Available online at www.sciencedirect.com
SCIENCE DIRECT®
Graphical Models 66 (2004) 263–302
Graphical Models
www.elsevier.com/locate/gmod

Virtual hairy brush for painterly rendering ☆

Songhua Xu,[a,b] Min Tang,[a] Francis C.M. Lau,[b,*] and Yunhe Pan[a]

[a] *CAD & CG State Key Lab of China, Zhejiang University, Hangzhou, PR China*
[b] *Department of Computer Science and Information Systems, The University of Hong Kong, Hong Kong*

Received 24 October 2002; received in revised form 9 April 2004; accepted 6 May 2004
Available online 10 July 2004

Abstract

We propose a novel "e-brush" for calligraphy and painting, which meets all the criteria for a good e-brush. We use only four attributes to capture the essential features of the brush, and a suitably powerful modeling metaphor for its behavior. The e-brush's geometry, dynamic motions, and pigment changes are all dealt with in a single model. A single model simplifies the synchronization between the various system modules, thus giving rise to a more stable system, and lower costs. By a careful tradeoff between the complexity of the model and computation efficiency, more elaborate simulation of the e-brush's deformation and its recovery for interactive painterly rendering is made possible. We also propose a novel paper–ink model to complement the brush's model, and a machine intelligence module to empower the user to easily create beautiful calligraphy and painting. Despite the complexity of the modeling behind the scene, the high-level user interface has a simplistic and friendly design. The final results created by our e-brush can rival the real artwork.

Keywords: Virtual hairy brush; Writing primitive; General sweeping operation; Solid modeling; Real-time simulation; Computer graphics; Design software

☆ Supplementary materials are available at http://www.csis.hku.hk/~songhua/e-brush/.

* Corresponding author. Fax: +2858-4141.
E-mail address: fcmlau@csis.hku.hk (F.C.M. Lau).

1524-0703/$ - see front matter
doi:10.1016/j.gmod.2004.05.006

1. Introduction

The problem of how to simulate Chinese calligraphy and paintings using the computer has attracted many researchers. A good method could produce calligraphic fonts or artwork that are useful in a wide variety of applications, and hence has a good market value. Among the many devices used in art or calligraphic creation, the hairy brush has for centuries been the most popular because of its versatility and special aesthetic and expressive power. It is therefore a meaningful pursuit for computer scientists to find a way to create an "e-hairy brush" that can effectively emulate a real brush. The pursuit is technically challenging because of the very complex structure and features of the hairy brush, especially when the brush is in motion during a calligraphy session. In fact, the brush is not the only object that needs modeling and simulation, but also the paper and the ink.

A good e-brush system should meet the following criteria.

- Easy and natural to use: With a suitable input device, the user should be able to mimic the way he/she uses a real brush to produce an artwork. No user should be required to change his habits or to go through tedious adaptation in order to be able to use the brush. The most ideal virtual brush may even provide feedback to simulate a sense of touch resembling that of the real brush.
- Expressive power and realistic results: The results that can be produced by the system should be a close approximation to the equivalent real artwork, and therefore would appear to be realistic. The power of the brush lies in its ability to simulate a wide variety of effects and styles renderable by the real brush. It is reasonable to imagine that an e-brush can do more than what a real brush can do.
- Flexibility to fit the user and convenience: For the sophisticated user, the system should provide delicate controls for adjusting the different features and parameters of the e-brush. For others, the system can provide a ready-to-use e-brush that needs no further tuning by most users. The system should also provide different brush types for the user to choose, as well as different types of paper and ink.
- Real-time response: A real brush gives real-time responses. If an e-brush is to rival a real one, it must respond instantaneously to the user's manipulation.
- Intelligent computer-aided art creation: To mimic the basic actions and features of a real brush requires certain intelligence on the part of the machine. A more powerful system can rely on artificial intelligence to implement features not present in any real brush, and thus could lead to results that surpass that of a real brush.

An e-brush system as we conceive it consists of three major components: an interactive input component to sample the user's input, a core component to simulate the dynamic behavior of the e-brush, and a component to render the generated result. The proper execution of these components relies on several models that are at the heart of the system, including a geometrical model of the brush, a dynamic model for the simulation, a pigment model for the continuous rendering of the ink mark at the brush tip, and a paper–ink model. Note that the challenge to produce realistic e-artwork lies not only in the modeling of the e-brush, but also the modeling of the paper and the ink [2]. We add a fourth component—a machine-intelligent compo-

nent, with which we offer easier and better manipulatability to the user as well as more optimized results.

1.1. Overview of e-brush and related research

Some previous work has used cubic Bezier curves [3,4], cubic B-spline [5] or skeletal strokes [6] to represent brush strokes, and combined strokes for doing calligraphy or painting. Hobby considered the problem of finding a discrete set of pixels that approximate the envelope of a convex brush shape with respect to a given trajectory [7]. In his approach, a given brush shape is represented by a polygon. In [8], brush stroke boundaries are represented by circles of different diameters along a middle trajectory in the image space. A further step along this direction is reported in [9], where the general sweep boundary of a 2D curved object is used. Also, a brush touch function can be used to construct the shape of strokes [10]. A structural method of using brush strokes to compose a character is discussed in [11]. Given the representation of a character, the technique of rasterization can then be used to generate the image of the character, to be used in applications such as desktop publishing [12]. Systematic creation of large sets of characters ascribing to a certain style leads to typographic fonts. New font creation is indeed an interesting and challenging problem [13]. Pan et al. [14] employed an algebra of geometric shapes to generate new Chinese fonts. Shamir and Rappoport [15] have introduced a parametric method to compactly represent existing outline-based oriental fonts. Ip et al. [16] discussed a method to encode Chinese calligraphic characters using automatic fractal shape coding. Models for generating realistic calligraphy are developed by Guo at [17,18].

Many discussion and research results on the "virtual brush" and its application and values can be found in [19–24]. The paper by Strassmann [25] presents a detailed analysis of the effects a virtual hairy brush can produce. Wong and Ip devised a virtual brush model for synthesizing Chinese calligraphic writings [26], in which the main working units are the cone and some ellipses. There exist many software approaches to modeling the brush [27–32], of which most are physically based solutions. There are also hardware approaches, such as the one by Greene [33].

Besides research on e-brush, some elaborate ink diffusion models have been proposed to simulate different ink spreading effects [34–37]. Artificial intelligence, fuzzy logic, and knowledge based engineering techniques have been found to be useful in equipping a virtual hairy brush to produce beautiful calligraphic artwork [38–42].

With a good e-hairy brush model and a good paper–ink model, beautiful paintings can be generated in the same way as generating beautiful calligraphic artwork [43,44]. Way and Shih [45] used a simple brush model to synthesize beautiful rock textures in Chinese landscape painting. With their method, the contours of the rocks and the areas to which textures are applied are manually supplied by the user using some existing image as reference. Many other papers have proposed similar e-brush approaches to tackle the problem of painterly rendering [46–51].

Simulating the hairy brush's various rendering effects falls into the research area of non-photorealistic rendering (NPR), of which a good survey can be found in the paper by Lansdown and Schofield [52]. Creating Chinese calligraphy or paintings by

an e-hairy brush in real-time bears close resemblance to real-time generation of pen-and-ink illustrations in terms of both the goal and the problem-solving strategies [53,54]. The aesthetic effects pen and ink can produce can be emulated by using a virtual hairy brush with a fine brush tip; but the reverse is not that feasible. The complexity of a virtual hairy brush is obviously much greater than that of a pen or pencil, as the brush is much more powerful in artistic expression. This paper describes a complex virtual hairy brush and its associated paper–ink model, and shows, despite the complexity, how the simulated brush can operate efficiently in real-time.

1.2. Our work and contributions

We developed an algorithmic software framework that can simulate the change of the physical conditions of a hairy brush, including the brush's geometric shape and its ink-related properties during the Chinese calligraphic writing process, where ink-related properties refer to the brush's degree of wetness and color. We have constructed an interactive software system implementing these algorithms which can be used to create calligraphic artwork fully electronically. Fig. 1 shows the overall architecture of our system.

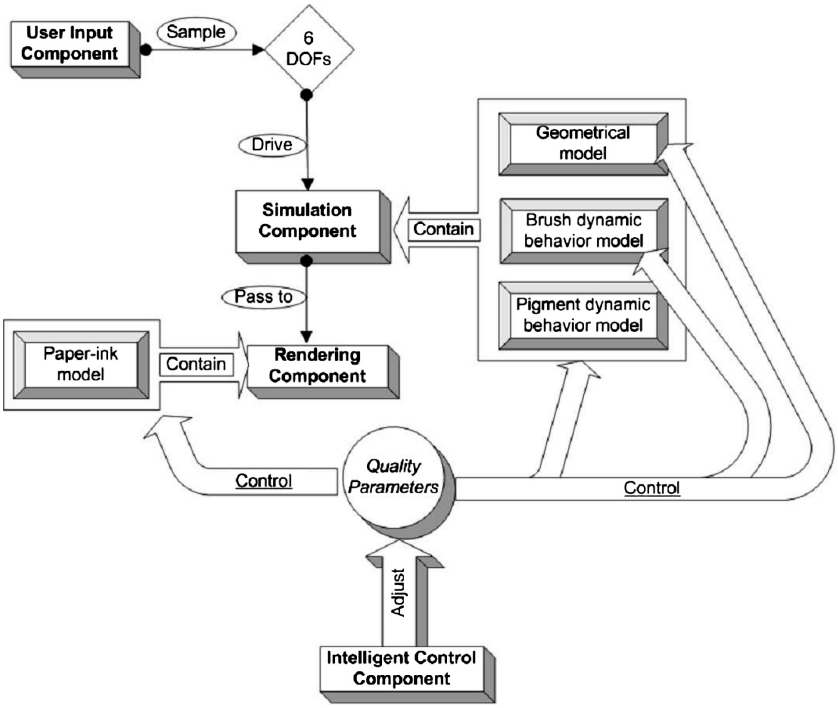

Fig. 1. Architecture of our virtual brush system.

Our virtual hairy brush is closer to the real brush than other similar brushes because our e-brush can automatically determine both the geometric contour and the texture of its current drawing mark on the virtual paper at the same time. This process is done in real-time and no human intervention is necessary, which is not the case in other similar systems. The drawing mark in our model can be varied and of irregular shape which is generated from any planar parametric curve instead of just an ellipse. This is a vital feature for achieving quality in an artist's work.

We introduce a useful concept called *writing primitive*, which is a hair cluster, to serve as the basic working unit of the virtual hairy brush. The cross section of each writing primitive as one indivisible entity intersecting the virtual paper plane and its ink-related information are computed only once during every time step. This makes real-time simulation of the hairy brush's writing and painting behavior possible in our system. We have implemented a prototype system to demonstrate the effectiveness of our algorithms in constructing a high-quality e-brush system.

By embedding ink-related information in the writing primitive's control axis, a single primitive can readily express reasonably complex, interesting, and even mysterious distribution of the ink, including its color and wetness. In our proposed ink model, we use probabilities to create realistic effects to be used in rendering the current ink-mark, thus enabling our virtual hairy brush to simulate the drying and running effects of calligraphic artwork. Multiple gray levels, full-color paintings, dry brush writing effects, and saturation effects can all be produced using this new ink model. All these writing effects contribute to the system's expressive power needed by computer-aided art creation.

We also introduce an inertia predictor to calculate the brush's virtual position based on its sampled position during the writing process. This inertia predictor simulates the acceleration of the virtual hairy brush, which gives the user the feeling of a real physical brush, and helps to produce output that rivals real artwork. In addition to various ways that allow the user to manually edit various quality parameters of the virtual hairy brush, special optimization algorithms are built into the system to automatically customize these parameters to achieve better maneuverability of the brush and improved quality in the output.

According to the six degrees of freedom of the hairy brush, which are sampled periodically, the computer can simulate the whole Chinese calligraphy process with high accuracy. The process can be performed with real-time response, as proven through experimentation. Since many of the geometrical and dynamic parameters of the brush can be automatically determined by the system, it is not necessary to store any bitmap image (for the brush's cross section) during the writing process. After completing the writing process, it is also not necessary to store the generated artwork in any standard format; a small file containing the changes of the virtual hairy brush's six degrees of freedom during the whole writing process is sufficient to reconstruct the full final image. Hence, the storage requirement of our approach is minimal.

Section 2 introduces and explains the concept of writing primitive. Section 3 presents the solid model of the virtual hairy brush. Section 4 discusses the sampling and processing of the brush's input. Section 5 discusses how the parameters in the

parametric models of the e-brush are adjusted dynamically. Section 6 presents the rendering of the brush's current ink mark at any time instant. Section 7 discusses how the brush and its quality parameters are configured by the system automatically. Section 8 gives an overview of the implemented system and showcases some examples of artwork created using the system. Section 9 presents the related work. Section 10 discusses some possible future extensions. Section 11 summarizes and concludes the paper.

2. Writing primitives

We rely on the concept of *writing primitive*, which represents a hair cluster (i.e., a small bundle of hair), to reduce the complexity of the modeling and thus the computational requirement for the system. A virtual hairy brush consists of one or more writing primitives. Each writing primitive is described by a NURBS surface, and is constructed through the general sweeping operation in CAD. The behavior of the virtual hairy brush is an aggregation of the behavior of all its writing primitives. This is in sharp contrast with the approach used by Wong and Ip [26], where every hair is operated on; it is also different from the DAB system [47], where the whole brush head is modeled as one subdivision surface. The use of writing primitives does not diminish in any way the power of the virtual hairy brush in satisfactorily simulating all possible behavior of a real hairy brush including the branching out behavior. This is because the clustering of hair is a natural phenomenon and writing primitive appears to be a good model for capturing this phenomenon. In the physical clustering of hair, hair in the same bundle share similar ink-related properties, which is also captured in our modeling. Our experimental results have confirmed the correctness of this approach in modeling the real brush, as well as the outstanding expressive power of our model. The artwork created using our system can be seen as better than those in the paper by Wong and Ip.

A writing primitive in the model is defined by its four attributes as shown in Fig. 2. Based on these four attributes, the model is constructed through the general sweeping operation in CAD. This operation will construct a NURBS sweeping surface by taking the curve of the writing primitive's middle control axis as its sweeping trajectory and the cross sections defined by the user at initialization as the given profiles. Note that at the start of the writing process, the brush consists of a single writing primitive. The general sweeping operation is implemented according to the minimized rotation frame algorithm by Maurer and Juttler [55]. During the simulation, three of the four attributes (not including the bottom control circle) of the writing primitive will be dynamically adjusted according to the input data which describe the brush's current position in the 3D space. The bottom control circle never changes during the writing process.

All the input data are preprocessed to take into account the inertia of the hairy brush, which creates a realistic "feel" of the brush. We refer to the cross section of the intersection between a writing primitive and the virtual paper plane as the writing primitive's current drawing mark. For every time slice of the writing process,

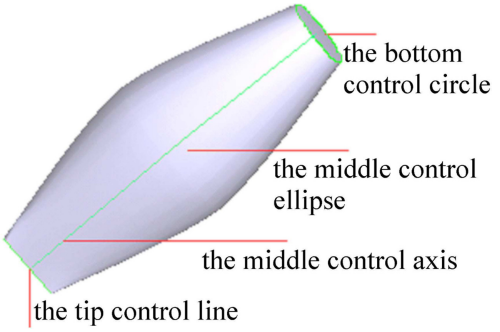

Fig. 2. A writing primitive and its four attributes.

