“Nijimi” Rendering Algorithm for Creating Quality Black Ink Paintings

Qinglian Guo* Tosiyasu L. Kunii**
*Kanazawa Institute of Technology
*kaku@infor.kanazawa-it.ac.jp
**tosi@kunii.com

Abstract

The paper presents an interactive painting system for generating high quality and artistic calligraphy characters and black ink paintings. The system is based on our original algorithm that is unique in physically simulating the dynamic diffusion of liquid ink into absorbent painting paper. The algorithm creates remarkably realistic strokes efficiently. For rendering 20 strokes in different size, it takes only 30 seconds on a personal computer, fast enough for interactive performance. By using this system, the users can create realistic strokes with delicate “nijimi”-rendering, “kasure”-rendering effects, which are essential to black ink painting and modern calligraphy.

1. Introduction

In recent years, people show great interest in non-photo-realistic systems, such as picture re-touching software, illustration generators, and digital painting systems. These systems have abundant functional filters, infinite range of colors, and various painting brushes to represent the materials and rendering effects of fine arts.

This paper presents a new system for generating quality black ink paintings and artistic calligraphy words. Black ink paintings differ from oil- and watercolor-paintings in several aspects. Firstly, a black ink painting usually consists of a few well-placed strokes. Overlapping of strokes is not often and white spaces are necessary and meaningful. Secondly, in black ink paintings the emphasis is in the quality of each stroke as well as in that of the entirety. Artists usually draw each stroke in one sweep without re-touching. In the case of oil paintings or watercolors, it is difficult to clearly distinguish one stroke to another, artists often use strokes work as a group and overlay strokes to get a desired form or color.

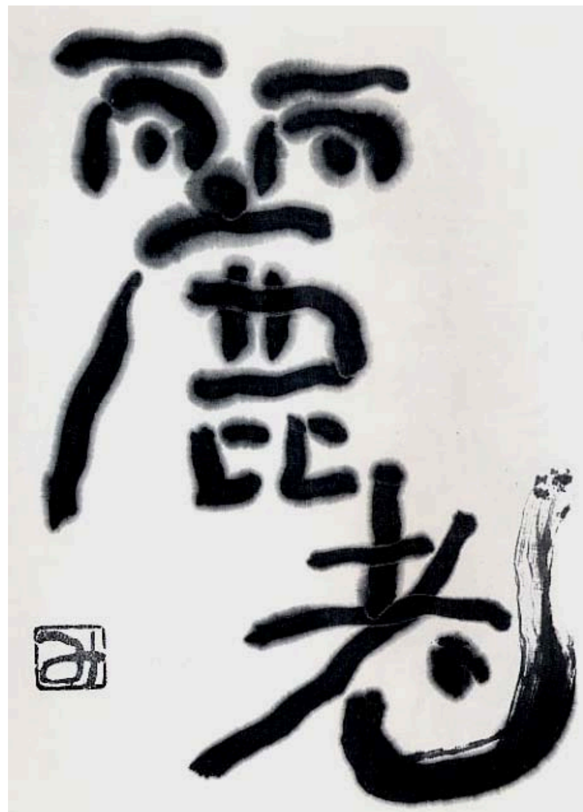


Fig.1 "Lei-rou (getting old gently)" A copy of the original art work drawn by a prestigious Japanese artist, Mitsuo Aida.

For drawing high quality and impressive strokes in one sweep, many techniques have been developed traditionally. The most remarkable techniques include the change of gray tones, diffusion of ink into absorbent painting paper (called "nijimi-rendering effect"), and brush un-touching (called "kasure-rendering effect"). Our research has been armed to simulating the nijimi rendering effect so as to create high quality strokes with rich and delicate variations.

Proceedings of the Computer Graphics International (CGI’03)
1530-1052/03 $17.00 © 2003 IEEE

Nijimi-rendering is caused by the diffusion of liquid ink into absorbent paper. It is a natural subtle and forked blurring area around the boundary of an original stroke to which liquid ink was directly applied. A nijimi stroke is consisted of two areas: the original stroke area and the surrounding nijimi area. The nijimi area illustrates delicate variation in pixel intensity and irregular heathery texture. Physically, nijimi is a complex hydrodynamic percolation phenomenon. It is determined by the fabric structure of painting paper and special characteristics of liquid ink.

