffusion is an essential part of ink painting simulation. We will try to reflect this characteristic in our system. To simulate the ink diffusion of watercolor or Chinese ink paintings, lots of methods adopt physically-based models to achieve good results. For example, Wang et al. [3] presented a physically-based color ink diffusion model by solving differential equations. However, physically-based models usually involve high computational cost, and do not provide real-time feedback. In this paper, we propose a simple yet efficient approach to simulate the ink diffusion. Our non-physically-based model can attain similar simulation results with much less computational cost.

The algorithm is described as follows. We first use Equation (1) to spatter the image derived from the image abstraction process:

$$P(x, y) = P(x + r_x, y + r_y), \quad r_x, r_y \in [-R, R], \tag{1}$$

where $P(x, y)$ denotes the pixel at point (x, y) in the image, r_x and r_y are two random numbers in x and y axis, respectively. r_x and r_y are limited by a spattering radius R. Replacing pixels by their near neighbors simulates the motion of ink particles in Chinese ink painting. However, the spattered image looks a bit strange because of the discontinuity between pixels. Therefore, we then employ a median filter to suppress the noises brought from the spattering filter:

$$P(x, y) = median\{P_1, P_2, \cdots, P_{n^2}\}, \tag{2}$$

where P_1, P_2, $\cdots$, P_{n^2} denote pixels in the $n \times n$ kernel whose center locates at (x, y). This filtering operation replaces each pixel in the image with the median of neighboring pixels in the kernel, and it can be regarded as a blurring filter of the spattered image. Depending on the size of the input image, n is set to 3, 4, 5. Although our ink diffusion model is non-physically-based, it renders visually pleasing results. Figure 4 shows our ink diffusion results using spattering filters with different radius R and the median filter. In these figures, we use $n = 5$. To get a desired result, the value of R must be chosen carefully. When $R = 5$, there is few diffusion because the spattering is largely smoothed by the median filtering (See Figure 4(b)). When $R = 20$, the figure looks like a mess because the spattering radius is much larger than the radius of the median filter (See Figure 4(d)). When R approximately equals to $2n$ (Here $R = 10$), the diffusion effect is the most similar to the real diffusion in Chinese ink painting (See Figure 4(c)). Empirically, we set $R \approx 2n$.

(a) Input image. (b) Result with $R = 5$. (c) Result with $R = 10$. (d) Result with $R = 20$.

Fig. 4 Ink diffusion results using spattering filter with different radius R and the median filter.

To implement the above ink diffusion process on the GPU, an auxiliary texture is required in the pixel shader – the random number texture. This texture has the same size as the input image and each pixel is assigned a pair of random numbers (r_x, r_y). These random numbers are generated in the CPU in the preprocessing stage. In the shader processing, Equation (1) is used to spatter pixels within the defined radius. In our current implementation, the value of radius R is proportional to the number of pixels M in the input image, and $R = 0.00001 \times M$. That is to say, for an 800×600 image, R is about 5.

Compared to other blur filters, the median filter is more difficult to implement on the GPU. As the filtering operation processes every pixel in images, the efficiency of the median calculation is a critical factor in determining how fast the algorithm can run. Sorting is certainly computationally expensive and will seriously slow down the overall rendering processing. Fortunately, the median filter can be implemented on the GPU by several compare and swap operations when kernels are small [22]. The algorithm is ideal for streaming parallel architectures and we employ it in our framework to achieve a fast speed.

3.5 Edge Line Map Computation

Flow-based difference-of-gaussian (FDoG) filter [23] is used to extract clean lines in our system. Traditional difference of Gaussian edges do not reassemble straight line and curve segments well. FDoG,

which is based on flow and is more noise insensitive, works around this limitation and can produce more coherent lines. FDoG first builds an Edge Tangent Flow (ETF) field $v(x)$ iteratively, then it uses the flow-based kernel instead of the isotropic kernel to sample points. The FDoG filter is formulated as followings:

$$D_g(x) = \frac{1}{d_g} \sum_{t \in [-\beta, \beta]} (G_{\sigma_c}(t) - \rho \cdot G_{\sigma_s}(t)) I(l(t, x)), \tag{3}$$

$$D_e(x) = \frac{1}{d_e} \sum_{s \in [-\alpha, \alpha]} G_{\sigma_m}(s) I(c(s, x)), \tag{4}$$