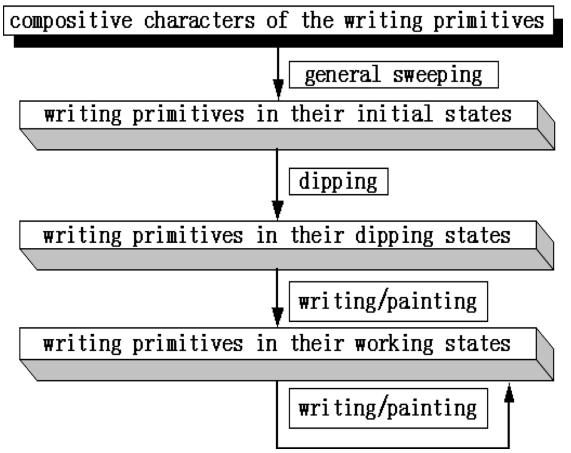

Fig. 3. State transitions of a writing primitive.

ink will be deposited according to the state of the current drawing mark. The union of the current drawing marks of all the writing primitives is the current mark made by the brush on the paper. The final artwork is the accumulation of such marks over all the time slices. Fig. 3 shows the state transition of a writing primitive. The virtual hairy brush's working diagram is shown in Fig. 4. It will be explained in detail in Sections 4–6.

3. The model and the states

3.1. The parametric model of the virtual hairy brush

Our model of a virtual hairy brush (**HB**) is in terms of the collection of writing primitives that the brush is composed of. Formulation 1 defines **HB**.

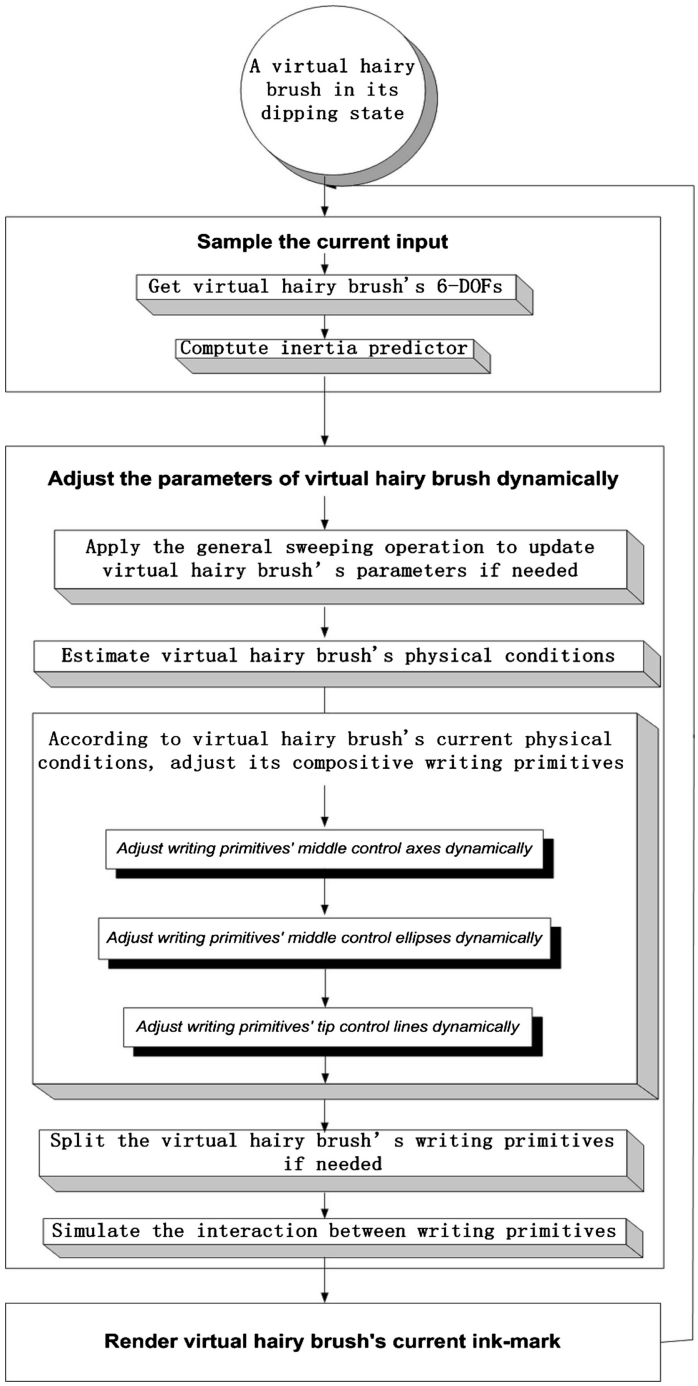

Fig. 4. Virtual hairy brush's working diagram.

Virtual Hairy Brush System's Quality Parameters			
Name	Concise Meaning	Category	Usage
e	**HB**'s degree of elasticity	$\mathbf{Qp}^1$	Eq. 12, Eq. 14
tre	**HB**'s threshold for splitting	$\mathbf{Qp}^1$	Eq. 20
ν	**HB**'s relocation factor	$\mathbf{Qp}^1$	Eq. 19
re	**HB**'s rotation coefficient	$\mathbf{Qp}^1$	Eq. 18
ie	**HB**'s elongation coefficient	$\mathbf{Qp}^1$	Eq. 18
sm	ψ's degree of smoothness	$\mathbf{Qp}^1$	Eq. 14
ren	**HB**'s color rendering control factor	$\mathbf{Qp}^2$	Eq. 29
η	**HB**'s ink diffusion distance factor	$\mathbf{Qp}^2$	Eq. 22
ab	ψ's absorbing ability	$\mathbf{Qp}^2$	Eq. 17, Eq. 26, Eq. 29
pw	ψ's diffusion control factor	$\mathbf{Qp}^2$	Eq. 22
dry	ψ's drying factor	$\mathbf{Qp}^2$	Eq. 26
Notes	**HB** refers to the virtual hairy brush; ψ refers to the virtual paper.		
	$\mathbf{Qp}^1$ is **HB**'s quality parameters for shape, and $\mathbf{Qp}^2$ quality parameters for texture.		

Fig. 5. Virtual hairy brush system's quality parameters.

$$\begin{cases} \mathbf{HB} \triangleq (\mathbf{CWP}, \mathbf{Qp}), \\ \mathbf{CWP} \triangleq (\mathbf{WP}_1, \mathbf{WP}_2, \ldots, \mathbf{WP}_m), \\ \mathbf{Qp} \triangleq (\mathbf{Qp}^1, \mathbf{Qp}^2), \\ \mathbf{Qp}^1 \triangleq (e, tre, v, re, ie, sm), \\ \mathbf{Qp}^2 \triangleq (ren, \eta, ab, pw, dry). \end{cases} \tag{1}$$

The brush's writing primitives are denoted as $\mathbf{WP}_1, \mathbf{WP}_2, \ldots, \mathbf{WP}_m$. Each writing primitive $\mathbf{WP}_i (i = 1, 2, \ldots, m)$, described by a NURBS surface, is constructed by the general sweeping operation in CAD. The number of **HB**'s compositive writing primitives, m, will be dynamically adjusted by the system. There are several parameters for controlling the artwork style generated by the brush, the values of which are within [0,1]. These parameters are called the quality parameters **Qp** of **HB**, which are classified into two categories. The first category, $\mathbf{Qp}^1$, affects the created brush strokes' boundary. The second category, $\mathbf{Qp}^2$, affects the texture of the created brush strokes. Fig. 5 contains a complete list of these parameters. The meaning and detailed usage of these parameters and how they are configured will be explained later in the paper.

3.2. The parametric model of a writing primitive

The parametric model of a writing primitive $\mathbf{WP}_i (i = 1, 2, \ldots, m)$ is defined as: $\mathbf{WP}_i \triangleq (\mathbf{C}_i, \mathbf{E}_i, \mathbf{L}_i, \mathbf{A}_i)$. To generate the parametric model of a writing primitive $\mathbf{WP}_i (i = 1, 2, \ldots, m)$ through the general sweeping operation, a circle $\mathbf{C}_i$, an ellipse

$\mathbf{E}_i$, and a line $\mathbf{L}_i$ are taken as the sweeping profiles, and an axis $\mathbf{A}_i$ is used as the sweeping trajectory. $\mathbf{C}_i$, $\mathbf{E}_i$, $\mathbf{L}_i$, $\mathbf{A}_i$ are called $\mathbf{WP}_i$'s bottom control circle, middle control ellipse, tip control line, and middle control axis, respectively (Fig. 2). Among these four attributes, $\mathbf{C}_i$ is a static attribute of $\mathbf{WP}_i$, that is, once $\mathbf{C}_i$ is initialized at the beginning, it remains unchanged during the whole process.

3.2.1. The bottom control circle of a writing primitive

The bottom control circle $\mathbf{C}_i$ of writing primitive $\mathbf{WP}_i$ is defined as:

$$\begin{cases} \mathbf{C}_i \triangleq (\mathbf{cen}_i, r_i, \mathbf{cori}_i), \\ \mathbf{cen}_i \triangleq (ccx_i, ccy_i, ccz_i), \\ \mathbf{cori}_i \triangleq (cox_i, coy_i, coz_i), \end{cases} \tag{2}$$

where $\mathbf{cen}_i$ is the coordinates of $\mathbf{C}_i$'s center, r_i its radius, and $\mathbf{cori}_i$, a 3D unit vector representing $\mathbf{C}_i$'s orientation. All the writing primitives of a brush share the same bottom control circle, and so this circle is also called the bottom control circle of the brush, denoted as $\mathbf{HB}.\mathbf{C}$.

3.2.2. The tip control line of a writing primitive

The tip control line $\mathbf{L}_i$ of $\mathbf{WP}_i$ is defined as:

$$\begin{cases} \mathbf{L}_i \triangleq (len_i, \mathbf{mid}_i, \mathbf{lori}_i), \\ \mathbf{mid}_i \triangleq (mix_i, miy_i, miz_i), \\ \mathbf{lori}_i \triangleq (lox_i, loy_i, loz_i), \end{cases} \tag{3}$$

where len_i is the current length of $\mathbf{L}_i$, $\mathbf{mid}_i$ is $\mathbf{L}_i$'s midpoint, and $\mathbf{lori}_i$, a 3D unit vector representing $\mathbf{L}_i$'s orientation.

3.2.3. The middle control axis of a writing primitive

The middle control axis $\mathbf{A}_i$ is a cubic B-spline curve with interpolated key points $\mathbf{P}_{i,1}, \mathbf{P}_{i,2}, \ldots, \mathbf{P}_{i,n_i}$. Each of these points carries both geometric and ink-related information, including $\mathbf{P}_{i,j}$'s color in RGB and its degree of wetness. $\mathbf{P}_{i,j}$ is defined in Formulation 4:

$$\begin{cases} \mathbf{P}_{i,j} \triangleq (\mathbf{cc}_{i,j}, \mathbf{col}_{i,j}, wet_{i,j}, \mathbf{wv}_{i,j}, \mathbf{cv}_{i,j}, wr_{i,j}, cr_{i,j}), \\ \mathbf{cc}_{i,j} \triangleq (cx_{i,j}, cy_{i,j}, cz_{i,j}), \\ \mathbf{wv}_{i,j} \triangleq (wx_{i,j}, wy_{i,j}, wz_{i,j}), \\ \mathbf{cv}_{i,j} \triangleq (cx_{i,j}, cy_{i,j}, cz_{i,j}), \\ i = 1, 2, \ldots, m; j = 1, 2, \ldots, n_i. \end{cases} \tag{4}$$

Here, $\mathbf{cc}_{i,j}$ is $\mathbf{P}_{i,j}$'s coordinates. $\mathbf{col}_{i,j}$ is $\mathbf{P}_{i,j}$'s color in RGB format. With this formulation, we can generate colorful calligraphic artwork and even create watercolor paintings. $wet_{i,j}$ is $\mathbf{P}_{i,j}$'s degree of wetness. For each $\mathbf{A}_i$, its first key point is always set at the center of the bottom control circle, and its last key point at the midpoint of the tip control line; that is,

$$\begin{cases} \mathbf{cc}_{i,1} \triangleq \mathbf{cen}_i, \\ \mathbf{cc}_{i,n_i} \triangleq \mathbf{mid}_i. \end{cases}$$

Each key point has four fields for specifying the point's ink-related information: *vector mode* wetness changing vector $\mathbf{wv}_{i,j}$, *vector mode* color changing vector $\mathbf{cv}_{i,j}$, *radiation mode* wetness changing factor $wr_{i,j}$, and *radiation mode* color changing factor $cr_{i,j}$. Based on these fields, we can compute the ink-related information for any point $\mathbf{Q}$ on a plane perpendicular to the middle control axis $\mathbf{A}_i$ passing through $\mathbf{P}_{i,j}$. We allow two modes of ink distribution in the virtual hairy brush: the *vector mode*, in which ink is distributed according to the direction of a certain vector; and the *radiation mode*, in which ink is distributed radically. There is an ink-related texture function associated with each key point, which produces some hybrid effect contributed by these two distribution modes. Fig. 6 explains these two modes of ink distribution. The contribution made by $\mathbf{P}_{i,j}$'s vector mode ink distribution is:

$$\begin{cases} \boldsymbol{\delta}_v wet_{\mathbf{Q}} = wet_{\mathbf{P}_{i,j}} \times ((\mathbf{cc}_{\mathbf{Q}} - \mathbf{cc}_{\mathbf{P}_{i,j}}) \cdot \mathbf{wv}_{i,j}), \\ \boldsymbol{\delta}_v \mathbf{col}_{\mathbf{Q}} = \mathbf{col}_{\mathbf{P}_{i,j}} \times ((\mathbf{cc}_{\mathbf{Q}} - \mathbf{cc}_{\mathbf{P}_{i,j}}) \cdot \mathbf{cv}_{i,j}), \end{cases}$$

where $\mathbf{cc}_{\mathbf{Q}}$, $wet_{\mathbf{Q}}$, and $\mathbf{col}_{\mathbf{Q}}$ are point $\mathbf{Q}$'s current coordinates in the 3D space, its degree of wetness, and its color, respectively; and $\mathbf{cc}_{\mathbf{P}_{i,j}}$, $wet_{\mathbf{P}_{i,j}}$, and $\mathbf{col}_{\mathbf{P}_{i,j}}$ are the corresponding values of point $\mathbf{P}_{i,j}$. Similarly for the contribution made by $\mathbf{P}_{i,j}$'s radiation mode ink distribution:

$$\begin{cases} \boldsymbol{\delta}_r wet_{\mathbf{Q}} = wet_{\mathbf{P}_{i,j}} \times \|\mathbf{cc}_{\mathbf{Q}} - \mathbf{cc}_{\mathbf{P}_{i,j}}\| \times wr_{i,j}, \\ \boldsymbol{\delta}_r \mathbf{col}_{\mathbf{Q}} = \mathbf{col}_{\mathbf{P}_{i,j}} \times \|\mathbf{cc}_{\mathbf{Q}} - \mathbf{cc}_{\mathbf{P}_{i,j}}\| \times cr_{i,j}. \end{cases}$$