Fig.1 gives an example of practical modern art of calligraphy, which has nijimi and kasure-rendering on its strokes. If there are none of them, the art is out off strength and loses its beauties. With previous published painting systems to the best of the authors' knowledge, it is difficult to generate such quality brush strokes.

Fig. 2 gives an example created by using our system. It is an imitation of Fig. 1. For defining the outlines of the strokes in Fig.2, Aida's work was used as a reference. Our system has taken only about 30 seconds in rendering the whole strokes. We have not use any other editing, picture re-touching, or computer-assisted techniques. A comparison has proved that our strokes are quite realistic.

1.1 The related previous works

There have been many researches on digital representation of brush strokes. Greene's drawing prism [6] was the most early interactive devices for changing a physically draw action to a digital stroke in computer.

Strassmann [12] proposed the first mathematical model for representing brush strokes. His model made it possible to flexibly control stroke shapes and to render strokes according to the distribution of liquid ink on brushes. Other methods [1, 8, 10] using Bezier functions to represent strokes built primitive foundation for synthesizing strokes in arbitrary shapes. These previous researches had great influence on the succeeding approaches on synthesizing calligraphy words [4, 14], Chinese landscape paintings [13], and 3D strokes [11].

Fractal Design's Painter [2] and Hertzmann's painting system [7] provided several types of bitmap brushes and paper textures, which represent the roughness of paper surface. These bitmap brushes and paper textures are considered a good solution for watercolor- and oil-paintings. However, we have conducted many experiments by using the software based on these previously published researches to draw black ink painting, but they have failed to produce a satisfactory result. The reason is due to the painting materials (black ink and absorbent paper) that are quite different from what used in oil- and watercolor- paintings. Another reason is due to the limitation of using bitmap based strokes.

Fig.2 An Fig. 1, An imitation of Fig.1, created by using our painting system

Kunii and Nosovskij [9] proposed a multi-dimensional diffusion method for simulating the specific features of intensity distribution in nijimi images. Such accurate simulation is important for investigating the physical phenomenon, imposing high calculation cost for rendering strokes in complex shapes. Hence, it is not for real time rendering.

A basic idea of synthesizing nijimi-rendering based on a generated fiber mesh was proposed early by Guo and Kunii [3]. Their weakness was realized not being able to evaluating the interaction between fiber mesh and liquid ink diffusion appropriately, and their results were far away from realistic.

This paper presents our novel and original algorithm for representing the nijimi rendering effect. The algorithm is remarkable in its results to produce highly realistic nijimi images. This is made possible by physically simulating the dynamic process of liquid ink flowing in digital fiber meshes. The algorithm is also very efficient. On a personal computer, it needs an average 1.5 to 2 seconds to render one stroke. This is fast enough for carrying out interactive performance. Another remarkable point is the composition of nijimi and kasure in rendering one stroke. We render a stroke with nijimi at the starting part, as if a virtual brush moving over the stroke, and change to render the stroke with kasure. This results in natural and dynamic strokes that have not been obtained by using any other software. Here we will focus our description only on the nijimi rendering algorithm.

2. Drawing a black ink painting with using the present system

We provided a simple interface to the users for inputting strokes, selecting rendering mode, and setting values for rendering parameters. We use a mouse device to pick control points to define a boundary for a stroke. We use the edit function to move/remove control points so as to get an exactly desired formation. Each stroke is defined by two boundary wire-lines with the picked control points on them. As shown in Fig.3-(a), we defined 12 strokes for painting a dog.

The next step is to specify rendering mode and set parameters for each stoke. Rendering parameters include paper absorbency, paper type, ink density, ink quantity, drawing speed, and brush size. We have selected to use the only nijimi mode to render the dog's back for showing its soft hairs. For the dog's chest line and collar, we have used the composition mode of nijimi and kasure. Kasure strokes give more strength and activity to the dog. Fig. 3-(b) shows the midterm state when rendering strokes, and Fig. 3-(c) is the final painting of a lovely dog.