$$d_g = \sum_{t \in [-\beta, \beta]} (G_{\sigma_c}(t) - \rho \cdot G_{\sigma_s}(t)), \tag{5}$$

$$d_e = \sum_{s \in [-\alpha, \alpha]} (G_{\sigma_c}(t), \tag{6}$$

where α and β are parameters for controlling the size of the flow-based kernel, σ_c and σ_s control the sizes of the center interval and the surrounding interval. After D_e is obtained, edges is then smoothed by the following function:

$$L(x) = \begin{cases} 1, & D_e(x) > 0 \\ 1 + \tanh(\varphi_e \cdot D_e(x)), & \text{otherwise} \end{cases}, \tag{7}$$

where φ_e is a parameter to control the line sharpness. More technical details about extracting edges using FDoG can be found in [23].

After the edges are extracted, we calculate the needed edges by utilizing the region of interest (ROI) function so that only lines depicting important information are kept. With the saliency map, we derive an ROI function and use it to weaken the background edges and eliminate the details in order to perform selective rendering. The resulting edge line map is defined as: $L_{new} = L_{ori} * S$, where and S represents the saliency value (or the value in the composite map, L_{ori} stands for the lines extracted by paralleled FDoG and L_{new} stands for the final result of edge line map.

3.6 Decolorization

Since most of the traditional Chinese ink paintings are in black and white, the decolorization is employed in our framework. A simple decolorization can be implemented by converting the input color image into a monochrome one. Though it works, it fails to represent some characteristics in Chinese ink paintings.

In traditional Chinese ink painting, artists usually grind their own inkstick over an inkstone to obtain ink. Typically, objects in a Chinese ink painting are drawn by deep black, while free space remains white. However, most images available online or photos snapped during our daily life possess many details. After the decolorization step, the resulting image looks quite gray instead of white and black, and this does not conform to real Chines ink paintings. Therefore, to mimic the characteristics in real paintings, we add an additional black-white enhancing step. We artificially increase the difference between black and white for the decolorized image. That is, to make the white even whiter and the black even blacker. The black-white enhancing step can been seen as a special form of gamma correction to increase the contrast, and it is implemented by the following formula:

$$C' = \begin{cases} 0, & C < L_{min} \\ (C - L_{min})/(L_{max} - L_{min}), & L_{min} \leq C \leq L_{max}, \\ 1, & C > L_{max} \end{cases} \tag{8}$$

where C is the original color of a pixel and C' is its enhanced color, L_{min} and L_{max} are two user-defined thresholds to control the enhancing. Figure 5 shows the enhanced images using different L_{min} and L_{max}. Empirically, $L_{min} = 20$ and $L_{max} = 200$ work well in our implementation.

(a) Input image. (b) Without enhancing. (c) Light enhancing. (d) Heavy enhancing.

Fig. 5 Rendering with different L_{min} and L_{max} presents different results. Figure 5(b) shows the result without enhancing. Figure 5(c) uses $L_{min} = 20$ and $L_{max} = 200$ and Figure 5(d) uses $L_{min} = 75$ and $L_{max} = 150$.

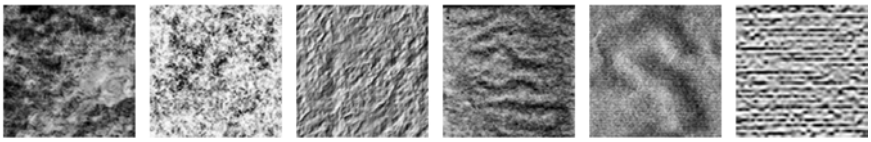


Fig. 6 Sample paper textures.

3.7 Paper Texture

Due to the features of *Xuan* paper mentioned above, there are varieties of ink diffusion effects in Chinese ink paintings. Moreover, the *Xuan* paper provides the "granulation" in paintings. The granulation is caused by the deposition of pigments on paper or canvas. Paper texture and the transparency of ink influence the granulation effect very much.