Thus, $\mathbf{P}_{i,j}$'s ink-related texture function, which computes the hybrid effect contributed by both the point's vector mode and radiation mode ink distribution patterns is defined as:

$$\begin{cases} wet_{\mathbf{Q}} \triangleq wet_{\mathbf{P}_{i,j}} + \boldsymbol{\delta}_r wet_{\mathbf{Q}} + \boldsymbol{\delta}_v wet_{\mathbf{Q}}, \\ \mathbf{col}_{\mathbf{Q}} \triangleq \mathbf{col}_{\mathbf{P}_{i,j}} + \boldsymbol{\delta}_r \mathbf{col}_{\mathbf{Q}} + \boldsymbol{\delta}_v \mathbf{col}_{\mathbf{Q}}, \end{cases}$$

or

$$\begin{cases} wet_{\mathbf{Q}} = wet_{\mathbf{P}_{i,j}} \times (1 + \|\mathbf{cc}_{\mathbf{Q}} - \mathbf{cc}_{\mathbf{P}_{i,j}}\| \times wr_{i,j} + (\mathbf{cc}_{\mathbf{Q}} - \mathbf{cc}_{\mathbf{P}_{i,j}}) \cdot \mathbf{wv}_{i,j}), \\ \mathbf{col}_{\mathbf{Q}} = \mathbf{col}_{\mathbf{P}_{i,j}} \times (1 + \|\mathbf{cc}_{\mathbf{Q}} - \mathbf{cc}_{\mathbf{P}_{i,j}}\| \times cr_{i,j} + (\mathbf{cc}_{\mathbf{Q}} - \mathbf{cc}_{\mathbf{P}_{i,j}}) \cdot \mathbf{cv}_{i,j}). \end{cases} \quad (5)$$

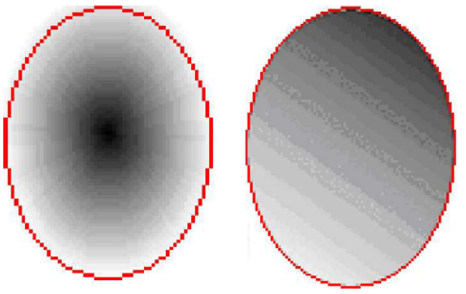

Fig. 6. Radiation (left) and vector (right) ink distribution modes.

Note that by vector mode ink distribution, we mean that ink distribution can be characterized by a certain vector. Vector mode contribution is not necessarily positive. If the vector $\overrightarrow{\mathbf{cc}_{\mathbf{P}_{i,j}}\mathbf{cc}_{\mathbf{Q}}}$ is along the direction of the vector, the contribution will be positive; otherwise, it could be negative. Please refer to Fig. 6 (right) for an illustration. Note also that only those $\mathbf{P}_{i,j}$s touching or just below or above the paper need to be evaluated in the virtual painting and writing process.

3.2.4. The middle control ellipse of a writing primitive

The middle control ellipse $\mathbf{E}_i$ of the writing primitive $\mathbf{WP}_i$ is defined as:

$$\begin{cases} \mathbf{E}_i \triangleq (a_i, b_i, loc_i, \mathbf{eori}_i), \\ \mathbf{eori}_i \triangleq (eox_i, eoy_i, oez_i), \end{cases} \tag{6}$$

where a_i is the length of $\mathbf{E}_i$'s major axis and b_i the length of $\mathbf{E}_i$'s minor axis. loc_i is $\mathbf{E}_i$'s location parameter, which indicates $\mathbf{E}_i$'s relative position along the middle control axis to which it belongs. $\mathbf{eori}_i$ represents $\mathbf{E}_i$'s minor axis's orientation.

3.3. The three states of a brush

A virtual hairy brush **HB** is assumed to have three possible states in its life cycle: the initial state, the dipping state, and the working state (Fig. 3).

3.4. The initial state of a virtual hairy brush

The initial state of the virtual hairy brush **HB** is when all of its compositive writing primitives are in their free states (Fig. 7).

The three dynamic attributes of a writing primitive are simplest when in this state: the tip control line, the middle control axis and the middle control ellipse are reduced to a point, a straight line, and a circle, respectively. Varying the radius of the circle and its position along the middle control axis can result in a series of modeling effects, as shown in Fig. 8.

3.5. The dipping state of a virtual hairy brush

All the writing primitives of the virtual hairy brush shift to the dipping state after the brush is dipped into the ink bottle and before touching the paper. Through dipping, writing primitives acquire ink-related information, which includes color and degree of wetness and is according to how the brush is dipped.

If $\mathbf{P}_{i,k}, \mathbf{P}_{i,k+1}, \ldots, \mathbf{P}_{i,n_i}$ are $\mathbf{WP}_i$'s key points that are soaked in ink, their color is simply set to the ink color. For the other key points, $\mathbf{P}_{i,l}, l = 2, 3, \ldots, k-1$, which are not soaked in ink, linear interpolations are applied to compute their individual

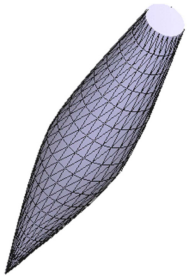

Fig. 7. A writing primitive in its initial state.

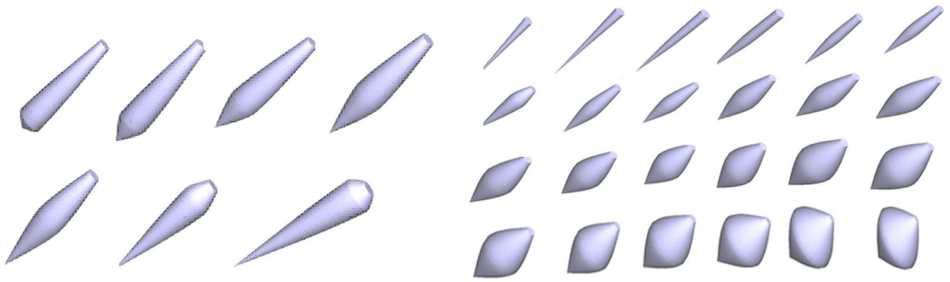

Fig. 8. Writing primitives with the middle control ellipse at different locations (left); writing primitives with different circle radii (right).

colors $\mathbf{col}_{\mathbf{P}_{i,l}}$ with the assumption that $\mathbf{col}_{\mathbf{P}_{i,1}} \equiv 0$, which is the color code for pure white. Here, we assume the paper is white. If that is not the case, we would substitute the color of the paper for pure white. The color distribution after dipping is depicted as:

$$\begin{cases} \mathbf{col}_{i,l} = \text{ink color}, \\ (l = k, k+1, \ldots, n_i), \\ \mathbf{col}_{i,l} = \frac{\|\mathbf{cc}_{i,k} - \mathbf{cc}_{i,l}\| \times \mathbf{col}_{i,1} + \|\mathbf{cc}_{i,l} - \mathbf{cc}_{i,1}\| \times \mathbf{col}_{i,k}}{\|\mathbf{cc}_{i,k} - \mathbf{cc}_{i,1}\|}, \\ (l = 2, 3, \ldots, k-1), \\ \mathbf{col}_{i,1} = 0. \end{cases} \tag{7}$$

Similarly for each key point's degree of wetness. We assume $wet_{\mathbf{P}_{i,1}} \equiv 0$, meaning that $\mathbf{P}_{i,1}$ is all dry; thus:

$$\begin{cases} wet_{i,l} = \text{the degree of wetness}, \\ (l = k, k+1, \ldots, n_i), \\ wet_{i,l} = \frac{\|\mathbf{cc}_{i,k} - \mathbf{cc}_{i,l}\| \times wet_{i,1} + \|\mathbf{cc}_{i,l} - \mathbf{cc}_{i,1}\| \times wet_{i,k}}{\|\mathbf{cc}_{i,k} - \mathbf{cc}_{i,1}\|}, \\ (l = 2, 3, \ldots, k-1), \\ wet_{i,1} = 0. \end{cases} \tag{8}$$

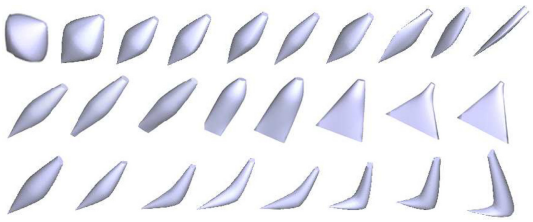

Fig. 9. Writing primitives with different ratios of major-to-minor axis length of the middle control ellipse (top row); writing primitives with different tip control line lengths (middle); writing primitives with different middle control axes (bottom).

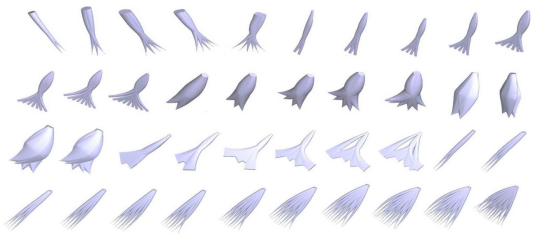

Fig. 10. Virtual brushes with hair split into several writing primitives.

3.6. The working state of a virtual hairy brush

The working state of a virtual hairy brush is the "deformation" state of the brush. The brush deforms due to touching or being pressed against the paper. This varies the eccentricity of the middle control ellipse, the tip control line and the middle control axis, leading to a series of modeling effects, as shown in Fig. 9. The pressure against the paper may build up to the point where the brush hair will split, as shown in Fig. 10.

4. Sampling of the input data

During the writing process, the brush's dynamic attributes are captured by sampling. Sampled input data are used to adjust the virtual brush dynamically. All such adjustments will preserve the validity (with respect to a real brush) of the parametric model. We need to first obtain the brush's six degrees of freedom. Assume that the sampled datum at time t is $\mathbf{Sam}^t \triangleq (x^t, y^t, z^t, d^t, q^t, r^t)$, where (x^t, y^t, z^t) is the center of **HB**'s bottom control circle **HB**.**C** in the 3D space; d^t the degree of **HB**'s sideways deflection; q^t the degree of **HB**'s forward deflection; and r^t the degree of **HB**'s rotation (Fig. 11).

There are many ways to input a solid object's six degrees of freedom, such as using some special device [47]. A 3D mouse, or a data glove or any other sensor that can

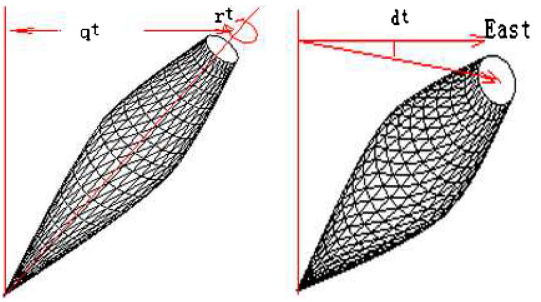

Fig. 11. The six degrees of freedom of an input sample.

take in the six degrees of freedom for a rigid body can also be used as the input hardware for the virtual hairy brush. Some other interesting methods have been introduced [56,57].

To obtain the six degrees of freedom as input to our virtual brush system, we offer two methods. One method is keyboard plus mouse, and the other method is to use a tablet to obtain five out of six degrees and key pressing for the sixth degree, namely the rotation degree of the brush. The sixth degree is in fact the degree with the least changes, unless the artist is drawing very cursive brush strokes. The keyboard-mouse method is designed to allow for wide adaptability by ordinary home PC users. For the sampling of the six degrees of freedom using this method, please refer to Fig. 12 which shows the user actions and their corresponding effects.

Note that during the writing process not all of the virtual hairy brush's six degrees of freedom experience the same degree of changing. Under most circumstances,

User Action	Corresponding Effect
Moving the mouse	Generate the (x^t, y^t) coordinates
Scrolling the middle wheel of the mouse	Generates the z^t coordinate
Pressing key "q" by the little finger	Reduces the value of d^t
Pressing key "w" by the ring finger	Increases the value of d^t
Pressing key "e" by the middle finger	Reduces the value of q^t
Pressing key "r" by the index finger	Increases the value of q^t
Pressing key "c" by the thumb	Reduces the value of r^t
Pressing key "v" by the thumb	Increases the value of r^t
Pressing a key while the left button is pressed	The key is pressed twice
Pressing a key while the right button is pressed	The key is pressed four times

Fig. 12. The 3-button-mouse and keyboard hybrid input method.

(x^t, y^t, z^t) will change much more frequently and sharply than d^t and q^t; r^t rarely changes. Our 3-button-mouse and keyboard hybrid input strategy offers an easy but effective way for the user to input (x^t, y^t, z^t); using the mouse to input d^t, q^t, r^t is a little more awkward but acceptable, as can be demonstrated by our experimental results. Of course, more expensive input devices such as the data glove or those with high dimensional sensors can certainly enhance the usability of the system.

During the writing process, **HB**'s virtual position at time t, $\mathbf{S}^t \triangleq (sx^t, sy^t, sz^t, sd^t, sq^t, sr^t)$, is computed from the current sampled datum $\mathbf{Sam}^t$ together with an inertia predictor $\mathbf{M}^t = (mx^t, my^t, mz^t, md^t, mq^t, mr^t)$, using the following formula:

$$\mathbf{S}^t = wei_{\text{sam}} \times \mathbf{Sam}^t + (1 - wei_{\text{sam}}) \times \mathbf{M}^t. \tag{9}$$

$\mathbf{M}^t$ comes from **HB**'s displacements in its last few sampling intervals $\mathbf{S}^{t-1}, \mathbf{S}^{t-2}, \ldots, \mathbf{S}^{t-n}$:

$$\mathbf{M}^t = \mathbf{Vel}^t \times \mathrm{d}T + \mathbf{S}^{t-1}, \tag{10}$$

$$\mathbf{Vel}^t = \frac{\sum_{k=1}^{el} \left(wei_k \times \frac{\mathbf{S}^{t-k} - \mathbf{S}^{t-k-1}}{\mathrm{d}T} \right)}{\sum_{k=1}^{el} wei_k}. \tag{11}$$

Here $\mathbf{Vel}^t$ is a weighted sum representing an estimate of **HB**'s velocity, $\mathrm{d}T$ the length of a sampling interval, and el the length of time over which our estimate is computed. wei_{sam} and the wei_ks are the relative weights given to the current sampled datum and the recent past velocities of the brush, respectively. A heavy wei_{sam} for instance means that the brush has a small inertia. This simple method to determine the position of **HB** turns out to be reasonable since the speed of the brush in real life rarely changes too abruptly, as can be easily observed when artists create Chinese calligraphic artwork using real physical brushes. By introducing an inertia predictor to influence the sampled position of the brush to yield its virtual position, the user can move the virtual hairy brush more continually to emulate the effect and enjoy the feeling of brush gliding; and although it is a simple mouse that is used to create electronic calligraphic artwork, the system could still give the user the approximate feeling of a real physical brush in action. For the above formulation, the default parameter configuration is: $el = 4$; $wei_1 = 4$, $wei_2 = 2$, $wei_3 = 1$, $wei_4 = 1$; and $wei_{\text{sam}} = 0.6$. These default values were obtained by experiments. Changing these values can yield different writing styles and feelings.

