Virtual hairy brush for digital painting and calligraphy

XU Songhua[1, 2], Lau Francis C. M.[3], XU Congfu[1] & PAN Yunhe[1]

1. State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310027, China;
2. Computer Science Department, Yale University, New Haven, CT 06511, USA;
3. Department of Computer Science, The University of Hong Kong, Pokfulam Road, Hong Kong, China
Correspondence should be addressed to Xu Songhua (email: songhua.xu@yale.edu)
Received August 3, 2003; received November 19, 2004

Abstract The design of user friendly and expressive virtual brush systems for interactive digital painting and calligraphy has attracted a lot of attention and effort in both computer graphics and human-computer interaction circles for a long time. Providing a digital environment for paper-less artwork creation is not only challenging in terms of algorithmic design, but also promising for its potential market values. This paper proposes a novel algorithmic framework for interactive digital painting and calligraphy based a novel virtual hairy brush model. The algorithms in the kernel of our simulation framework are built upon solid modeling techniques. Implementing the algorithms, we have developed a virtual hairy brush prototype system with which end users can interactively produce high-quality digital paintings and calligraphic artwork. (The latest progress of our virtual brush project is reported at the website "http://www.cs.hku.hk/~songhua/e-brush/".)

Keywords: virtual hairy brush, digital painting and calligraphy, simulation algorithm, solid modeling, non-photorealistic rendering.

DOI: 10.1360/03yf0389

The art of Chinese calligraphy and painting has evolved incessantly over the long history of Chinese culture. It is a beautiful flower in the garden of traditional oriental art forms[1]. Since the emergence of modern computers, to design and develop an interactive and user friendly digital painting environment has been the long cherished dream for many researchers in computer graphics and human-computer interaction[2—10]. With the ever-increasing computing power of today's hardware, researchers can now make the dream a reality. This paper proposes a novel solid model based virtual hairy brush system for interactive digital painting and calligraphy[11—13]. Experiment results show that end users can produce expressive electronic painting and calligraphic artwork using our system that runs on very modest hardware. In comparison with other existing virtual brush systems[2—10], our system is better in terms of naturalness of user control, system response time and expressiveness for painting and calligraphy.

1 Related work

The starting point—Strassmann proposed a simple brush model in 1986[2]. In his model, there are four working elements being simulated: *Brush*, *Stroke*, *Dip* and *Paper*. A *Brush* is composed of a number of *bristles;* a *Stroke* refers to the *trajectory* of a painting brush; *Dip* describes the initial condition of the *Brush*; and *Paper* is the virtual painting canvas. By implementing Strassmann's brush model, people can paint with many different strokes using a computer. Strassmann's paper ushered in a whole series of subsequent work on virtual brushes. In 1994, Hsu et al.[3] proposed an interactive painting method using skeletal strokes. Later, they successfully applied this technique to building a commercial painting system. Their painting system, however, is somewhat unnatural for the users when compared to the real brush. In 1997, Schlechtweg et al.[4] proposed 3-*D Linestyles*. *Linestyles* is a parametric line model. The model carries with it path, style, light beam, depth and possibly other additional 3-D information. Using this model, people can draw strokes with different widths, brightnesses and degrees of saturation. Compared with Hsu's painting system, Schlechtweg's 3-*D Linestyles* is a step forward due to its improved artistic expressiveness. Unfortunately, the texture variation in the 3-*D Linestyles* model is not rich enough to support realistic water-color or oriental painting. In 1999, Lee[5] proposed a virtual brush model based on elastic analysis. Lee created a fine orchid painting in traditional oriental painting style using his system. However, Lee's system does not provide a sufficiently efficient support for the splitting of the brush tip bundle. In 2000, Wong et al.[6] proposed another virtual brush model for writing Chinese characters. The main working unit in their system is a cone placed upside-down. Ink is deposited onto the intersection area between the cone and virtual paper. This model is developed for creating black and white calligraphy only. The variation of the brush tip bundle's geometry in their model is limited, which forbids the system to realistically simulate all the possible deformations during calligraphy. In 2001, Baxter et al.[7] studied the modeling of oil paintbrush. However, their paintbrushes are usually made of hard hair fibers. In comparison, oriental paintbrushes are usually made of soft hair. It is well known that the difficulty of simulating the behavior of soft objects is orders of magnitude harder than simulating hard, rigid objects. In 2002, Chu et al.[8] simulated the dynamics of an oriental brush by modeling a running brush as a spring system. They use energy minimization of a spring system to simulate the small deformation of the brush tip bundle. Unfortunately, large scale deformation is beyond their energy minimization approach. And lack of geometry level support for brush head forking is another drawback of their work, which seriously limits the expressiveness of their painting system. Greene[9] simulated the brush writing effect using an optical component following a pure hardware approach.