Because of the similarity between Chinese ink paintings and western watercolors, pigment granulation method for watercolor [15] is also helpful to simulate the *Xuan* paper effects in Chinese ink painting rendering. It transfers a color image C to color C' according to a texture. Let $P \in [0, 1]$ denote the gray-scale pigment in the texture. We utilize the following revised function to produce the granulation effect on the image:

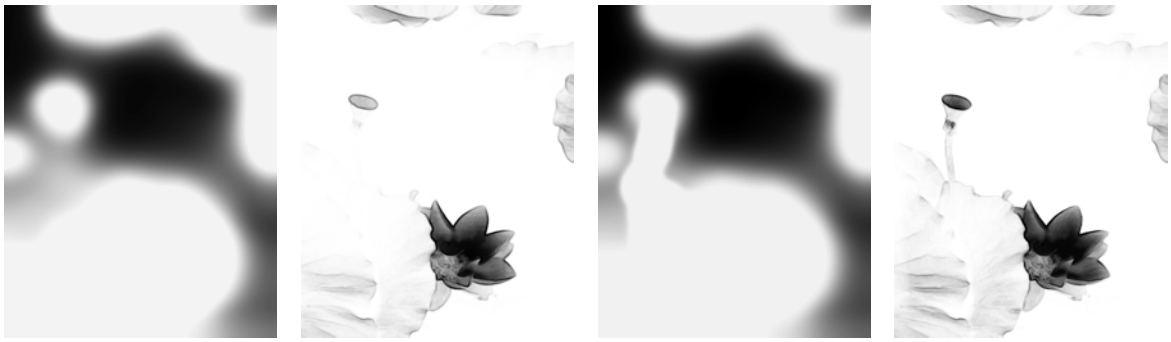
$$C' = C(1 - \omega_g(1 - C)(P - 0.5)). \tag{9}$$

It is slightly different from the equation proposed by Bousseau et al. We add ω_g, which represents the weight for granulation effect, to their equation. Compared to their method, ours is much more flexible for various painting styles. In our implementation, we set $\omega_g = 0.8$. Experiments show that this value is suitable for most paintings in Chinese style.

Paper textures can be obtained through many different ways. One simple way is to scan the paper against a dark background, as suggested by Chu [2]. Another way is to use computer-generated textures. Some sample paper textures are shown in Figure 6.

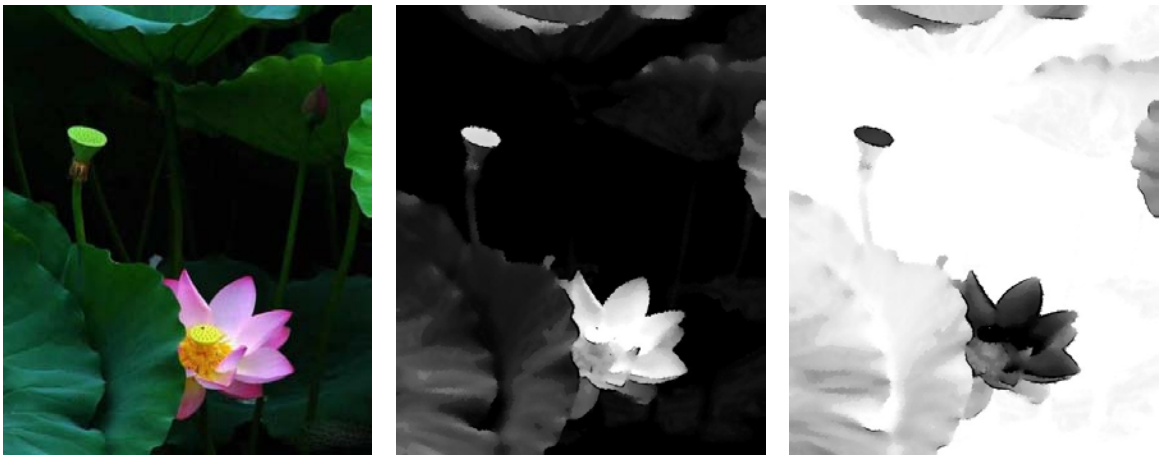
3.8 Optional Processing

There are varieties of Chinese ink painting styles and they depict various objects. A fixed automatic system cannot always render appealing results. Therefore, some optional steps are proposed in our framework. By making use of these options, better Chinese ink painting paintings can be rendered for some examples. One optional step is to segment the foreground from the input image using grabcut [24]. The extracted foreground is then combined with the saliency map to generate a new composite map, which more accurately reflects the visual importance of pixels for Chinese ink painting. Figure 7 shows the difference between using the saliency map and the composite map. Another optional step is inversion. Since we use color images as input, simply decolorizing them may result in white foreground and black background. This may contradict the rules of Chinese ink paintings. For these kind of examples, we provide the image inversion as an option. Figure 8 shows a typical example to achieve a better result by using the inversion operation.