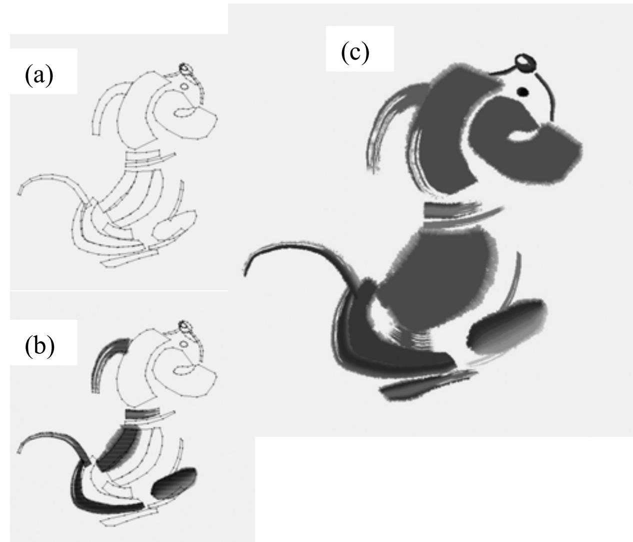


Fig.3 Use our system to create a black ink painting

3. Modeling the fibrous structure of absorbent painting paper

As Guo and Kunii described [7], "from the microscopic pictures of painting paper, we can see that its detail structure is a fibrous mesh consisting of irregularly distributed fibers". The fiber distribution is considered homogeneously randomness. This means the fiber density and the average fiber orientation is globally uniform, but the local fiber density varies irregularly from place to palace. The irregular variation makes nijimi-rendering look natural and dynamic.

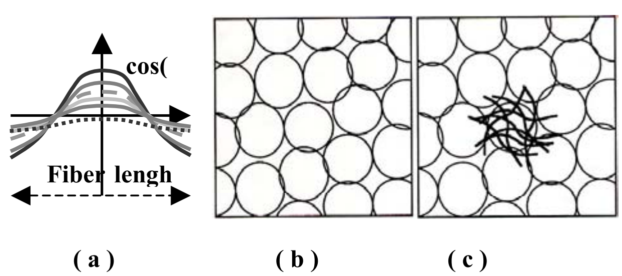


Fig. 4 Generate the fiber meshes that represent the micro-structure of painting paper. (a) define a fiber; (b) Evenly subdivide a large field into small circle regions; (c) Distribute fibers in each small region.

We have represented a single fiber as a curve-line determined from simple mathematic functions (Fig. 4-(a)).

A direct way to generate a fiber mesh is to randomly distribute a lot of such fibers on a large 2-dimensional field. However, it is difficult to create a global uniform distribution if only a random function is used, since a random function leads to a Poisson distribution that is not uniform. We solved this problem by using an original algorithm, which contains the following core procedures:

(1) Subdivide the large field into small regions of radius R.(Fig. 4-(b)).

(2) Decide a center point *(x_0, y_0)* for each small region, and distribute *D* fibers in each small region. For a particular fiber, its position (f_x, f_y) and orientation f_θ are determined by

$$f_x = x_0 - eR + random - fun(f_x, 2eR)$$

$$f_y = y_0 - eR + random - fun(f_y, 2eR)$$

$$f_\theta = random - fun(f_\theta, w1 - w0)$$

A fiber, defined as in Fig. 4-(a), is then rotated in f_θ angle, and located in the position (f_x, f_y). Here, positional evenness *e* takes a value in [0.1, 1.0], *D* is fiber distribution density, and *random-fun(seed, specified-range)* gives random value in the specified range. By changing the value of *e*, it is possible to concentrate fibers at the center point or diversify fibers over the small region. By controlling the angles *w_0* and *w_1*, it is able to make fibers lie in a narrow range of directions, or lie evenly in all directions from *0* to *360*.

Obviously, these parameters enable the users to introduce intentional control to the generated fiber meshes. Fig. 5 shows some examples: the left-up is much close to the fibrous structures of real painting paper, whereas others are intentionally made different.

(3) While distributing fibers, recording each fiber into a 2D buffer *M*, which is called "fiber-mesh-data".