5. Dynamic adjustments of the brush

5.1. Estimating the physical conditions of the brush

Based on the sampled input data, the physical conditions of the virtual hairy brush are estimated at every time step; these conditions include the writing primitive $\mathbf{WP}_i$'s inner stress str_i^t at time t and the pressure of the primitive due to its interaction with the paper. The greater the inner stress, the more likelihood there is for the brush

hair to split, and the pressure is the force per unit area as experienced by the virtual paper due to the inner stress of the brush. The parametric model of the e-brush is updated dynamically based on these estimates.

5.1.1. Estimating a writing primitive's current inner stress

Intuitively, the rigidity of $\mathbf{WP}_i$'s hair, $\mathbf{WP}_i$'s historical deformation, the wetness of $\mathbf{WP}_i$, and the size of the part of $\mathbf{WP}_i$ that is against the virtual paper plane $\boldsymbol{\psi}$ all have an effect on the value of str_i^t. We devised accordingly a formula to estimate str_i^t based on these factors:

$$str_i^t \triangleq (1-e) \times \frac{n_i}{\sum_{k=1}^{n_i} wet_{i,k}^t} \times \frac{S_i^t \times hei_i^t}{3} \times his_i^t \times \frac{\pi \times a_i^t \times b_i^t}{4}. \tag{12}$$

The term e which has a value between 0 and 1 represents the elasticity of **HB**'s hair, and hence $(1-e)$ indicates the rigidity of the hair. Since a_i^t and b_i^t are the lengths of $\mathbf{WP}_i$'s middle control ellipse $\mathbf{E}_i$'s major axis and minor axis at time t, respectively, the term $(\pi \times a_i^t \times b_i^t)/4$ is the area of $\mathbf{E}_i$. It is used to approximate $\mathbf{WP}_i$'s number of hair threads. The term $(\sum_{k=1}^{n_i} wet_{i,k}^t)/n_i$ is used to approximate $\mathbf{WP}_i$'s overall average degree of wetness, where n_i is the number of key points along the middle control axis $\mathbf{A}_i$ of the ith writing primitive $\mathbf{WP}_i$. The reciprocal is used in the formulation since a wet brush will experience less force than a dry brush. S_i^t is the area of $\mathbf{WP}_i$'s cross section against the virtual paper $\boldsymbol{\psi}$ at time t approximated by the area of the section's smallest bounding box. hei_i^t is the distance between the cross section and the middle point $\mathbf{mid}_i$ of $\mathbf{WP}_i$'s tip control line. We use a cone to approximate the part of $\mathbf{WP}_i$ that is under $\boldsymbol{\psi}$, which has a volume of $(S_i^t \times hei_i^t)/3$. The larger this volume is, the more it contributes to the inner stress.

The deformation of the brush also has a bearing on the inner stress, which can be viewed as the accumulated result of a series of per-time-step deformation since the starting of the writing process. The factor his_i^t represents this deformation factor in the formulation. It is decomposed into two parts: the displacement from the initial state, hdd_i^t, and shape deformation, hsd_i^t. And so his_i^t is defined as:

$$\begin{cases} his_i^t \triangleq hsd_i^t + hdd_i^t, \\ hsd_i^t \triangleq \frac{a_i^t}{b_i^t} \times len_i^t, \\ hdd_i^t \triangleq \frac{\sum_{j=2}^{n_i} (\|\mathbf{cc}_{i,j}^0 - \mathbf{cc}_{i,j}^t - \mathbf{mc}^t\| \times \|\mathbf{cc}_{i,j}^t - \mathbf{cc}_{i,j-1}^t\|)}{\sum_{j=2}^{n_i} \|\mathbf{cc}_{i,j}^t - \mathbf{cc}_{i,j-1}^t\|}, \\ \mathbf{mc}^t \triangleq \mathbf{cc}_{i,1}^0 - \mathbf{cc}_{i,1}^t. \end{cases} \tag{13}$$

We use the ratio of middle control ellipse $\mathbf{E}_i$'s major axis' length to its minor axis' length, $(a_i^t)/(b_i^t)$, and $\mathbf{L}_i$'s length, len_i^t, to evaluate $\mathbf{E}_i$'s shape deformation due to past writing. hdd_i^t is a weighted sum of the displacement, $\|\mathbf{cc}_{i,j}^0 - \mathbf{cc}_{i,j}^t - \mathbf{mc}^t\|$, of each key point $\mathbf{P}_{i,j}$, where $\mathbf{cc}_{i,j}^0$ is $\mathbf{P}_{i,j}$'s coordinates when $\mathbf{WP}_i$ is at its initial free state, $\mathbf{cc}_{i,j}^t$ is $\mathbf{P}_{i,j}$'s coordinates at time t during writing, and $\mathbf{mc}^t$ is the accumulated displacement of $\mathbf{WP}_i$'s bottom control circle $\mathbf{C}_i$. Recall that $\mathbf{WP}_i$'s first key point

($j = 1$) is set as $\mathbf{C}_i$'s center, $\mathbf{cen}_i$. Here the weights are the difference of the relative positions of every pair of adjacent key points along the control axis $\mathbf{A}_i$. In our experiments, we found this formula to be a good approximation of the behavior of a real brush. In comparison with the classical formulation in solid mechanics [58], this approximation formulation is simpler, which is helpful in reducing the computation time.

In our current design, we use the term "elasticity" (e) to refer to an inherent physical property of the paintbrush, that is independent of the brush's wetness. During brush dynamics simulation, we will then take into account the factor contributed by the brush's wetness. It is more or less a personal preference whether or not to include the wetness factor early in the elasticity equation or later in the dynamics simulation equation. Based on our study of material science, we adopt the current strategy because it seems more reasonable to model the elasticity as an essential material property of the brush hair.

5.1.2. Estimating the pressure due to interaction with the virtual paper

Given that we have a value for the inner stress, we can devise a formula to estimate $\mathbf{WP}_i$'s pressure γ_i^t due to interaction between a writing primitive and the virtual paper $\boldsymbol{\psi}$. This pressure contributes directly to the degree of deformation of the brush during writing. It can be easily seen that fast movements of the virtual hairy brush, a high degree of inner stress of the current writing primitive $\mathbf{WP}_i$, and a coarse virtual paper can all severely deform **HB** (i.e., the middle control axes of the brush's compositive writing primitives). The formulation is as follows.

$$\gamma_i^t \triangleq \|\mathbf{Vel}^t\| \times str_i^t \times sm \times e, \tag{14}$$

where $\mathbf{Vel}^t$ is the virtual hairy brush's velocity, str_i^t its inner stress, sm the virtual paper's degree of smoothness, and e the e-brush's degree of elasticity.

5.2. Dynamic adjustment of the middle control axis

5.2.1. The current active point

During the writing process, the intersection of the middle control axis $\mathbf{A}_i$ and the virtual paper $\boldsymbol{\psi}$ is the current active point. The current active point is inserted into $\mathbf{A}_i$'s series of key points dynamically. The point's ink-related information is computed from the neighboring points in $\mathbf{A}_i$ by linear interpolation. Supposing at time t the active point being inserted into $\mathbf{A}_i$'s set of key points is $\mathbf{P}_{i,j}^t$, and $\mathbf{P}_{i,j+1}^t$ and $\mathbf{P}_{i,j-1}^t$ are its neighboring key points, we have:

$$\begin{cases} \mathbf{col}_{i,j}^t = \dfrac{\mathbf{col}_{i,j-1}^t \times \|\mathbf{cc}_{i,j+1}^t - \mathbf{cc}_{i,j}^t\| + \mathbf{col}_{i,j+1}^t \times \|\mathbf{cc}_{i,j}^t - \mathbf{cc}_{i,j-1}^t\|}{\|\mathbf{cc}_{i,j+1}^t - \mathbf{cc}_{i,j-1}^t\|}, \\ wet_{i,j}^t = \dfrac{wet_{i,j-1}^t \times \|\mathbf{cc}_{i,j+1}^t - \mathbf{cc}_{i,j}^t\| + wet_{i,j+1}^t \times \|\mathbf{cc}_{i,j}^t - \mathbf{cc}_{i,j-1}^t\|}{\|\mathbf{cc}_{i,j+1}^t - \mathbf{cc}_{i,j-1}^t\|}. \end{cases} \tag{15}$$

where $\mathbf{col}_{i,j}^t$ is $\mathbf{P}_{i,j}^t$'s color, $\mathbf{cc}_{i,j}^t$ is $\mathbf{P}_{i,j}^t$'s coordinates, and $wet_{i,j}^t$ is $\mathbf{P}_{i,j}^t$'s degree of wetness at time t.

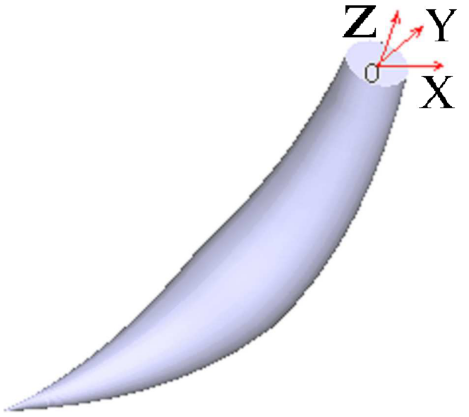

Fig. 13. A deformed writing primitive.

5.2.2. Deformation of the middle control axis A_i

The middle control axis $\mathbf{A}_i$ of $\mathbf{WP}_i$ changes form when subject to forces acting against the brush and the paper. A local reference frame is set up by taking $\mathbf{WP}_i$'s bottom circle $\mathbf{C}_i$ as the X–Y plane, its center $\mathbf{cen}_i$ as the origin, and the brush shaft's direction as the Z-axis (Fig. 13).

If the current active point $\mathbf{P}_{i,j}$ travels a certain distance in the reference frame during time slice t, then all the key points that are underneath the virtual paper $\boldsymbol{\psi}$ will also travel the same distance, plus an additional displacement *dis* in the local reference frame. We estimate this distance to be proportional to the product of **HB**'s elasticity and $\mathbf{WP}_i$'s current pressure, namely $dis = e \times \gamma_i^t$. Note that by using a time slice which is reasonably small, it is safe to assume that this active point remains to be the true active point for the duration covered by the time slice.

5.2.3. Recovery of the middle control axis A_i

As an elastomer, a writing primitive $\mathbf{WP}_i$ will recover in a certain fashion once the outer force exerted on it is released, such as when the brush is partially or completely lifted. Each time when the virtual hairy brush **HB** is lifted, every key point on each of **HB**'s compositive writing primitives will change its place in the 3D space, that is, all the key points' z components will increase by a certain amount. The higher the virtual brush is lifted, the more intense the current writing primitive's inner stress would be, and so would be the recovery. That is, for every key point $\mathbf{P}_{i,j} (i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n_i)$ along $\mathbf{A}_i$ which is deformed, an additional vertical displacement will be exerted to recover its previous deformation:

$$\gamma_i^t \times |sz^{t+1} - sz^t| \times tr(sz^{t+1} - sz^t), \tag{16}$$

where $cz_{i,j}^t$ is $\mathbf{P}_{i,j}$'s coordinates' z component, γ_i^t is $\mathbf{WP}_i$'s pressure against the virtual paper $\boldsymbol{\psi}$ at time t, and the term $|sz^{t+1} - sz^t|$ is the amount by which the virtual hairy brush **HB** is lifted between the time t and $t + 1$; $tr(\,)$ is the truncation function, which is defined as

$$tr(x) \triangleq \begin{cases} 0(x < 0), \\ 1(x \geqslant 0). \end{cases}$$

We have made a simplification in our modeling, which combines the flexibility and the "springiness" of the brush into the single concept of elasticity. The decision was based on the observation that having the distinction between flexibility and springiness would make very minor differences in the final output.

5.2.4. Dynamic adjustment of the wetness

During the writing process, **HB**'s degree of wetness will be dynamically updated. Suppose the intersecting point between the middle control axis $\mathbf{A}_i$ and the virtual paper plane $\boldsymbol{\psi}$, namely the current active point is inserted into $\mathbf{A}_i$'s key points' sequence and denoted as $\mathbf{P}_{i,j}$. The degree of wetness of all the key points on $\mathbf{A}_i$ will decrease by a certain amount because of their contact with or proximity to the paper, which is estimated to be proportional to the product of $\boldsymbol{\psi}$'s ink absorbing ability ab and $\mathbf{WP}_i$'s current pressure γ_i^t against $\boldsymbol{\psi}$:

$$wet_{i,j\pm s}^{t+1} = wet_{i,j\pm s}^{t} - ab \times \gamma_i^t \times \frac{1}{2^{s+d}}, \tag{17}$$

$$\text{where } d = \begin{cases} 2(s \in \mathbf{N}) \\ 1(s = 0) \end{cases} \quad \text{and } (j \pm s) \in [1, n_i].$$

In our current design, the color of **HB** is assumed to be constant throughout the virtual writing and painting process. We plan to add a mechanism to dynamically vary the color of the brush in our future work.

5.3. Dynamic adjustment of the middle control ellipse

The deformed virtual hairy brush has an orientation which is determined by the orientation of the middle control ellipse. Our formulation for the latter is based on the phenomenon that if **HB**'s moving direction is the same as the orientation of writing primitive $\mathbf{WP}_i$'s minor axis $\mathbf{eori}_i^t$ at time t, further writing movements will rotate the ellipse by a certain angle $\mathbf{rot}_i^t$. If the moving direction does not coincide with the orientation of the minor axis, this movement will increase the length of $\mathbf{WP}_i$'s major axis a_i^t; the amount of increase is denoted by $\mathbf{inc}_i^t$. $\mathbf{inc}_i^t$ and $\mathbf{rot}_i^t$ are defined as:

$$\begin{cases} \mathbf{rot}_i^t = re \times \gamma_i^t \times (\mathbf{Vel}^t \cdot \mathbf{eori}_i^t), \\ \mathbf{inc}_i^t = ie \times \gamma_i^t \times \|\mathbf{Vel}^t \times \mathbf{eori}_i^t\|. \end{cases} \tag{18}$$

where re and ie are **HB**'s rotation and elongation coefficients, respectively. Since writing primitive $\mathbf{WP}_i$'s number of hair threads, approximated by $\mathbf{E}_i$'s area, $(\pi \times a_i^t \times b_i^t)/4$, is a constant if $\mathbf{WP}_i$ does not split during its writing process, $\mathbf{E}_i$'s minor axis' length b_i^t can be determined given a certain value for its major axis' length a_i^t.