(a) (b) (c) (d)

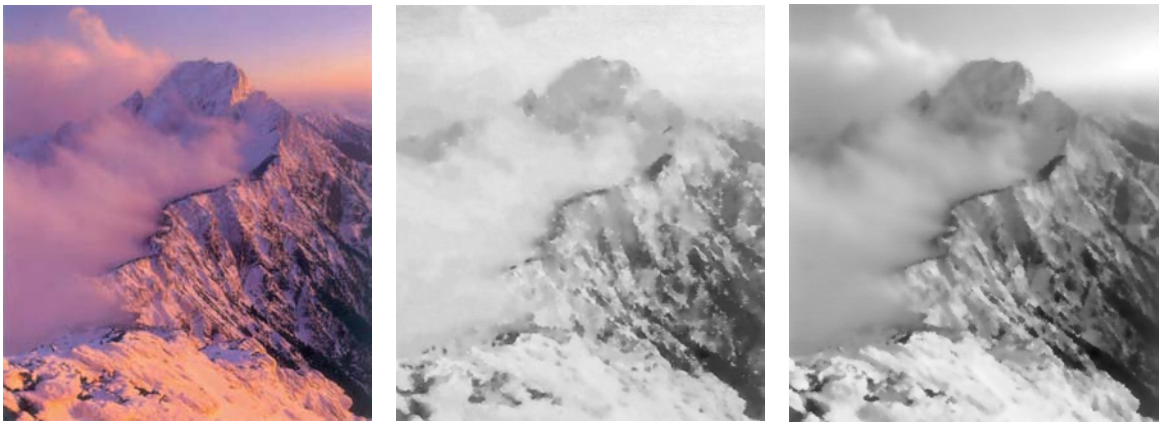
Fig. 7 Upper: the saliency map (a) and its corresponding result (b). Bottom: the composite map (c) and its corresponding result (d).



(a) (b) (c)

Fig. 8 The inversed image in Figure 8(c) looks better than the one in Figure 8(b).

4 Experimental Results



(a) (b) (c)

Fig. 9 Comparison between IBCIDR and our approach. The input image (a), the synthesized result by IBCIDR (b) and our ink diffusion result (c).

We have run our rendering system using C++ and DirectX 10 on a 2.0GHz Intel Core 2 Duo T7250 CPU and an NVIDIA GeForce GT8600M GPU with 256M memory size. Techniques described above are all based on pixel-level operations, which means that our algorithm can take full advantages of high-level parallelism. In our framework, all the steps besides the optional grabcut step are implemented on the GPU with High Level Shading Language (HLSL). Combining all the steps together, the system produces the final Chinese ink painting from an input image automatically in real time. Figure 10 and Figure 11 shows some results rendered by our system. The resolution of the input image and the resulting image is the same.

Figure 9 compares the rendering result of our ink diffusion with IBCIDR (Image Based Color Ink Diffusion Rendering) [3]. Figure 9(a) is the original image, Figure 9(b) is the result of IBCIDR in the black-white style, and Figure 9(c) is our ink diffusion result. The objective criterion for a Chinese Ink Painting consists of three characteristics. First, the image is similar to East Asian calligraphy; only black ink is used. Second, the granulation of pigments will arise in the ink diffusion process by interactions between ink and absorbent Xuan paper. Third, abstraction is a distinctive characteristic in freehand Chinese ink paintings where details are normally ignored. Both the results of our method and IBCIDR [3] meet the above three characteristics. Experimental results show that our non-physically-based approach can produce comparable results with their method.

IBCIDR takes several hundred seconds to render an image with 0.25 Mega pixels. For example, for an image with size 590×520 pixels, the total execution time is 256 seconds with an AMD Athlon XP 2000+ 1.7GHz CPU. However, because of our simple ink diffusion algorithm and our GPU acceleration, we can render the Chinese ink painting in real time. We have tested the performance of our pipeline using different NVIDIA GeForce GPUs and images with different sizes, and the time statistics are reported in Table 1. For an image with size 640×480 pixels, the rendering time is 0.063 second with a cheap NVIDIA GeForce GT8600M GPU. Even taking the the different CPUs into account, our rendering approach is still about 200 to 1000 times faster than IBCIDR.