For a fiber mesh of size *X* by *Y*, we used a fiber-mesh-data buffer of *X*Y* elements. As Fig. 6, for each point P of the fiber mesh, there are two groups of data: fiber structure data $M_1 M_2 \ldots M_8$ and capillary structure data $S_1 S_2 \ldots S_8$. If $P_1 P_2 \ldots P_8$ are the eight neighbor points of P, Mi represents the number of fibers passing *P* and *P_i*, and *S_i* represents whether *P* and *P_i* are connecting to form a capillary tube between the two fibers lying parallel and near enough.

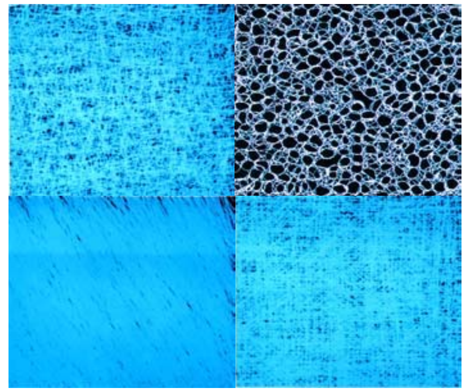


Fig. 5 Examples of computer generated fiber meshes

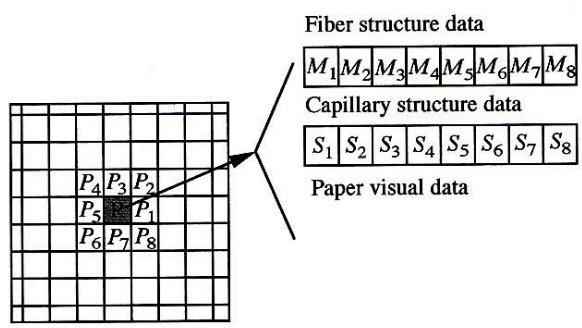


Fig. 6 The definition of the paper data

The capillary structure data *S_i* was determined from *M_i*. We simply assumed that all the capillary tubes formed between interlacing fibers had same radius. When $M_i \geq 2$, we assumed that there is capillary tube between *P* and *P_i*, assign *S_i = 1*; When *M_i < 2*, *S_i* = 0, denoting there was no capillary tube connecting P to *P_i*. The capillary structure data *S_i* is a basis for simulating nijimi diffusion.

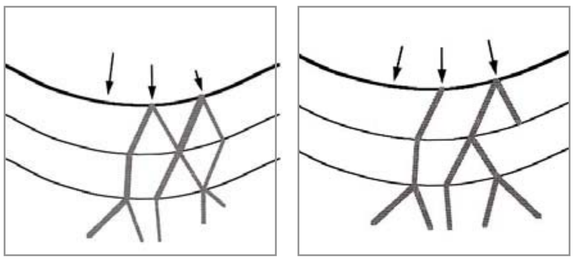
4. The nijimi-rendering algorithm

From the experiments on several types of paper, it is realized that nijimi images have three basic characteristics:

(1) On different types of paper, nijimi image shows different texture that is determined by the fabric structure of paper. Liquid ink flows along the capillary tubes formed between interlacing fibers from one point to another points and this active flowing process decides nijimi image.

(2) Because of the deposition of ink particles on fibers, a gradually decrease in ink density occurs during the diffusion. We refer this as a "filtering" effect.

(3) The absorbency at each point varies delicately depending local fiber density.



(a) **(b)**

Fig. 7 Our simplifications to practical nijimi diffusion. we transform a complex case of ink flowing as in (a) to a simpler case as in (b)

Our nijimi-rendering algorithm is focused on the physical representation of these characteristics, while making several simplifications to practical nijimi diffusion for decreasing calculation cost. We assumed that: 1) at every point of a fiber mesh, the radius of capillary tube was the same; 2) the speed of diffusion is constant. Under the simplifications, we can transform a complex case of ink flowing as in Fig.7-(a) to a simpler case as in Fig.7-(b).

We also used several temporal buffers to assist our implementation.

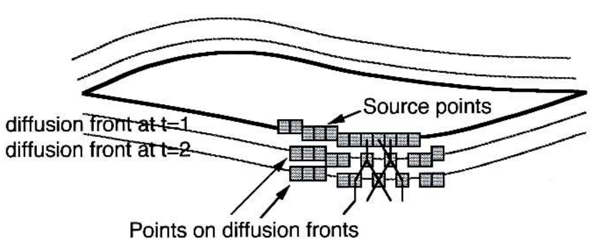


Fig. 8 The illustration shows the source points and the points on a diffusion front.