The middle control ellipse $\mathbf{E}_i$'s position loc_i^t within the middle control axis $\mathbf{A}_i$ may also vary during the writing process. We assume from intuition that the ideal position of the middle control ellipse should be such that it divides the key points of the middle control axis into two equal groups, because at this

location the ellipse's profile has the maximum capacity to control the writing primitive's geometric modeling characters. Since the speed in which the ellipse can relocate is limited by the mechanical and flowage properties of the virtual hairy brush, the actual position of the middle control axis is a linear interpolation of its ideal position (based on the ellipse's ideal location) and its previous position. This is depicted as:

$$loc_i^{t+1} = v \times loc_i^t + (1-v) \times \frac{\sum_{j=1}^{n_i-1} \frac{\sum_{k=1}^{j} \|\mathbf{cc}_{i,k+1}^{t+1} - \mathbf{cc}_{i,k}^{t+1}\|}{\sum_{k=1}^{n_i-1} \|\mathbf{cc}_{i,k+1}^{t+1} - \mathbf{cc}_{i,k}^{t+1}\|}}{n_i}. \tag{19}$$

Here, v is the relocation factor, its default value is $v = 0.75$. And $\mathbf{cc}_{i,k}^t$ is the coordinates of $\mathbf{WP}_i$'s key point $\mathbf{P}_{i,k}$ at time t. The complex summation term is used to estimate the ideal position of the middle control ellipse. Here we use the term $\|\mathbf{cc}_{i,k+1}^{t+1} - \mathbf{cc}_{i,k}^{t+1}\|$ to approximate the distance between the middle control axis $\mathbf{A}_i$'s two neighboring key points $\mathbf{P}_{i,k+1}$ and $\mathbf{P}_{i,k}$ at time $t+1$. Thus, the sum $\sum_{k=1}^{j} \|\mathbf{cc}_{i,k+1}^{t+1} - \mathbf{cc}_{i,k}^{t+1}\|$ is the distance of key point $\mathbf{P}_{i,j+1}$ from the the center of the bottom control circle of the brush, **HB**.**C**. The fraction

$$\frac{\sum_{k=1}^{j} \|\mathbf{cc}_{i,k+1}^{t+1} - \mathbf{cc}_{i,k}^{t+1}\|}{\sum_{k=1}^{n_i-1} \|\mathbf{cc}_{i,k+1}^{t+1} - \mathbf{cc}_{i,k}^{t+1}\|}$$

indicates key point $\mathbf{P}_{i,j+1}$'s position along $\mathbf{A}_i$. According to the assumption about the ideal position of the middle control ellipse, this position actually is the arithmetic mean of all the key points' positions along $\mathbf{A}_i$. Notice that **HB**.**C**'s position along $\mathbf{A}_i$ is always zero.

5.4. Dynamic adjustment of the tip control line

The tip control line of a writing primitive $\mathbf{WP}_i$ is assumed to be a single point in its initial state. It changes into a real line during writing. The line's elongation and rotation are simulated by employing the same strategy as is applied to the middle control ellipse $\mathbf{E}_i$—that is, the tip control line $\mathbf{L}_i$ will increase in length by the amount of $\mathbf{inc}_i^t$ and be rotated by the same amount of $\mathbf{rot}_i^t$ as for the major axis of the middle control ellipse $\mathbf{E}_i$. The tip control line does not have a direct effect on the current ink mark during writing in our modeling, but it will define the shape of the end of a stroke, at the time when the brush is about to leave the paper.

5.5. Splitting of the virtual hairy brush

There is a threshold *tre* which specifies the extent to which $\mathbf{WP}_i$ can be deformed before splitting of the hair occurs. When this threshold is reached or exceeded, the current writing primitive will split into several smaller writing primitives. This simulates the "branching out" behavior of the virtual hairy brush during the writing

process. Specifically, if writing primitive $\mathbf{WP}_i$'s current inner stress str_i^t becomes greater than *tre*, $\mathbf{WP}_i$ will split into

$$k = \lfloor \frac{str_i^t}{tre} \rfloor \tag{20}$$

new writing primitives $\mathbf{WP}_i^1, \mathbf{WP}_i^2, \ldots, \mathbf{WP}_i^k$.

Each of the new writing primitives, $\mathbf{WP}_i^j (j = 1, \ldots, k)$, has a number of hair threads which is equal to $\frac{1}{k}$ of $\mathbf{WP}_i$'s total number. Note that in the virtual hairy brush, we use the area of the middle control ellipse $(\pi \times a_i^t \times b_i^t)/4$ and the length of the tip control line len_i^t to compute the number of hair threads. Therefore the lengths of the middle control ellipse's major axis $a_i^{t,j}$ and minor axis $b_i^{t,j}$ of each of the new writing primitives $\mathbf{WP}_i^j$ are set to $\frac{1}{\sqrt{k}}$ of $\mathbf{WP}_i$'s original values $(j = 1, 2, \ldots, k)$; and the tip control line's length $len_i^{t,j}$ is set to $\frac{1}{k}$ of the original value. That is:

$$\begin{cases} a_i^{t,j} = \frac{a_i^t}{\sqrt{k}} \\ b_i^{t,j} = \frac{b_i^t}{\sqrt{k}} \\ len_i^{t,j} = \frac{len_i^t}{k} \end{cases} \quad (j = 1, 2, \ldots, k). \tag{21}$$

At the beginning, the virtual hairy brush **HB** contains only one writing primitive, and so $m = 1$. During the writing process, the number of **HB**'s compositive writing primitives may increase because of the split operation. A brush with one writing primitive is probably good enough for official scripts, but for cursive scripts, the brush must split into at least a dozen of primitives in order to achieve the necessary effects.

During the split operation, every new writing primitive generated, $\mathbf{WP}_i^j (j = 1, \ldots, k)$, has the same number of key points as the original one, $\mathbf{WP}_i$, with coordinates at a certain distance from $\mathbf{WP}_i$'s. This distance is proportional to the amount of $\mathbf{WP}_i$'s current inner stress exceeding the split threshold *tre*, and the direction of this distance is assumed random. Therefore, for each key point $\mathbf{P}_{i,l} (l = 1, 2, \ldots, n_i)$ in $\mathbf{WP}_i$, there is a corresponding key point $\mathbf{P}_{i,l}^j$ in $\mathbf{WP}_i^j (j = 1, 2, \ldots, k)$. The coordinates of $\mathbf{P}_{i,l}$ and $\mathbf{P}_{i,l}^j$ have the following relationship: $\mathbf{cc}_{\mathbf{P}_{i,l}^j} = \mathbf{cc}_{\mathbf{P}_{i,l}} + \mathbf{S}_{\mathbf{P}_{i,l}^j}$, where $\mathbf{S}_{\mathbf{P}_{i,l}^j}$ is determined by $\mathbf{S}_{\mathbf{P}_{i,l}^j} = \mathbf{rand} \times (str_i^t - tre) \times tr(str_i^t - tre)$, and **rand** is a random unit vector in the 3D space.

5.6. Ink flowage between writing primitives

Although each primitive has full control over its behavior during the writing process, due to reciprocity in mechanics and ink flowage, there could be interaction between writing primitives that are close to each other. To simulate this interaction, we allow each key point's degree of wetness to be affected by its neighboring key points if the distance separating them is within the ink diffusion distance factor η. Linear interpolation is used to compute one key point's current degree of wetness based on its previous value and the average degree of wetness of those neighboring key points. That is formulated as:

$$wet_{i,k}^{t+1} = wet_{i,k}^{t} \times (1 - pw) + \frac{\sum_{j=1,j\neq i}^{m} \sum_{l=1}^{n_j} (wet_{j,l}^{t} \times tr(\eta - \|\mathbf{cc}_{j,l}^{t} - \mathbf{cc}_{i,k}^{t}\|))}{\sum_{j=1,j\neq i}^{m} \sum_{l=1}^{n_j} tr(\eta - \|\mathbf{cc}_{j,l}^{t} - \mathbf{cc}_{i,k}^{t}\|)} \times pw \quad (i = 1, 2, \ldots, m;\ k = 1, 2, \ldots, n_i). \tag{22}$$

Here, pw is the diffusion control factors of virtual hairy brush **HB**; $wet_{i,k}^{t}$ is key point $\mathbf{P}_{i,k}$'s degree of wetness and $\mathbf{cc}_{i,k}^{t}$ is its coordinates at time t; η is the ink diffusion distance factor, and $tr(\,)$ is the truncation function.

6. The writing process

At time t, each of **HB**'s writing primitives will intersect with the virtual paper plane to yield a cross section. The drawing operations are executed taking into account the writing primitives' ink-related information. The following paragraphs outline the algorithm for this process—the virtual hairy brush's real-time writing/painting algorithm.

The algorithm begins with a given writing primitive $\mathbf{WP}_u$ constructed by the general sweeping operation. The generated sweeping surface is a NURBS surface denoted by $SS_u(s,t)$; s and t are the parameters of this parameterized surface, whose values are within $[0,1]$. The direction of s is the same as that of $\mathbf{WP}_u$'s middle control axis. The algorithm intersects $SS_u(s,t)$ with the virtual paper plane $\boldsymbol{\psi}$ to get an intersecting curve cur_u that encloses an area which is the current ink mark (Fig. 14). It can be easily seen that by our intersection operation, the contour of our drawing mark is not always (as a matter of fact, it seldom is) an ellipse. The more deformation our brush tip experiences, the more different our drawing mark would be from an ellipse.

Each point $\mathbf{V}_{u,v}$ in cur_u is projected onto $\mathbf{WP}_u$'s middle control axis $\mathbf{A}_u$ to obtain the point $\hat{\mathbf{V}}_{u,v}$. The ink-related information of the two nearest key points $\boldsymbol{\delta}_{u,v}^{1}, \boldsymbol{\delta}_{u,v}^{2}$ on $\mathbf{A}_u$ to $\hat{\mathbf{V}}_{u,v}$ is then used to compute $\hat{\mathbf{V}}_{u,v}$'s ink-related information by linear interpolation, as follows.

$$\begin{cases} \mathbf{col}_{\hat{\mathbf{V}}_{u,v}} = \dfrac{\mathbf{col}_{\boldsymbol{\delta}_{u,v}^{2}} \times \|\mathbf{cc}_{\boldsymbol{\delta}_{u,v}^{1}} - \mathbf{cc}_{\hat{\mathbf{V}}_{u,v}}\| + \mathbf{col}_{\boldsymbol{\delta}_{u,v}^{1}} \times \|\mathbf{cc}_{\hat{\mathbf{V}}_{u,v}} - \mathbf{cc}_{\boldsymbol{\delta}_{u,v}^{2}}\|}{\|\mathbf{cc}_{\boldsymbol{\delta}_{u,v}^{1}} - \mathbf{cc}_{\boldsymbol{\delta}_{u,v}^{2}}\|}, \\ wet_{\hat{\mathbf{V}}_{u,v}} = \dfrac{wet_{\boldsymbol{\delta}_{u,v}^{2}} \times \|\mathbf{cc}_{\boldsymbol{\delta}_{u,v}^{1}} - \mathbf{cc}_{\hat{\mathbf{V}}_{u,v}}\| + wet_{\boldsymbol{\delta}_{u,v}^{1}} \times \|\mathbf{cc}_{\hat{\mathbf{V}}_{u,v}} - \mathbf{cc}_{\boldsymbol{\delta}_{u,v}^{2}}\|}{\|\mathbf{cc}_{\boldsymbol{\delta}_{u,v}^{1}} - \mathbf{cc}_{\boldsymbol{\delta}_{u,v}^{2}}\|}. \end{cases} \tag{23}$$

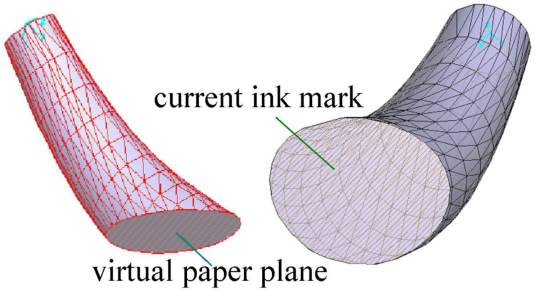

Fig. 14. The virtual paper plane and its drawing mark.

$\mathbf{V}_{u,v}$'s ink-related information is derived from $\hat{\mathbf{V}}_{u,v}$'s by following $\hat{\mathbf{V}}_{u,v}$'s texture function (Eq. 5):

$$\begin{cases} wet_{\mathbf{V}_{u,v}} = wet_{\hat{\mathbf{V}}_{u,v}} \times (1 + \|\mathbf{cc}_{\mathbf{V}_{u,v}} - \mathbf{cc}_{\hat{\mathbf{V}}_{u,v}}\| \times wr_{u,v} + (\mathbf{cc}_{\mathbf{V}_{u,v}} - \mathbf{cc}_{\hat{\mathbf{V}}_{u,v}}) \cdot \mathbf{wv}_{u,v}), \\ \mathbf{col}_{\mathbf{V}_{u,v}} = \mathbf{col}_{\hat{\mathbf{V}}_{u,v}} \times (1 + \|\mathbf{cc}_{\mathbf{V}_{u,v}} - \mathbf{cc}_{\hat{\mathbf{V}}_{u,v}}\| \times cr_{u,v} + (\mathbf{cc}_{\mathbf{V}_{u,v}} - \mathbf{cc}_{\hat{\mathbf{V}}_{u,v}}) \cdot \mathbf{cv}_{u,v}). \end{cases} \tag{24}$$

And for any pixel $\boldsymbol{\tau}$ on the virtual paper plane enclosed by the the curve cur_u, it must lie uniquely on a certain line segment, $\overline{\mathbf{V}_{u,v}\hat{\mathbf{V}}_{u,v}}$. Its ink-related information is computed based on $\mathbf{V}_{u,v}$'s and $\hat{\mathbf{V}}_{u,v}$'s ink-related information:

$$\begin{cases} \mathbf{col}_{\boldsymbol{\tau}} = \frac{\mathbf{col}_{\hat{\mathbf{V}}_{u,v}} \times \|\mathbf{cc}_{\mathbf{V}_{u,v}} - \mathbf{cc}_{\boldsymbol{\tau}}\| + \mathbf{col}_{\mathbf{V}_{u,v}} \times \|\mathbf{cc}_{\boldsymbol{\tau}} - \mathbf{cc}_{\hat{\mathbf{V}}_{u,v}}\|}{\|\mathbf{cc}_{\mathbf{V}_{u,v}} - \mathbf{cc}_{\hat{\mathbf{V}}_{u,v}}\|}, \\ wet_{\boldsymbol{\tau}} = \frac{wet_{\hat{\mathbf{V}}_{u,v}} \times \|\mathbf{cc}_{\mathbf{V}_{u,v}} - \mathbf{cc}_{\boldsymbol{\tau}}\| + wet_{\mathbf{V}_{u,v}} \times \|\mathbf{cc}_{\boldsymbol{\tau}} - \mathbf{cc}_{\hat{\mathbf{V}}_{u,v}}\|}{\|\mathbf{cc}_{\mathbf{V}_{u,v}} - \mathbf{cc}_{\hat{\mathbf{V}}_{u,v}}\|}. \end{cases} \tag{25}$$

Denote the point that is within the volume of the solid model of the virtual hairy brush and that coincides with the pixel $\boldsymbol{\lambda}_{u,v}$ on $\boldsymbol{\psi}$ as $\boldsymbol{\tau}_{u,v}$. $\boldsymbol{\lambda}_{u,v}$'s new degree of wetness is linear-interpolated from $\boldsymbol{\lambda}_{u,v}$'s previous value with $\boldsymbol{\tau}_{u,v}$'s degree of wetness and by using $\boldsymbol{\psi}$'s absorbing ability factor *ab* as the interpolation weight. A drying factor *dry* is introduced to automatically reduce each pixel's degree of wetness periodically until it reaches 0. That is,

$$wet^{t+1}_{\boldsymbol{\lambda}_{u,v}} = (1 - ab) \times wet^{t}_{\boldsymbol{\lambda}_{u,v}} + ab \times wet^{t}_{\boldsymbol{\tau}_{u,v}} - dry. \tag{26}$$