All in all, we find that hardware solutions are costly while currently available software solutions still cannot perfectly meet the demand of realistic simulation of a physical paintbrush for digital painting and calligraphy.

Along another line, the problem of parameterizing existent painting and calligraphy artwork has also aroused much research interest. Lee[10] simulated the ink diffusion phenomenon, which is an important aspect in digital painting and calligraphy. Shao et al.[14] proposed a method to represent the contour of Chinese characters using *Bezier* curves and straight lines. Shamir et al.[15] compressed the representation of Chinese characters using a parameterization model.

In this paper, we propose a novel algorithmic simulation framework for a new virtual hairy brush model. We use solid modeling techniques and focus exclusively on oriental painting and calligraphy. Our simulation framework encompasses brush geometry modeling, brush dynamics, and ink diffusion simulation. As regards human computer interaction when using our virtual hairy brush, end users are only expected to manipulate the six degrees of freedoms of the virtual hairy brush. The rest of the simulation work will be automatically carried out by the system. Such a user interaction pattern is a close resemblance to that of traditional painting using a real physical paintbrush. It represents a sharp improvement over traditional non-brush based graphics software, e.g. Adobe Photoshop, with which users have to edit control points or image properties in order to do painting digitally. Interactivity and artistic expressiveness are two other distinctive features of our new virtual brush system.

This paper is organized as follows: Section 2 introduces our solid model based virtual hairy brush. Section 3 discusses the simulation algorithm for digital painting and calligraphy using our virtual hairy brush. Section 4 discusses virtual hairy brush customization through brush quality parameter optimization. Section 5 presents some experiment results and possible future work.

2 A solid model based virtual hairy brush

To model the geometry of a paintbrush realistically, we introduce the concept of *Writing Primitive* (*WP*). With this concept, the complete geometry of a virtual hairy brush head can be represented by a collection of writing primitives. And the painting and calligraphy effects achievable by our virtual hairy brush are essentially the cumulative effects of the instant painting and calligraphy effect arising from the interaction between virtual paper and all the writing primitives in the paintbrush being modeled. In this section, we will discuss the definition of writing primitive, followed by algorithms to simulate the writing primitives' dynamics and the ink diffusion on the fly.

2.1 Writing primitive (*WP*)

By our definition, a writing primitive represents a bundle of brush hair threads, which is the minimal simulation granularity of our system. We define the geometry model of a writing primitive as a *NURBS* surface established through *the general sweeping operation* in solid modeling, as shown in Fig. 1. All the *WPs* in a virtual hairy brush function independently. That is, each *WP* dynamically adjusts its modeling parameters according to the external forces it receives and then deposits ink marks onto the virtual paper in-

dependently. The final brush strokes created are the cumulative effect of instantaneous ink marks, or more exactly, the union of all the instantaneous ink marks. Unlike traditional methods which simulate the behavior of each brush hair individually, our virtual brush model relies on the concept of writing primitives to simulate the macro behaviors of a bundle of hair threads. The main advantage is the possibility of simulating realistic brush dynamics in high realism while avoiding a massive, redundant simulation of a great many hair threads individually. This design improves the interactivity of our overall system significantly.