- **Source point buffer**

The area within the defined stroke boundary is called “source area” which supplies the liquid ink for nijimi diffusion. The pixels on the boundary edge of the source region are called source points. All the source points are recorded in order by using a buffer, called “source point buffer”.

For each source point, the buffer saves its index number, coordinates, the amount of liquid ink kept in it. The amount depends on the distance from the point to the starting point of the stroke, the amount of liquid ink applied to brush, and the width of the stroke.

- **Diffusion front buffers**

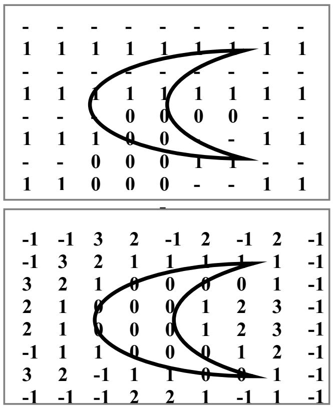
A diffusion front is a collection of points on the profile of nijimi area at a moment. For each point on a diffusion front, its coordinates and absorbed liquid-ink quantity, and the index number of a corresponding source point were saved.

We used two buffers, “current diffusion-front buffer” and “next diffusion-front buffer”. The distance between two succeeding diffusion fronts is one pixel. The time counter t determines a sequence among the diffusion fronts. The source points are set $t = 0$, the first diffusion front is $t = 1$, and so on. The essential idea of simulating ink diffusion is to determine the points on the next diffusion front when $t = k+1$ from which on the current diffusion front when $t = k$.

Fig. 8 illustrates the difference between the source and the diffusion front. The source points succeed one to another, but the pixels of a diffusion front may not, usually separated in space.

- **Paper-state buffer**

We used a paper-state buffer to temporally record the occupation state on the paper-data buffer. The two buffers have the same size $X*Y$ The paper state buffer saves the time counter value, which indicates when a point is occupied. During the implementation of nijimi-diffusion, the paper-data buffer doesn't change, but the paper-state buffer varies each time. Before calculating a new diffusion front, the current diffusion front is saved into the paper state buffer, and the points on the next diffusion front buffer are copied into the current diffusion front buffer. As a result, all of the diffusion fonts are saved in the paper-state buffer. Fig. 9 illustrates the value of time counter recorded in the paper-state buffer before and after a nijimi diffusion process.



(a)

(b)

Fig. 9 The value of time counter recorded in the paper-state buffer before (a) and after a nijimi diffusion (b)

Here, the values of time counter also represent occupation state. At the beginning of nijimi diffusion, all the pixels outside a stroke's boundary are initialized to be *–1*. During nijimi diffusion, some of these pixels are changed to have a time counter value larger than *0*. As a result, only the pixels not occupied by ink remain to have the value of *–1*, such points are assigned the color of paper (white). For a pixel having a value larger than *0*, a gray tone is determined depending on the density and amount of ink absorbed on it.

Our nijimi-rendering model consists of three calculation schemas:

(1) Tree transportation schema determines how liquid ink flows in a fiber mesh. To indicate the points:

(2) Liquid ink can flow from point *P* to point *P′*, if there is capillary tube connecting them, and *P′* has not got any ink yet. We call *P′* the extension point of *P*, and *P* the previous of *P′*. One point can only have one previous point, but it can have more than one extension point.

(3) Liquid ink can flow from point *P* to zero, one, or more than one points.

(4) Point *P* absorbs a maximum amount of liquid ink *Q(p)* before it transports liquid ink to other points.

Fig. 10 The main procedures of the nijimi-rendering algorithm

(5) Source ink quantity variation schema determines how long a diffusion process will persistent. To indicate the points:

- If point P absorbed the amount of liquid ink *Q(p)*, the amount of liquid ink remaining at its source point decreases by *Q(p)*.
- When all the source points have no liquid ink remaining, the diffusion process gets stop.