If the degree of wetness exceeds the upper bound of virtual paper $\boldsymbol{\psi}$'s degree of wetness, saturation takes place. We assume that saturation would only affect $\boldsymbol{\lambda}_{u,v}$'s eight neighboring pixels on $\boldsymbol{\psi}$. The degree of wetness of a pixel will increase if it is not saturated. The increased amount is proportional to the unsaturated degree. That is, $\boldsymbol{\lambda}_{i,j}$, which is one of $\boldsymbol{\lambda}_{u,v}$'s eight neighboring pixels on $\boldsymbol{\psi}$, will have its degree of wetness increased according to the following formulation.

$$wet^{t+1}_{\boldsymbol{\lambda}_{i,j}} = wet^{t}_{\boldsymbol{\lambda}_{i,j}} + \frac{(1 - wet^{t}_{\boldsymbol{\lambda}_{i,j}}) \times tr(1 - wet^{t}_{\boldsymbol{\lambda}_{i,j}})}{\varrho^{t}_{u,v}} \times (wet^{t}_{\boldsymbol{\lambda}_{u,v}} - 1) \times tr(wet^{t}_{\boldsymbol{\lambda}_{u,v}} - 1), \tag{27}$$

where $wet^{t}_{\boldsymbol{\lambda}}$, a real number between 0 and 1, is point $\boldsymbol{\lambda}$'s degree of wetness at time t, and

$$\varrho^{t}_{u,v} = \sum_{g=u-1}^{u+1} \sum_{h=v-1}^{v+1} (1 - wet^{t}_{\boldsymbol{\lambda}_{g,h}}) \times tr(1 - wet^{t}_{\boldsymbol{\lambda}_{g,h}}),$$

indicates the total degree of unsaturation that $\boldsymbol{\lambda}_{u,v}$'s eight neighboring pixels have attained. Note that $tr(1 - wet^{t}_{\boldsymbol{\lambda}_{g,h}})$ is non-zero only when the point $\boldsymbol{\lambda}_{g,h}$ is not saturated. Note that it is possible that after one pass of simulated ink diffusion, there could still be some points that are over saturated. This situation would occur if the ink mark originally deposited on the paper is too wet. Thus, in our simulation, an iterative diffusion process is employed, which stops only when all the points on the paper become unsaturated.

The rendering of the current ink-mark is based on a unique ink model we propose. The current color of the pixel $\boldsymbol{\lambda}_{u,v}$ is the linear interpolation of $\boldsymbol{\lambda}_{u,v}$'s previous color and $\boldsymbol{\tau}_{u,v}$'s color, as follows:

$$\mathbf{col}^{t+1}_{\boldsymbol{\lambda}_{u,v}} = \mathbf{col}^{t}_{\boldsymbol{\tau}_{u,v}} \times Cr^{t}_{u,v} + \mathbf{col}^{t}_{\boldsymbol{\lambda}_{u,v}} \times (1 - Cr^{t}_{u,v}), \tag{28}$$

where $Cr^{t}_{u,v}$, the interpolation weight, is a random number, which has the value of either 0 or $wet^{t}_{\boldsymbol{\lambda}_{u,v}}$ based on the following probabilities.

$$\begin{cases} P\{Cr^{t}_{u,v} = 0\} = 1 - min(ren \times ab \times \gamma^{t}_{i}, 1), \\ P\{Cr^{t}_{u,v} = wet^{t}_{\boldsymbol{\lambda}_{u,v}}\} = min(ren \times ab \times \gamma^{t}_{i}, 1), \end{cases} \tag{29}$$

where *ren* is **HB**'s color rendering control factor, *ab* is the virtual paper $\boldsymbol{\psi}$'s absorbing ability, and γ^{t}_{i} is the pressure due to the interaction between the writing primitive and the virtual paper plane. We use this formulation to simulate the dry brush drawing effect and the running style effect.

In the above discussions, for simplicity, to tackle the problem of mixing pigments in full color painting, we adopt directly the RGB color model as our active working color space. Under some very rare circumstances, this might not lead to the best results, for the simple reason that the R, G, and B channels in the physical world are not really independent of each other (see [60] for more details). But this problem is easy to solve: we first convert RGB-formatted ink color to HSV-formatted ink color; in the color space of HSV, we separately process the H, S, and V channels using the same procedures for the R, G, and B channels described above; we finally convert the synthesized results from HSV-formatted ink color back to RGB-formatted ink color for rendering.

7. Customizing the brush

7.1. Quality parameters

In real life, brushes having soft hair tend to branch out easily during writing. Some brushes have a good deal of hair and tend to suck in more ink and cause serious saturation during the writing process, while other brushes have rather long hair and their tip tends to get deformed and rotated to a great extent easily. For our virtual hairy brush, a number of quality parameters can be set to simulate these different kinds of brush character. What comes out as the final electronic artwork from using the virtual hairy brush can be much affected by the values of these parameters. Similarly, the virtual paper has a set of quality parameters to be assigned a value for simulating different kinds of paper.

All the quality parameters **Qp** are classified into two categories. The first category, $\mathbf{Qp}^1 = (e, tre, v, re, ie, sm)$, can affect the brush strokes' boundaries while the second category, $\mathbf{Qp}^2 = (ren, \eta, ab, pw, dry)$, can affect the brush strokes' texture. Different combinations of possible values for these quality parameters of the virtual hairy brush would result in an e-brush with different qualities. Please refer to Fig. 5 for the concise meaning of each of these parameters.

To ease the task of selecting the appropriate values to achieve the desired quality, our implemented system offers a set of predefined quality configurations in a library for the user to choose from. This is similar to what happens in reality when a calligrapher chooses the most suitable real hairy brush from many brushes in his collection or in a shop. Some of which could have been contributed by the users themselves. Others are prefabricated based on empirical knowledge provided by real-life calligraphers and painters. After choosing or creating a certain quality configuration, the end user can write/paint with the chosen or created virtual hairy brush. He can change his decision later and choose another new configuration until he is satisfied with the e-brush and the created artwork. Although not implemented in the current prototype, it is possible to allow an artwork to be automatically transformed using a new configuration because all the input leading to the creation of the artwork is already recorded in a file.

The implemented system provides a window in which the user can adjust the above parameters visually. If he feels that the virtual hairy brush dries too quickly and the final artwork should have more versatile color layers, he can increase the value of the color rendering control parameter *ren*. If he feels that the virtual hairy brush deforms too slowly, he can increase the values of the parameters *v*, *re*, and *ie* which govern the deformability of the brush. And if the virtual hairy brush recovers too quickly from deformation, he can decrease the value of the elasticity parameter *e*. To increase the tendency of brush splitting, he can assign a small value to *tre*, the splitting threshold. If his strokes are fast so that the brush tends to brush out more easily, he may need to adjust the parameters *pw* and *ab* which control the diffusion and absorption abilities of the paper, since fast movements of the brush leave little time for the paper to diffuse or absorb the ink, and hence the proper setting of these parameters is important for any desired effect. Of course, the user can save all this trouble of assigning values to the parameters by simply accepting an offered configuration.

7.2. Configuring the brush with machine intelligence

In addition to user-created configurations, the system can configure a brush automatically. To enable the computer to adjust the quality parameters of the virtual hairy brush as well as those for the virtual paper automatically, a special procedure needs to be carried out to train the computer. The training samples consist of brush strokes being painted within boundaries specified by the training module. The procedure is similar to a beginner starting to learn how to use a hairy brush in real life, referred to as the "MiaoHong" process in Chinese calligraphy. The number of training samples can be set by the user. Of course, the more samples used the better would be the resulting quality of the brush. Simple artwork such as the one in Fig. 19 can be rendered after a few minutes of training.

Let $S^i[len][wid]$ and $C^i[len][wid]$ be two matrices where each of the elements is the RGB color code of a pixel of the virtual paper ψ. The first matrix corresponds to the ith sample of n training samples, and the second matrix the ith user-generated result. Figs. 15 and 16 show some examples of real training samples.

Fig. 15. Some selected training samples for the first category of quality parameters.

Fig. 16. Some selected training samples for the second category of quality parameters.

We define a target function ϑ to indicate the difference between the system's specified training samples and the user-created images using the virtual hairy brush:

$$\vartheta \triangleq \sum_{i=1}^{n} \vartheta^i \triangleq \sum_{i=1}^{n} \sum_{p=1}^{len} \sum_{q=1}^{wid} (S^i[p][q] \otimes C^i[p][q]), \tag{30}$$

where the operator $\otimes$ is defined as $(r_1, g_1, b_1) \otimes (r_2, g_2, b_2) \triangleq |r_1 - r_2| + |g_1 - g_2| + |b_1 - b_2|$. All the quality parameters of the virtual hairy brush contribute to ϑ, in the sense that with a certain set of initial values for all the quality parameters, a collection of user-created images, based on which the corresponding ϑ can be computed. Thus, the problem to determine a good set of values to configure the virtual hairy brush with is reduced to the problem of finding a configuration which can minimize or nearly minimize ϑ, which is a typical optimization problem. We devised and used an optimization algorithm based on the Steepest Descent Algorithm [59] in nonlinear programming to find the needed solution, as described below.

We use ordered training procedures to compute the optimal configuration of **HB**'s quality parameters **Qp**. First, we need to train and determine all the first-category quality parameters $\mathbf{Qp}^1$, followed by the second-category parameters $\mathbf{Qp}^2$. Like the quality parameters, the training samples are divided into two classes. The first class of samples are used to train the e-brush to draw brush strokes with proper stroke boundaries, and so parameters that have to do with the texture of the brush stroke are ignored, namely $\mathbf{Qp}^2$. Hence, to determine $\mathbf{Qp}^1$, our virtual brush simulates no drying brush effect nor diffusion effect; that is, all the pixels on the virtual paper that are covered by the ink mark at a certain simulated time are rendered as completely dark. After the algorithm yields the value for $\mathbf{Qp}^1$, the value for $\mathbf{Qp}^2$ is computed, this time with $\mathbf{Qp}^1$ being assigned the value derived from the previous computation.

Because the values of all the quality parameters of the virtual hairy brush share the same range of [0,1], we can view this problem as a minimization problem with its feasible solution space being a unit cube in a 6-dimensional space. We equally divide the edges of the cube to decompose the cube into subcubes, and take every

resultant grid point (at a corner of a subcube) as a possible initial point; so each initial point's coordinates in the 5-dimensional space are in the form of $\mathbf{X}^0 = (\frac{d_1}{d}, \frac{d_2}{d}, \ldots, \frac{d_6}{d})$, where $d \in N$ is the number of parts an edge is divided into, and $d_i = 0, 1, \ldots, d (i = 1, 2, \ldots, 6)$. With $\mathbf{X}^0$ as the initial point, we use the Steepest Descent Algorithm [59] to find a satisfactory solution that minimizes $\vartheta(\mathbf{X}^0)$. That is,

1. Set $k = 0$. Compute $\mathbf{y}^k = -\nabla\vartheta(\mathbf{X}^k)$;
2. If $\|\mathbf{y}^k\| < \varepsilon$, the algorithm stops, where ε is the error bound;
3. Determine $\boldsymbol{\lambda}_k$, such that $\vartheta(\mathbf{X}^k + \boldsymbol{\lambda}_k \times \mathbf{y}^k) = min(\mathbf{X}^k + \boldsymbol{\lambda} \times \mathbf{y}^k), \boldsymbol{\lambda} \geqslant 0$;
4. Let $\mathbf{X}^{k+1} = \mathbf{X}^k + \boldsymbol{\lambda}_k \times \mathbf{y}^k$; $k = k + 1$; go to Step 2.

The value of $\nabla\vartheta(\mathbf{X}^k)$ is approximated by ϑ's differential at the point $\mathbf{X}^k$. That is, we disturb the position of $\mathbf{X}^k$ along the coordinate system's axis x_i by a short distance $\triangle x_i$; and run the algorithm for the virtual hairy brush to yield $\vartheta(\mathbf{X}^k + \triangle x_i)$. The differential of $\vartheta(\mathbf{X}^k)$ can then be approximated by $\frac{\vartheta(\mathbf{X}^k + \triangle x_i)}{\triangle x_i}$. After taking several $(\frac{d_1}{d}, \frac{d_2}{d}, \ldots, \frac{d_6}{d}), d_i = 0, 1, \ldots, d (i = 1, \ldots, 6)$, as the initial point $\mathbf{X}^0$, we can determine an $\mathbf{X}$ $(= (\frac{d_{i,1}}{d}, \frac{d_{i,2}}{d}, \ldots, \frac{d_{i,6}}{d}))$ which yields the nearly minimum $\vartheta(\mathbf{X})$. We then take $[\frac{d_{i,1}-1}{d}, \frac{d_{i,1}+1}{d}] \times [\frac{d_{i,2}-1}{d}, \frac{d_{i,2}+1}{d}] \times \cdots \times [\frac{d_{i,6}-1}{d}, \frac{d_{i,6}+1}{d}]$ as a new cube, and re-run the above procedure to search for a smaller ϑ, until meeting a solution that satisfies the user's specified error bound.

Of course, the user can adjust the the parameters of the configuration if he feels that the result from the above procedure still falls short of his demand. Any of the preset configurations in the system's library can be used as the initial point for the optimization procedure so that a solution to the problem can be arrived at much more quickly and accurately. In this sense, all the above strategies to determine the quality parameters for the virtual hairy brush can be combined to achieve better performance and results.

8. System implementation and experiment results

Fig. 17 shows a screen shot of the implemented system in action, where there is one window responsible for displaying the current sampled input and the history of all the sampled inputs (window 8); one for the current drawing mark (window 3); one for the current writing mark imprinted on the paper (window 1). There are several additional windows for displaying the 3D model of the virtual hairy brush in action, including: one for the part of virtual brush penetrating the virtual paper (window 2); one for the complete solid model of the virtual brush in the 3D space (window 4); one for the 3D model of virtual brush in silhouette form (window 6); one for the part of the virtual brush which is above the virtual paper (window 5), and one for the part that is below (window 7). There are several other optional windows which can be displayed by the user's choice. They include a window to display the current control parameters derived from the input, one for the parameter values automatically assigned to the writing primitives by an internal algorithm, and one for the configuration of current quality parameters.