Table 1 Rendering performance in seconds.

Image Resolution	GTX 280	GT 240	8600M GT
320×240	0.004	0.009	0.027
480×360	0.007	0.017	0.044
640×480	0.011	0.027	0.063
800×600	0.019	0.041	0.090
960×720	0.031	0.059	0.125
1280×960	0.048	0.099	0.448

5 Conclusions and Future Work

We have presented a novel image-based framework to render Chinese ink paintings automatically. Our approach first employs the saliency map to abstract images and selectively extract lines. Then, we use our ink diffusion, decolorization and paper pigment granulation algorithms to simulate Chinese ink painting. Our approach has the following features:

- Image-based conversion. Different from most of the non-image-based approaches, our system takes images as input and convert them into Chinese ink painting style.
- Automatic synthesis. Our system can convert input images automatically. Even casual users can create Chinese ink paintings easily.
- Selective rendering. Our framework performs selective rendering under the guidance of the saliency map, and this leads to pleasing results.
- Real-time performance. Because we propose a simple yet efficient method to simulate the ink diffusion process instead of the computational intensive physical simulation, and all the steps are implemented in highly parallel GPU, our method is much faster than previous image-based synthesis methods.
- Black-and-white enhancement. Due to the black foreground and white background characteristic in Chinese ink paintings, we add a black-and-white enhancement step in our framework. It helps simulate the real appearance of Chinese ink paintings more faithfully.

Several issues remain to be addressed in future work. First, we would like to apply our technique to render a video into Chinese ink painting style. Primary tests show that our approach cannot be extended to videos directly. The biggest problem is the noticeable temporal aliasing effect because of the spattering operation. A pixel at (x, y) in one frame may move to a different position in the next frame in our ink diffusion operation. Bousseau et al. [25] proposed a method to preserve the coherence of paper texture but not the coherence of ink diffusion texture. Second, our rendering algorithm may produce poor results for some photos. As traditional Chinese ink paintings seldom depict scenes other than plants, mountains, villages and traditional architectures, using these kind of photos as input may not produce desirable results. We would like to expand the applicable scope of our method in future. Third, we are interested in extending our work to 3D volumes as most of the filters described in our approach can be modified for volumes by adding the z axis.