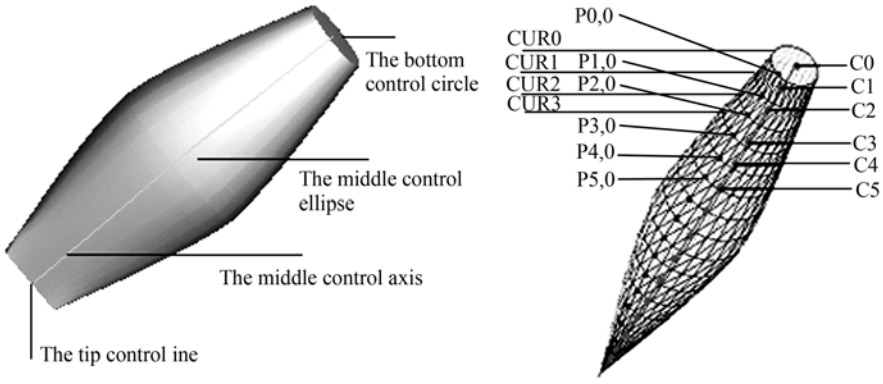

Fig. 1. Four attributes of a writing primitive (left) and the wire-frame structure of a writing primitive (right).

2.2 Modeling attributes of a writing primitive

A writing primitive carries four modeling attributes*: middle control axis* (*MCA*), *bottom control circle* (*BCC*), *middle control ellipse* (*MCE*), and *tip control line* (*TCL*). By our observation, the deformation of *BCC* is very trivial in almost all the brush painting and calligraphy processes. Therefore we safely assume that *BCC* always remains constant. Before a writing primitive experiences any deformation (i.e. when a writing primitive is at its initial status), *MCA, MCE* and *TCL* are degenerated into a straight line, a circle, and a point respectively. In the digital painting and calligraphy process, these three features will be adjusted accordingly to reflect the deformation of a running writing primitive. Meanwhile, parameters related to ink distribution will also be adjusted dynamically to simulate the diffusion of ink on the writing primitive. When the deformation of a writing primitive becomes so severe that its inner stress exceeds a certain threshold, the primitive will split into several new writing primitives. In this way, we can simulate the splitting of a brush head bundle during digital painting and calligraphy. We will discuss the three variable modeling attributes (*MCA, MCE* and *TCL*) one by one in the following.

(i) Middle control axis (*MCA*). The middle control axis is established through interpolation based on a few user input raw shape control points. In addition to using this

control axis as the skeleton of a writing primitive, we also record the virtual ink's local color and wetness in each of the control points on *MCA*. All the discrete points on *MCA* form a *control point sequence* (*PS*). During virtual painting and calligraphy, if the *current active point* (*CAP*) of a writing primitive, which is the current intersection point between *MCA* and the virtual paper plane, is not in *PS*, we will insert the newly generated intersection point into *PS*. The local ink color and wetness for the newly inserted point are derived through linear interpolation on the corresponding information carried by *CAP*'s two adjacent control points in *PS*. We also estimate *MCA*'s historical deformation degree as the cumulative displacement of all the control points in *PS* with respect to the center of the writing primitive's bottom control circle.

(ii) Middle control ellipse (*MCE*). The main features of a middle control ellipse are its major axis and the control ellipse's location parameter on its corresponding middle control axis. The orientation of the major axis of *MCE* is used to represent the asymmetric geometry of a writing primitive after its deformation due to the friction between the primitive and the virtual paper. It is easy to observe that if a writing primitive does not experience any splitting, the total number of its hair threads is constant. Based on this physical property, we assume that the area of *MCE* will remain constant during the writing primitive's deformation process as long as the primitive does not split.

(iii) Tip control line (*TCL*). Two major modeling features of the tip control line are its length and orientation. When a writing primitive is at its initial free state, its tip control line is degenerated into a point. When the geometry of the writing primitive gets deformed due to external forces exerted on it, the tip control line would also gradually be stretched into a real line with a certain length and orientation. Like the use of the orientation information of the middle control ellipse, the orientation of the tip control line is also used to represent the anisotropy of the geometry of a writing primitive after its deformation. Factors affecting the tip control line's deformation include the displacement of all the control points on the tip control line, the writing primitive's acceleration, average wetness, the stiffness of the brush hair and the current length of the tip control line.

2.3 Simulation of deformation and splitting of writing primitives

During digital painting and calligraphy, pressing a writing primitive against the virtual paper will deform its geometry and develop an inner stress. When the deformation exceeds a certain maximum tolerance threshold, the writing primitive will automatically split into several sub-primitives. Both the deforming and the splitting of the writing primitive are driven by the primitive's inner stress, which is estimated by the stiffness/elasticity of the brush hair, the number of hair threads in the writing primitive, the wetness, velocity, surface smoothness of the virtual paper and the historical deformation of the writing primitive. We discuss the details behind the dynamics simulation of a writing primitive now.