(6) Gray tone calculation schema determines the intensity for every pixel in the nijimi area.

- Gray tone (or intensity) at each pixel is determined by the amount and the density of liquid ink absorbed there, i.e. $I = Q(p)*V(p)$
- The amount of liquid ink *Q(p)* is evaluated as a

statistical function of the number of fibers passing through the point. See [11] for details.

- The density of liquid ink *D(p)* is evaluated as a function of the time counter value and the density of ink on its corresponding source point.

Figure 10 illustrates the dataflow and the relationship among these procedures. Figure 11 gives two circle strokes with nijimi based on different fiber meshes. Figure 12 shows the results of rendering one stroke in various rendering effects. Figure 13 gives an example of black ink painting generated by using our system..

5. Conclusions

The system was implemented on a Dell PC and coded in Visual C++. A calculation time from 3 to 5 seconds is required to rendering a single stroke with nijimi and kasure effects. This is rapid enough to consider that the system works in real time. Although, it is still not perfect to be used as general-purpose painting software, it delivers enough functionality for the users to creating original black ink paintings. We still have a lot learn from art works drawn by calligraphers. Yet we believe the results presented here are quite encouraging.

As an extension of the present research, we are going to make it possible to generate stroke outlines by a semi-automatic way. Since we have experienced hard work in inputting strokes, we do hope this work can be done more easily. It is also desirable to take advantage of more powerful input devices.

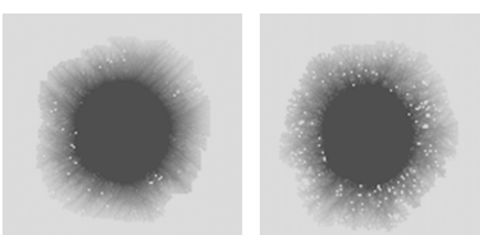


Fig.11 Nijimi based on different fiber meshes.

References

[1] Y. Chua, "Bezier brushstrokes", Computer-Aided Design. Vol.22, No.9, 1990, pp. 550-555.

[2] Fractal Design Corporation, Painter 3.1, 1994.

[3] Q. Guo, T. L. Kunii, "Modeling diffuse paintings of `Nijimi'", Proc.IFIP WG 5.10 Working Conference (Tokyo, Japan), May 1991, pp. 329-338.

[4] Q. Guo, "Generating Realistic Calligraphy Words", IEICE Trans., Vol.E78-A, No.11, November, 1995.

[5] Q. Guo, "Dynamic modeling of "nijimi" diffusion and its application to black ink painting", submitted to the graduate school of the University of Tokyo for the Degree of Doctor of Science in Information Science, Dec. 1995.

[6] R. Greene, "The Drawing Prism: A Versatile Graphic Input Devic'", Computer Graphics, 1985, pp. 103-110.

[7] A. Hertzmann, "Painterly rendering with curved brush strokes of multiple size," Computer Graphics, July 1998, pp.453-460.

[8] S. C. Hsu, I. H. Lee, "Drawing and Animation Using Skeletal Strokes", Computer Graphics, 1994, pp. 109-118.

[9] T. L. Kunii, G. V. Skij, T. Hayashi, "A diffusion model for computer animation of diffuse ink painting", Proceedings of Computer Animation'95, 1995 Los Alamitos, CA, USA, pp.98~102.

[10] T. Nishita, S. Takita, E. Nakamae, "A display algorithm of brush strokes using Bezier functions", CGI93, 1993, pp.244~257.

[11] S. Saito, M. Nakajima, "3D physics-based brush model for painting", Computer Graphics, 1999, pp.226-226.

[12] S. Strassmann, "Hairy brushes", Computer Graphics, Vol. 20, 1986, pp.225-232.

[13] D. Way, Y. Lin, and Z. Shin, " The synthesis of tree in Chinese Landscape painting using silhouette and texture strokes", Journal of WSCG, Vol. 10, 2002.

[14] H. T. F. Wong, H. H S. Ip, "Virtual brush: a model-based synthesis of Chinese calligraphy", Computer and Graphics, 24(1), 2000, pp.99-113.

Fig. 12 'One stroke' in various rendering effects.

Fig.13 A black ink painting created by using our system