References

1. Chu S, Tai C (2004) Real-time painting with an expressive virtual chinese brush. IEEE Comput Graph Appl 24(5):76-85
2. Chu S, Tai C (2005) MoXi: real-time ink dispersion in absorbent paper. ACM Trans Graph 24(3):504-511
3. Wang C, Wang R (2007) Image-based color ink diffusion rendering. IEEE Trans Vis Comput Graph 13(2):235-246
4. Strassmann S (1986) Hairy brushes. In: Proc the 13th annual conference on computer graphics and interactive techniques. ACM, New York, pp 225-232
5. Hsu S, Lee I (1994) Drawing and animation using skeletal strokes. In: Proc the 21st annual conference on computer graphics and interactive techniques. ACM, New York, pp 109-118
6. Saito S, Nakajima M (1999) 3D physics-based brush model for painting. In: Proc SIGGRAPH '99 conference abstracts and applications. ACM, New York, pp 226
7. Zhang Q, Sato Y, Takahashi J, Muraoka K, Chiba N (1999) Simple cellular automaton-based simulation of ink behaviour and its application to suibokuga-like 3D rendering of trees. Vis and Comput Animation 10(1):27-37
8. Lee J (2001) Diffusion rendering of black ink paintings using new paper and ink models. Comput Graph 25(2):295-308
9. Kunii TL, Nosovskij GV, Hayashi T (1995) A diffusion model for computer animation of diffuse ink painting. In: Proc computer animation '95. Geneva, Switzerland, pp 98-102
10. Guo Q, Kunii TL (2003) "Nijimi" rendering algorithm for creating quality black ink paintings. In: Proc computer graphics international '03, pp 152-159
11. Yu J, Luo G, Peng Q (2003) Image-based synthesis of chinese landscape paintings. Journal of Computer Science and Technology 18(1):22-28
12. Wen S, Shih Z, Chiu H (1999) The synthesis of chinese ink painting. In: Proc national computing symposium, pp 461-468
13. Curtis J, Anderson E, Seims JE, Fleischer W, Salesin H (1997) Computer-generated watercolor. In: Proc SIGGRAPH'97. ACM, New York, pp 421-430
14. Hays J, Essa I (2004) Image and video based painterly animation. In: Proc international symposium on non-photorealistic animation and rendering. ACM, New York, pp 113-120
15. Bousseau A, Kaplan M, Thollot J (2006) Interactive watercolor rendering with temporal coherence and abstraction. In: Proc international symposium on non-photorealistic animation and rendering. ACM, New York, pp 141-149
16. Itti L, Koch C, Niebur E (1998) A model of saliency-based visual attention for rapid scene analysis. IEEE Trans Pattern Anal Mach Intel 20(11): 1254-1259
17. Lee CH, Varshney A, Jacobs DW (2005) Mesh saliency. In: Proc SIGGRAPH'05. ACM, New York, pp 659-666
18. Small D (1991) Simulating watercolor by modeling diffusion, pigment, and paper fibers. SPIE Image Handling and Reproduction Systems Integration 1460(15): 140-146
19. Zhao H, Mao X, Jin X, Shen J, Wei F, Feng J (2009) Real-time saliency-aware video abstraction. Vis Comput 25(11): 973-984
20. Torralba A, Castelhano M, Oliva A, Henderson (2006) Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. Psychological Review 113(4): 766-786
21. Harel J, Koch C, Perona, P (2007) Graph-based visual saliency. In: Proc advances in neural information processing systems. pp 545-552
22. McGuire M (2008) A fast, small-radius GPU median filter. ShaderX6: advanced rendering techniques, Charles River Media, pp 165-174
23. Kang H, Lee S, Chui CK (2009) Flow-based image abstraction. IEEE Trans Vis Comput Graph 15(1): 62-76
24. Rother C, Kolmogorov V, Blake A (2004) "Grabcut": interactive foreground extraction using iterated graph cuts. ACM Trans Graph 23(3): 309-314
25. Bousseau A, Neyret F, Thollot J, Salesin D (2007) Video watercolorization using bidirectional texture advection. ACM Trans Graph 26(3): article 104