(i) Estimation of the inner stress of a writing primitive. According to the studies in materials and fluid dynamics, the following simplified equation for estimating the cur-

rent inner stress developed inside a writing primitive can be used:

$$\gamma = sm \times e \times ||vec|| \times \overline{wet}^{-1} \times vol \times his ,$$

where γ is the estimated inner stress of the writing primitive, *sm* is the surface smoothness factor of the virtual paper, *e* is an elastic factor of the brush hair, *vec* is the velocity vector of a moving writing primitive, $\overline{wet}$ is the average wetness of the writing primitive, *vol* is the volume of the part of writing primitive currently under the virtual paper, and *his* is a factor indicating the historical deformation of the writing primitive. Here *his* is estimated according to the variations of the writing primitive's three variable modeling attributes (middle control axis, middle control ellipse and tip control line) with respect to their respective initial states.

(ii) Deformation of a middle control axis. During digital painting and calligraphy, if the end user presses the virtual brush against the virtual paper by a displacement of *D*, all the points on the middle control axis staying above the virtual paper will automatically have a displacement of *D*. For those points under the virtual paper, they will go through a displacement of *D*-*dis*, where *dis* is a deformation extent determined by γ. We introduce this step to simulate the geometric deformation of the writing primitive due to the friction between the primitive and the virtual paper.

A deformed writing primitive will also get its deformation recovered to a certain extent when the writing primitive is lifted. If a user lifts the pen by a distance of *S*, all the previously deformed points on the middle control axis will have a vertical displacement of the amount, which is also determined by γ, to recover their previous deformation. Such a deformation recovery process is mainly triggered due to the relief of the inner stress of a writing primitive when it is being lifted.

(iii) Deformation of a middle control ellipse. Driven by the inner stress of the writing primitive, the major axis of the middle control ellipse will have a rotation by the angle of *rot*. Meanwhile, the length of the major axis will also extend to be *inc* times its initial value:

$$\begin{cases} rot = re \times \gamma \times (vec \cdot eori), \\ inc = ie \times \gamma \times ||vec \times eori|| . \end{cases}$$

Here, *re* and *ie* are the rotation and prolongation factors of the writing primitive respectively. *vec* is the velocity of the writing primitive, and *eori* is the unit orientation vector of the middle control ellipse. As discussed earlier in this paper, the total number of hair threads inside a writing primitive always remains constant, assuming that there is no brush splitting. Therefore during the major axis adjustment process, our system also needs to automatically update the length of the minor axis of the middle control ellipse to satisfy the area conservation assumption.

(iv) Deformation of a tip control line. Like the middle control ellipse, given the in-

ner stress of a writing primitive, the writing primitive's tip control line will also get rotated by the angle of *rot* and has its length extended to be *inc* times its initial value. The exact values of *rot* and *inc* are derived following the equation in the previous paragraph.

(v) Split of a writing primitive. Currently, we implement a simple strategy to simulate the splitting of writing primitives, which becomes a key feature of our realistic virtual brush simulation. The ideas behind are as follows. Given an upper bound *tre* of the maximum tolerable inner stress of a writing primitive, if the current inner stress γ of the writing primitive exceeds this upper bound, the current writing primitive will split into $k = \lfloor \gamma / tre \rfloor$ sub-entities, where each sub-entity becomes a new writing primitive. Meanwhile, the number of hair threads and the length of the tip control line of each of the newly generated writing primitive will both be reduced to be $1/k$ times their original values. For the lengths of the middle control ellipse's major and minor axes, they will be reduced to be $1/\sqrt{k}$ of their respective initial values. The rationale behind derives from two assumptions: 1) During the brush split, the number of all the hair threads of all the writing primitives inside the virtual hair brush should be conservative. 2) The volume of a writing primitive is proportional to the number of hair threads it contains. Note that in 1-D space, the volume of an object should be read as the object's length, and in 2-D space, the volume should be read as the object's area. Apart from all the above mentioned updates on modeling the attributes of the writing primitive, all the other aspects of the parametric model of the writing primitive keeps constant during the split operation. Fig. 2 shows some virtual brushes with heavily split writing primitives.