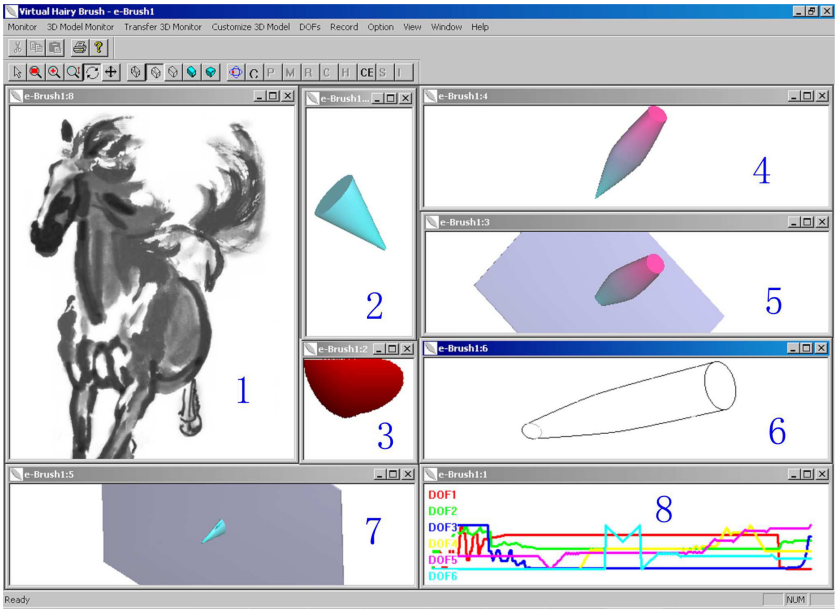

Fig. 17. The running system.

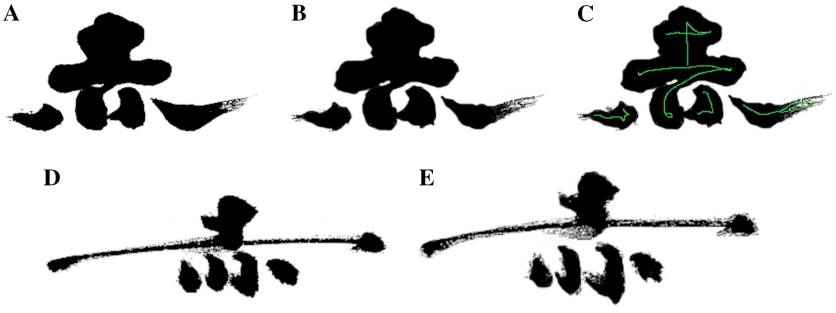

Fig. 18. Real artwork (A,D) and imitations (B,E). The brush's trajectories at (B) are highlighted at (C).

Figs. 18–20 show some real artwork digitized from calligraphy samples together with the imitation artwork created by our virtual hairy brush. We highlight using red circles a few spots that are the result of the dry brush effect in our design; such an effect is difficult to achieve in other e-brush projects. These samples prove that very realistic-looking calligraphic artwork can be generated by our virtual hairy brush. If carefully tuned, the simulation's result can be nearly indistinguishable from the original one to a human viewer. By fiddling with the quality parameters and the input data for the brush's six degrees of freedom, users can create interesting calligraphy fully electronically. It is well known that to imitate an original calligraphic artwork is a nontrivial task. With our electronic environment, however, such an imitation task becomes relatively easy. The system offers a friendly user interface,

Fig. 19. Real artwork (left column) vs. imitation (middle column). And in the right column, the rich and fine details of the dry brush effect produced by our virtual brush are enclosed by red ellipses.

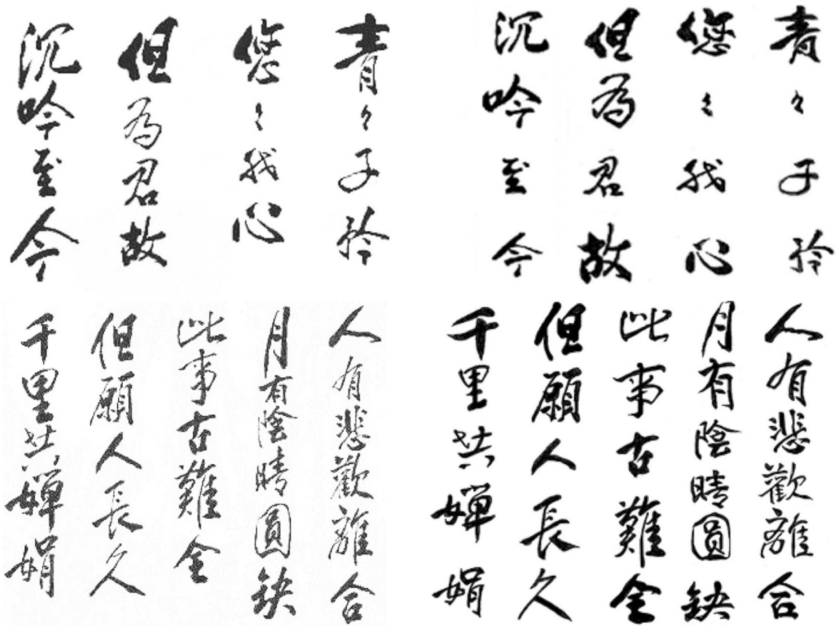

Fig. 20. Two Chinese love poems: real artwork (left) vs. imitation (right).

which supports adjusting the quality parameters and the input data dynamically to achieve whatever delicate effect the user desires. Fig. 21–25 show a series of computer artwork created using the virtual hairy brush. To give an idea on the effort required

Fig. 21. Pinasters (left) and stones (right).

Fig. 22. A running horse (left) and parrots (right).

Fig. 23. Orchid.

Fig. 24. Bamboos.

Fig. 25. Song birds (left) and mandarin ducks (right).

for the more complex artwork, the horse picture (the horse in the left part of Fig. 22) took about 1.5 hours to complete.

Although these figures show only imitation artwork generated by our virtual brush, the real imitation that our design and implementation have set out to achieve is to imitate the manipulatability, aesthetic features, and expressive power of tradi-

tional hairy brushes. A person who has mastered our virtual brush should be able to create high-quality calligraphic artwork as can be done by a real brush.

9. Related work

Pure hardware approaches such as Greene's [33] tend to be expensive and not easily applicable in all environments. The method by Pan et al. [14] can only generate new fonts from existing ones. The method by Way and Shih [45] requires the user to specify the contour and the parameters of the object to be painted via a reference image or figure, which deviates from the way traditional artists perform their painting tasks.

In the following, we make a detailed comparison with two projects that in many ways are closest among all related work to our work—the first one is the DAB system by Baxter et al. [47], a painting system based on a deformable 3D brush model and with an intuitive haptic interface; the second work is the "virtual brush" by Wong and Ip [26], a model-based approach to synthesizing realistic Chinese calligraphic writings. We also survey briefly several other e-brush systems.

9.1. DAB

Although DAB is a complex system [47], it cannot capture all the complex physical properties and conditions of a hairy brush and its ink distribution, which are necessary for simulating Chinese calligraphy and painting. DAB relies on a special device with a 6-DOF input, and a 3-DOF force output that implements a haptic feedback. The device is expensive, and therefore limits its wide application. In comparison, either a tablet or a keyboard-and-mouse strategy would suffice as an input method for our e-brush prototype. These simple methods, when coupled with the simulation of brush stroke inertia and a machine intelligence component, provide a suitable level of comfort and manipulatability to the user. DAB's complex algorithms require the use of powerful hardware to give real-time response. Our e-brush system can achieve interactive response on an ordinary PC due to a carefully picked tradeoff between the quality of the final rendering and computation efficiency.

In DAB, a layer carries only one color, and blending of colors happens when different layers interact. Our e-brush employs a more direct approach, with the pigment model being an integral part of the e-brush, which makes it possible to simulate highly complex ink distribution patterns within the e-brush volume. This enables the user to create more elaborate and complex blends in a more natural manner than DAB.

DAB's one-color-per-layer strategy limits the richness of the output texture. It seems not at all easy to improve on this, such as by adding ink-related information to DAB's simple particle-system model or its complex subdivision model. DAB needs to integrate a new, elaborate pigment model in order to generate varied-texture output. In our model, ink-related information are naturally and inherently embedded in the e-brush, which makes possible the efficient and powerful rendering of

ink marks with complex textures. We have also designed a new pen-and-ink model to render the brush's footprint, which is probabilistic and considers the pressure on the virtual paper, the inner stress of the e-brush, and the wetness of both the brush and the virtual paper. This should be much better that DAB's simple alpha blending. Besides, our e-brush is tuned to optimal performance in handling paint representation, paint mixing, and the drying of the canvas.

DAB uses two separate models—a subdivision surface mesh wrapped around a spring-mass particle system skeleton for the basic motion and behavior of the brush head, and a deformable mesh skinned around this skeleton for the actual shape of the head—which leads to the synchronization problem of how to properly anchor the control points of the subdivision surface relative to the mass particles. To have a tradeoff between the ease to place the control points and the modeling ability makes DAB only capable of supporting four styles of brushes used in oil-like painting. In comparison, we use four compositive attributes and a suitably powerful modeling metaphor, the general sweeping operation, to model our e-brush. Thus, our e-brush is much simpler than DAB's as we use only one model to cover the geometrical shape, the dynamic behavior, and pigment changes. The result is a simulated brush capable of expressing all kinds of brush tips, and better efficiency.

DAB likens the brush tip to a piece of cloth, and uses cloth simulation to construct the brush surface. To solve for the mesh at the brush tip, DAB relies on an approximated implicit integration method. This poses two problems: modeling with cloth is not the best choice as far as approximating the reality is concerned, and solving for the cloth can make the system unstable. To maintain stability, modeling accuracy has to be sacrificed. In our modeling, by relying on the concept of writing primitives and choosing the general sweeping operation as the modeling metaphor, our e-brush can readily simulate almost all kinds of brush heads and their deformation, including splitting of the brush tip. The general sweeping operation also has an advantage in terms of the running system's stability. Hence, our e-brush is more stable, efficient, and expressive than DAB's.

The simulation of our e-brush's dynamics and its visual effects incurs less cost than DAB. This is due to our combining the geometrical modeling, dynamic behavior, and paint blending all into a single model which is evaluated only once per time slice. Because of the reduced complexity of the e-brush simulation, we could use the saved time and resources to perform a more delicate simulation of the brush's dynamic deformation behavior. The simulated deformation of our virtual brush can be in a variety of ways and to a large extent, which is not the case with DAB.

9.2. Virtual brush by Wong and Ip

The virtual brush model by Wong and Ip [26] also offers a feasible means for artists to produce Chinese calligraphic characters electronically. Their method, however, is inconvenient to use because of an intricate set of interrelated parameters for controlling the shape, density, and opacity of the current drawing mark. These parameters need to be specified manually by the user, thus limiting the interactivity of the system. More specifically, the user needs to specify the profiles for seven pa-

rameters to control the brush motion dynamic model, and another three parameters to control the ink deposition—a total of ten profiles. Only two of these profiles, the stroke's trajectory in the X and Y directions, can be sampled by a mouse or a digitizing pen; so there are eight profiles to be input manually. Whereas in our approach, there are many input devices that can take in the six degrees of freedom of a solid object, and no profile needs to be supplied by the user. For instance, MacKenzie [64] discusses input devices that can capture six degrees of freedom for brush/paint systems. Ware and Baxter [63] describes a model for converting hand position and orientation into six useful variables for computer input and its application in an experimental computer paint program.

Other than the input parameters for the brush's six degrees of freedom, several parameters have been introduced in our design for controlling the virtual brush's quality, whose values can be automatically set via an optimization algorithm. Although our e-brush model is more elaborate than that of Wong and Ip, all the modeling details and quality parameters of the e-brush can be hidden from the end user. The user can in fact use the e-brush in the same way he/she uses the traditional hairy brush. We can therefore say that our e-brush is simpler and easier to manipulate than the e-brush by Wong and Ip. But because of the more complex model we use behind the scene and the automatically tunable quality parameters we introduce, we can achieve better output results with our e-brush.

Referring to the examples in Figs. 19 and 20, although it may be possible for an experienced artist to produce a similar artistic calligraphy result using the system described in [26]; but as we have already explained, that will take a much greater effort because of the need to adjust many parameters that control the shape as well as the texture of the strokes. In comparison, our brush is much more "usable"—it is easy to use but yet sufficiently expressive. Users can conveniently manipulate our e-brush by controlling its six DOFs using either a tablet pen or the keyboard plus a mouse.

Although [26] provides an extensible library of pre-set stroke parameters through a convenient panel. During the digital painting process, however, these parameters are visible to and need to be adjusted by the end users. Also, some of those parameters are not quite intuitive. In comparison, our e-brush hides all the complex parameters and only requires the six DOFs input from the user. The quality parameters associated with our e-brush all have an intuitive meaning and are therefore easy to adjust by the user. A machine training module can further make the quality parameter tuning process intelligent and most convenient to the user.

9.3. Other virtual brush models

The very recent work by Chu and Tai [61] proposed a virtual brush which is very similar to ours. The geometrical aspects of their brush are essentially the same as those in an early report of our work [1]. They however use simple blending for paint mixing, which severely restricts the richness of the resulting e-paintings. They model the dynamics of the e-brush as springs which are deformed through constrained energy minimization. This way of modeling can only simulate small-scale deformation of the e-brush but not large-scale bending or stretching due to the restriction of con-

strained energy minimization. By the same reason, spreading and splitting of brush tips cannot be simulated physically in their system. [62] presents two methods to create Chinese paintings. One of them creates 3D Chinese painting animation using existing commercial software packages. Their second method is based on a simple strategy in which many bristles which are arranged randomly within a circular area constitute the brush tip. As a result, only a small number of modeling effects are possible using this method. There is no formal ink model being proposed for their system. The water in each brush bristle is directly used as the transparency for the generated brush strokes, and so both the shapes and the textures of the strokes produced by their system appear stiff and artificial. Their results are far less expressive than the real artwork.

10. Future work

A real brush operates in a fashion that is orders of magnitude more complex than can be modeled by a computer. Features that can be added in the future versions to further enhance the modeling include dynamic merging of writing primitives, repeated dipping effects, and more user's control of the brush during writing. Other features requiring longer-term effort include mapping of 2D calligraphic artwork to 3D and feature extraction of real artwork. We explain some of these in the following.

- **More delicate modeling.** This includes making the elasticity of our brush hair a variable rather than a constant in our system; and adopting a more sophisticated model for hair splitting rather than the current random splitting since the probability of hair threads on the outside splitting should be larger than that of the inner ones.
- **Multiple dipping effects.** In the current design, dipping operations only take place before the first time the virtual hairy brush touches the paper plane. In real life, an artist may dip the brush either partially or completely multiple times. Partial dipping would cause local changes in the brush's wetness and color.
- **More control.** Another problem with the current prototype is that, once the simulation is on its way, the number of key points in the middle control axis would keep increasing, which is not quite in line with the real case where an artist may caress a deformed brush to restore its shape. A variety of interfaces are required to allow the user to control and edit the writing primitive's geometry, as well as its color and its degree of wetness as the linear interpolation in the current design is just one of many possibilities.
- **Hair merging.** In the current virtual hairy brush model, writing primitives will split into several smaller ones, which simulates the branching out of the hair in a real writing process. In real life, branched-out hair bundles may merge together later. Such a feature is not in the current implementation.
- **3D writing effects.** Many famous Chinese calligraphic artwork creations are chiseled in monuments. How to map 2D calligraphy work into 3D is an interesting research problem. So is the problem of directly creating 3D calligraphic artwork.

- **Automatic imitation.** Pattern recognition and machine learning mechanisms can equip the system with the ability to duplicate existing artwork fully automatically, and in the process to extract useful components such as the brush's path and the thicknesses of the strokes. These components will then be useful in automatic imitation of calligraphic artwork or computer-assisted calligraphy.