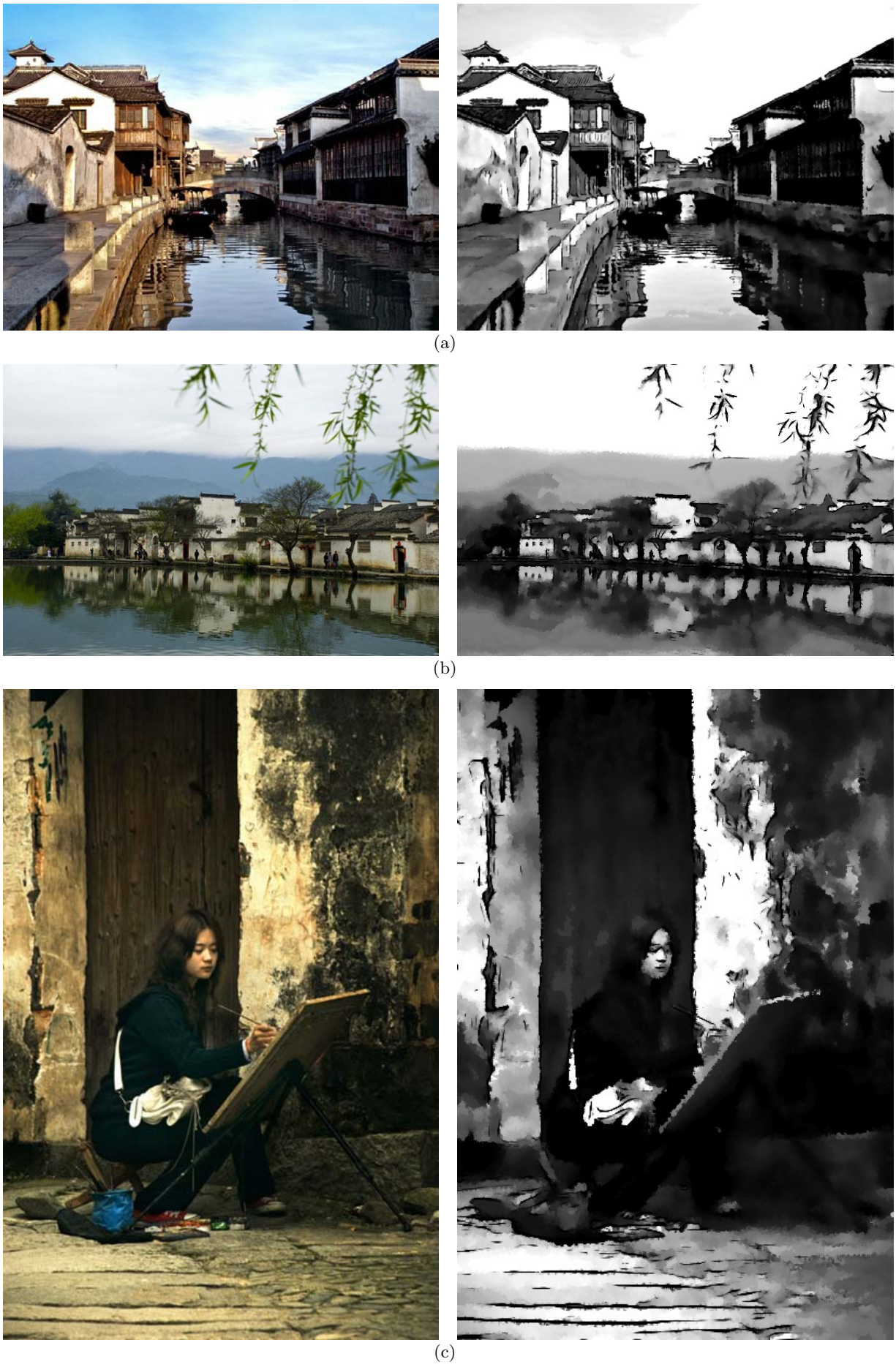
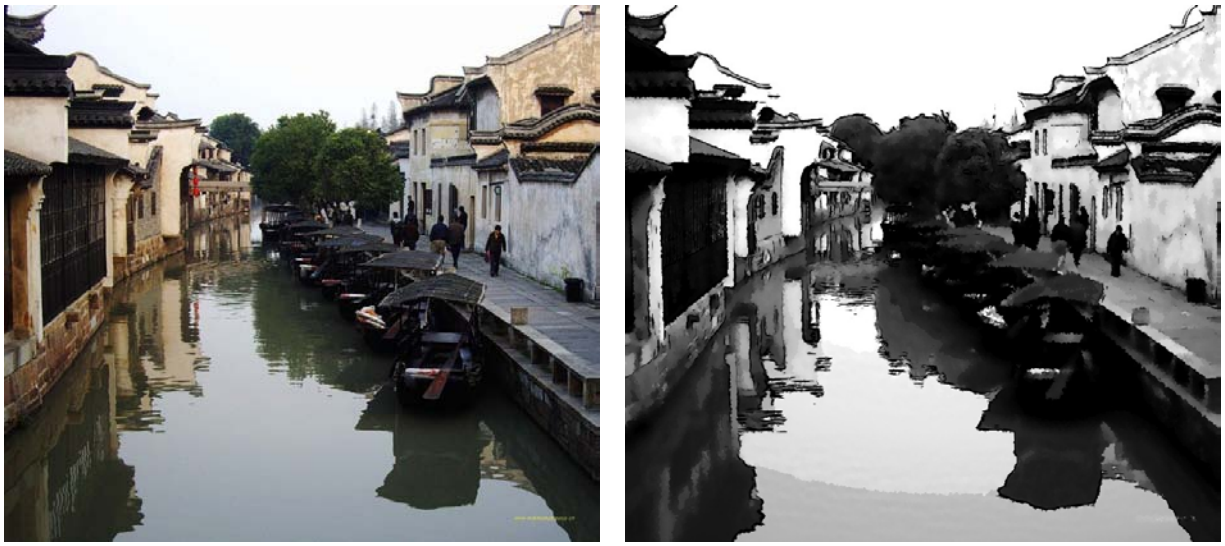


Fig. 10 Our rendering results. The input images (left) and its ink painting results (right).

(a)



(b)

(c)

Fig. 11 More rendering results. The input images (left) and its ink painting results (right).