Fig. 2. Virtual brushes with heavily split writing primitives.

From the above discussions, readers may already notice that some parameters, e.g. *sm*, *e*, *re*, *ie*, *tre*, will affect the behavior of our virtual hairy brush, which will likely lead to the generation of different digital painting results with even the same set of user input. We call those parameters the *quality parameters of our virtual hairy brush*. We will give a more detailed discussion on the functionalities of these quality parameters in Section 4.

2.4 Simulation of ink diffusion on a writing primitive

For simplicity, in the current virtual hairy brush model, we assume the color of ink on a writing primitive will remain constant during the digital painting and calligraphy process. Such a color is determined when initially the brush is dipped into the ink bottle.

The only changing aspect of virtual ink is its wetness. And such a change in wetness is subject to the displacement and acceleration of the writing primitive, the absorption quality of the virtual paper, and the paper's surface smoothness. In our virtual brush system, we also provide a graphical user interface to let the end users interactively control the variation of ink information by directly specifying the ink information for a writing primitive during the digital painting and calligraphy process.

3 Digital painting and calligraphy using a virtual hairy brush

3.1 Interactive user control of a virtual hairy brush

Since our virtual hairy brush system is intended for digital artwork creation, providing a convenient and natural-to-use human computer interface is very important. Our current system supports two methods for user input. One is through the use of a WACOM tablet pen[16], which can sample the various degrees of freedoms of the virtual brush directly. Another method is through a combined mouse keyboard input strategy. In this method, we sample the placement of the virtual brush along the X, Y, and Z dimensions by moving the mouse horizontally or vertically and scrolling its middle wheel. A few combinations of key presses are chosen for inputting the remaining three degrees of freedoms of the virtual hairy brush. This mouse keyboard hybrid input strategy works much better and less clumsily than people would imagine. The reason is from observing that statistically 80% of operations on the virtual hairy brush are on the first three degrees of freedoms, which favors the mouse keyboard input strategy.

Among the two input methods, the first one offers a comfortable human computer interaction, which is suitable for professional artists for real painting tasks, while the second approach offers an inexpensive and acceptable way of human computer interaction. Without investing an expensive tablet pen, each ordinary home PC customer can play with our system to do rough drawing for fun with literally no cost. It is also useful to note that a better input device than WACOM tablet pen is the 3-D mouse [17], which can sample the movement of a virtual brush up to a total of 12 degrees of freedoms.

In order to further improve the usability of our virtual hairy brush, making the interaction with end users resemble more painting with a physical brush, we introduced a light inertia simulation procedure. In our prototype system, we estimate a brush velocity calibrator to account for the inertia of our virtual hairy brush. Such a calibrator is derived according to the sampled user input in the previous several simulation periods. Once the calibrator's value has been predicated, it is a linearly interpolated version of the current user input with a modification exerted by the calibrator that is actually passed on to all our software simulation algorithms, rather than the raw user input.

3.2 Intersection algorithm for a writing primitive against the virtual paper

During each time interval dT of the digital painting and calligraphy process, we intersect the writing primitive WP against the virtual paper plane PP to get a current intersection section (cross-section) M. We then fill the cross-section M on the virtual canvas ac-

cording to *WP*'s ink color and its wetness. By this area filling process, we can paint strokes on the virtual canvas. It is obvious that speeding up the above cross-section derivation calculation is critical to the overall performance of our system. We take note of the fact that only the part of the writing primitive that is in direct contact with the virtual paper will deposit ink onto the paper. By this observation, we designed the following Algorithm 1 to efficiently calculate the instantaneous ink mark having both geometry and color information for painting and calligraphy.

Algorithm 1. Intersection algorithm for a writing primitive against the virtual paper

Step 1. First, we normalize the control point sequence $J = \{C_0, \cdots, C_n\}$, which is from the middle control axis of the writing primitive *WP* currently under simulation,

while (there are two adjacent points C_i and C_{i-1} whose distance is farther than R)

{Insert the midpoint of C_i and C_{i-1} into the point sequence J.}

Here R is a resolution control factor which is related to the current screen space.