11. Summary and conclusion

In this paper, by modeling the hairy brush using a few writing primitives which work independently, a real-time virtual hairy brush system has been implemented for creating artwork by the computer. Against the criteria presented at the beginning of this paper, our e-brush is easy to use, and people can more or less treat the e-brush like a traditional brush; with the support of an input component and a machine intelligence component, it is quite comfortable and easy to manipulate; the output samples show that indeed our e-brush can generate more expressive and realistic rendering results than all the previous e-brush systems; the machine intelligence component allows different levels of automation in controlling the e-brush to be enabled; fast response time is guaranteed due to the use and efficient implementation of writing primitives; and our e-brush can offer more functionalities than traditional brushes and other e-brushes via the machine intelligence component.

We use writing primitives instead of individual hair threads to serve as the basic working units in the virtual hairy brush. In general, a single primitive is probably enough for regular scripts, official scripts, and most running scripts, and perhaps a dozen or so for cursive scripts. Hence, our software needs to handle only a small number of writing primitives most of the time. In comparison with [26], where the brush is modeled as a cone, our virtual brush can produce a drawing mark that is more variable in shape.

Like the virtual hairy brush itself, the proposed ink model is also physically based, which blends wetness, ink color, and the current inner stress of the brush with rendering probabilities, and reduces all the complex interactions into a set of simple equations. Compared to [48], where complex differential equations are used, our model is computationally efficient. The ink model is integrated into the e-brush system so that much of the computation can also be used in computing for the ink model.

With the ink-related information that is lodged in the writing primitive's control axis, a single primitive can readily express complex, interesting and even mysterious distribution of the ink, color, and wetness. Together with the use of rendering probabilities, our brush can achieve the effects of multiple gray levels, colorful painting, dry brush writing, and saturation.

During the writing process, the simulation takes into account the acceleration of the virtual hairy brush, producing an effect that emulates the manipulation of a real physical brush.

Although the system provides methods to manually edit a collection of parameters that can affect the rendering and the final result, it is not necessary for the user

to interactively input values for these parameters. All that is needed from the user is the series of six degrees of freedom of the hairy brush, based on which the computer can simulate the whole writing process. This leads to an efficient system operable in real-time, and which is easy to use for the user. Since most of the handwriting's geometrical and dynamic parameters can be automatically tuned by the system on the fly, little storage space is required to keep track of state information.

The simplicity and expressive power of the system introduce a wide space to carry out computer-aided artwork creation.

Acknowledgments

We thank www.chinapage.com for supplying us with real artwork by famous Chinese calligraphers. We are grateful for the comments made by the Eurographics 2002's reviewers and the help from Victor Ostromoukhov.

References

[1] S. Xu, M. Tang, F.C.M. Lau, Y.-H. Pan, A solid model based virtual hairy brush, in: Eurographics 2002, Saarbrucken, Germany, 2002. (Also Computer Graphics Forum, 21(3) (2002) 299–308).
[2] J. Lee, Diffusion rendering of black ink paintings using new paper and ink models, Computers and Graphics 25 (2) (2001) 295–308.
[3] Y.S. Chua, Bezier brushstrokes, Comput. Aided Des. 22 (9) (1990) 550–555.
[4] T. Nishita, S. Takita, E. Nakamae, A display algorithm of brush strokes using Bezier functions, Proc. CG Int.'93 (1993) 244–257.
[5] B. Pham, Expressive brush strokes, CVGIP: Graph. Models Image Process. 53 (1) (1991) 1–6.
[6] S.C. Hsu, I.H.H. Lee, Drawing and animation using skeletal strokes, in: Proc. of SIGGRAPH '94, Orlando, Florida, July 24–29, 1994, pp. 109–118, July 1994.
[7] John D. Hobby, Digitized Brush Trajectories. Ph.D. thesis, Stanford University, 1985. (Also Technical Report STAN-CS-85-1070, Stanford University, 1985).
[8] K.C. Posch, W.D. Fellner, The circle-brush algorithm, ACM Trans. Graph. (TOG) 8 (1) (1989) 1–24.
[9] J.-W. Ahn, M.-S. Kim, S.-B. Lim, Approximate general sweep boundary of a 2D curved object, CVGIP: Graph. Models Image Process. 55 (2) (1993) 98–128.
[10] M. Shiono, O. Yoshimura, H. Sanada, Y. Tezuka, Generation of brush written Kanji patterns using brush touch function controlled by three dimensional motion of brush core, Trans. IEICE Japan J72-D-11 (1) (1989) 76–84.
[11] S.-B. Lim, M.-S. Kim, Oriental character font design by a structured composition of stroke elements, Comput. Aided Des. 27 (3) (1995) 193–207.
[12] L. Shao, H. Zhou, A new contour fill algorithm for outlined character image generation, Comput. Graph. 19 (4) (1995) 551–556.
[13] P. Coueignoux, Character generation by computer, Comput. Graph. Image Process. 16 (3) (1981) 240–269.
[14] Z. Pan, X. Ma, M. Zhang, J. Shi, Chinese font composition method based on algebraic system of geometric shapes, Comput. Graph. 21 (3) (1997) 321–328.
[15] A. Shamir, A. Rappoport, Quality enhancements of digital outline fonts, Comput. Graph., Special Issue on Graphics in Electronic Printing and Publishing 21 (6) (1997) 713–725.
[16] H.S.H. Ip, T.F.H. Wong, Y.F. Mong, Fractal coding of chinese scalable calligraphic fonts, Comput. Graph. 18 (3) (1994) 343–351.

[17] Q. Guo, Rendering calligraphy words with 'Kasure' variations, in: Proceedings of the Six International Conference on Human-Computer Interaction (HCI International'95), 1995, pp. 129–134.
[18] Q. Guo, Generating realistic calligraphy words, IEICE Trans. Fundam. Electr. Commun. Comput. Sci. E78-A (11) (1995).
[19] J.S. Lee, C.M. Wang, Computer-generated calligraphy: a precise calligraphic learning system, in: Proc. of E-Learning and Continuing Professional Education 2001, Chiayi, Taiwan, 2001, pp. (I)167–172.
[20] T. Yamasaki, S. Inokuchi, Computer coaching for beautiful handwriting of Japanese characters in elementary school, in: Proc. 5th World Conf. on Computers in Education (WCCE90), Sydney, Australia, 1990, pp. 725–728.
[21] T. Yamasaki, T. Hattori, Training system for well writing of Chinese characters based on their local structure, IEICE Trans. Japan J67-D (4) (1984) 442–449.
[22] T. Yamasaki, M. Yamamoto, S. Inokuchi, CAI system for acquiring good writing skills based on the analysis of pen speed, IEICE Trans. Japan J70-D (11) (1987) 2071–2076.
[23] J. Zeng, X. Zhang, H. Sanada, Y. Tezuka, A computer generation-model of brush-used handwritten chinese characters and its application in education, in: P.R. Leedham (Ed.), Computer Processing of Handwriting, World Scientific Publishing, 1990, pp. 363–400.
[24] K. Henmi, T. Yoshikawa, Virtual lesson and its application to virtual calligraphy system, Proc. of 1998 IEEE Int. Conf. Robot. Automat. (1998) 1275–1280.
[25] S. Strassmann, Hairy brush, Proc. SIGGRAPH '86 20 (4) (1986) 225–232.
[26] H.T.F. Wong, H.H.S. Ip, Virtual brush: a model-based synthesis of Chinese calligraphy, Comput. Graph. 24 (1) (2000) 99–113.
[27] T. Yamasaki, T. Hattori, Computer calligraphy—brush written kanji formation based on the brush-touch movement, in: Proc. of IEEE International Conference on Systems, Man and Cybernetics, IEEE Press, 1996, pp. 1736–1741.
[28] J.S. Lee, C.M. Wang, Toward a physically-based model for monochromatic ink rendering, CD-ROM Proc. 14th Comput. Vis. Graph. Image Process. (2001).
[29] M.C. Lin, A. Gregory, S. Ehmann, S. Gottschalk, and R. Taylor, Contact determination for real-time haptic interaction in 3D modeling, editing and painting, in: Proc. of 1999 Workshop for PhanTom User Group, 1999.
[30] S. Hangai, S. Yamanaka, T. Hamamoto, Writer verification using altitude and direction of pen movement, Proc. Int. Conf. Pattern Recognit. 3 (2000) 483–486.
[31] J. Mano, L. He, T. Nakamura, H. Enowaki, A. Mutoh, H. Itoh, A method to generate writing-brush-style Japanese Hiragana character calligraphy, in: Proceedings of IEEE International Conference on Multimedia Computing and Systems, vol. 1, June 7–11, Florence, Italy, 1999, pp. 787–791.
[32] J. Lee, Physically-based modeling of brush painting, Comput. Networks ISDN Syst. 29 (14) (1997) 1571–1576.
[33] R. Greene, The drawing prism: a versatile graphic input device, Proc. SIGGRAPH '85 (1985) 103–110.
[34] T.L. Kunii, G.V. Nosovskij, T. Hayashi, A diffusion model for computer animation of diffuse ink painting, in: Proc. of Computer Animation '95, April 19–21, Geneva, Switzerland, 1995, pp. 98–102.
[35] Q. Zhang, Y. Sato, J. Takahashi, K. Muraoka, N. Chiba, Simple cellular automaton-based simulation of ink behavior and its application to Suibokuga-like 3D rendering of trees, J. Vis. Comput. Animat. 10 (1) (1999) 27–37.
[36] T.L. Kunii, G.V. Nosovskij, V.L. Vecherlinin, Two-dimensional diffusion model for diffuse ink painting, Int. J. Shape Model. 7 (1) (2001) 45–58, World Scientific Publishing Company, Singapore.
[37] Q. Guo, T.L. Kunii, Modeling the diffuse paintings of 'Sumie', in: B. Falcidieno, T.L. Kunii (Eds.), Modeling in Computer Graphics (Proceedings of IFIP), Springer, New York, 1991, pp. 329–338.
[38] T. Nakamura, H. Itoh, H. Seki, T. Law, A writing system for brush characters using neural recognition and fuzzy interpretation, Proc. 1993 Int. Joint Conf. Neural Networks (1993) 2901–2904.
[39] T. Yamasaki, T. Hattori, Forming square-styled brush written kanji through calligraphic skill knowledge, in: IEEE 1996 International Conference on Multimedia Computing and Systems (ICMCS '96), June 17–23. Hiroshima, Japan, 1996, pp. 501–505.

[40] X. Wei, S. Lu, M. Song, B. Luo, Computer pattern design and painting technique based on aesthetics knowledge, Comput. Aided Draft., Des. Manufact. 2 (2) (1992) 32–40.
[41] T. Nakamura, H. Itoh, H. Seki, T. Law, A writing system for brush characters using neural recognition and fuzzy interpretation, in: Proc. of 1993 International Joint Conference on Neural Networks, 1993, pp. 2901–2904.
[42] T. Yamasaki, T. Hattori, Computer calligraphy—brush written Kanji formation based on the calligraphic skill knowledge, IEICE Trans. Inf. Syst. E80-D (2) (1997).
[43] Y. Pang, H. Zhong, T. Kunii, Drawing Chinese traditional painting by computer, in: Modeling in Computer Graphics, Proceedings of IFIP WG5.10. Berlin, Springer, 1991, pp. 321–328.
[44] Y.J. Pang, H.X. Zhong, Drawing Chinese traditional painting by computer, in: Modeling in Computer Graphics, Proceedings of IFIP, 1991, pp. 321–328.
[45] D.-L., Way, Z.-C. Shih, The synthesis of rock textures in Chinese landscape painting, in: Proc. of Eurographics 2001, 2001, pp. C123–C131.
[46] J. Lee, Simulating oriental black-ink painting, IEEE Comput. Graph. Appl. 19 (3) (1999) 74–81.
[47] B. Baxter, V. Scheib, M.C. Lin, D. Manocha, DAB: interactive haptic painting with 3D virtual brushes, Proc. SIGGRAPH 2001 (2001) 461–468.
[48] C.J. Curtis, S.E. Anderson, J.E. Seims, K.W. Fleischer, D.H. Salesin, Computer-generated watercolor, Proc. SIGGRAPH '97 (1997) 421–430.
[49] A. Hertzmann, Painterly rendering with curved brush strokes of multiple sizes, Proc. SIGGRAPH '98 (1998) 453–460.
[50] S. Saito, M. Nakajima, 3D physically based brush model for painting, SIGGRAPH '99 Conf. Abstracts Appl. (1999) 226.
[51] A. Gregory, S. Ehmann, M.C. Lin, inTouch: interactive multiresolution modeling and 3D painting with a haptic interface, in: Proceedings of IEEE Virtual Reality Conference, 2000.
[52] J. Lansdown, S. Schofield, Expressive rendering: a review of nonphotorealistic techniques, IEEE Comput. Graph. Appl. 15 (3) (1995) 29–37.
[53] G. Winkenbach, D.H. Salesin, Computer-generated pen and ink illustration, Proc. SIGGRAPH '94 (1994) 91–100.
[54] L. Markosian, M.A. Kowalski, D. Goldstein, S.J. Trychin, J.F. Hughes, L.D. Bourdev, Real-time nonphotorealistic rendering, Proc. SIGGRAPH 97 (1997) 414–420.
[55] C. Maurer, B. Juttler, Rational approximation of rotation minimizing frames using Pythagorean-hodograph cubics, J. Geometry Graph. 3 (2) (1999) 141–159.
[56] M. Agrawala, A. Beers, M. Levoy, 3D painting on scanned surfaces, in: Proc. of 1995 Symp. on Interactive 3D Graphics, 1995, pp. 145–150.
[57] D. Hohnson, T.V. Thompson II, M. Kaplan, D. Nelson, E. Cohen, Painting textures with a haptic interface, in: Proc. of IEEE Virtual Reality Conference, 1999.
[58] L.D. Landau, E.M. Lifshitz, in: Theory of Elasticity, Course of Theoretical Physics, vol. 7, Pergamon Press, Oxford, 1986.
[59] D.G. Luenberger, Linear and Nonlinear Programming, second ed., Addison-Wesley, Reading, MA, 1984.
[60] Joint ISO/CIE Standard ISO 10526:1999/CIE S005/E-1998 CIE Standard Illuminants for Colorimetry. International Commission on Illumination.
[61] N. Chu, C. Tai, An Efficient Brush Model for Physically-Based 3D Painting, in: Proc. of Pacific Graphics 2002, IEEE Computer Society, 9–11 October, Beijing, China, 2002, pp. 413–421.
[62] C. Chan, E. Akleman, J. Chen, Two Methods for Creating Chinese Painting, in: Proc. of Pacific Graphics 2002, IEEE Computer Society, 9–11 October, Beijing, China, 2002, pp. 403–412.
[63] C. Ware, C. Baxter, Bat brushes: on the uses of six position and orientation parameters in a paint program, ACM SIGCHI Bull., v.20 (1989) 155–160.
[64] I.S. MacKenzie, in: W. Barfield, T.A. FurnessIII (Eds.), Input devices and interaction techniques for advanced computing, virtual environments and advanced interface design, Oxford University Press, Oxford, UK, 1995, pp. 437–470.