Step 2. Denote the normal plane of the middle control axis at $C_i (0 \le i \le n)$ as CP_i. Intersect CP_i with the writing primitive *WP* to get a closed curve CUR_i.

Step 3. Select an arbitrary radius of the writing primitive *WP* and sweep the radius along its middle control axis. During the sweeping, the radius will intersect with $CUR_0, \cdots, CUR_n$. We denote the resultant intersection points as $P_{0,0}, \cdots, P_{n,0}$. We then divide the tip control line uniformly into $\lceil N/2 \rceil$ segments with a resultant point sequence $CL = \{Q_0, \cdots, Q_{\lceil N/2 \rceil}\}$. Here N is another resolution control factor related to the current screen space and $\lceil \cdot \rceil$ is the rounding operator.

Step 4. Identify m, *s.t.* $P_{m,0}, P_{m+1,0}$ are on the opposite side of *PP*. ① If $0 \le m \le n-1$, we will uniformly divide CUR_m, CUR_{m+1} into N segments according to their arc length. The resultant point sequences are $CJ_m = \{P_{m,0}, \cdots, P_{m,N}\}$ and $CJ_{m+1} = \{P_{m+1,0}, \cdots, P_{m+1,N}\}$. We then use $P_{m,i}$ as the starting point and $P_{m+1,i}$ as the end point to make a line segment $(0 \le i \le N)$. Doing this will generate a sequence of line segments $LN_m = \{P_{m,0}P_{m+1,0}, \cdots, P_{m,N}P_{m+1,N}\}$. Again, we take $P_{m,i}$ as the starting point and choose $P_{m+1,i+1}$ as the end point to make a line segment $(0 \le i \le N-1)$, which will generate another sequence of line segments $LM_m = \{P_{m,0}P_{m+1,1}, \cdots, P_{m,N-1}P_{m+1,N}\}$. Finally we add $P_{m,N}P_{m+1,0}$ into LM_m. ② If $m = n$, we will use $P_{n,i}$ as the starting point and $Q_{\lceil i/2 \rceil}$ as the end point $(0 \le i \le N)$ to generate a sequence of line segments $LN_m = \{P_{n,0}Q_0, \cdots, P_{n,N}Q_{\lceil N/2 \rceil}\}$. We then choose $P_{n,i}$ as the starting point and $Q_{\lceil (i+1)2 \rceil}$ as the end point $(0 \le i \le N-1)$ to generate a sequence of line segments $LM_m = \{P_{n,0}Q_0, \cdots, P_{n,N-1}Q_{\lceil N/2 \rceil}\}$. Finally we add $P_{n,N}Q_0$ to LM_m. ③ If we cannot find any satisfying m, the algorithm terminates.

Step 5. For $\forall j$ ($0 \leq j \leq N$), if $P_{m,j}$, $P_{m+1,j}$ are on the opposite side of PP, the algorithm goes to the next step. Otherwise, without loss of generality, we assume $t = \max\{e | \forall k\ (0 \leq k \leq e), R_{m,k}, P_{m+1,k}$ are on the opsite side of $PP\}$ We then uniformly divide CUR_{m+1}, CUR_{m-1} into N pieces to get two point sequences CJ_{m+1}, CJ_{m-1}. In the following, we will only consider the spatial relationship of $P_{m+1,t+1}$, $P_{m-1,t+1}$ and $P_{m,t}$ with respect to PP. If there is one point among $P_{m+1,t+1}$, $P_{m-1,t+1}$ which is on the opposite side of $P_{m,t}$ with respect to PP, without loss of generality we can assume this point to be $P_{m-1,t+1}$. And then we set $m = m-1$, and make the algorithm go back to the beginning of Step 5. Otherwise, $P_{m+1,t+1}$, $P_{m-1,t+1}$ and $P_{m,t}$ must be all on the same side of PP. In that case, without loss of generality, we assume $P_{m+1,t+1}$ is closer to PP than $P_{m-1,t+1}$. We can then uniformly divide CUR_{m+2} into N segments according to its arc length. The resultant point sequence is CJ_{m+2}. And then we will consider the case of $P_{m+2,t+1}$. The same analogy goes on until we can find $l = \min\{w | P_{w,t+1}, P_{m,t}$ are on the opposite side of $PP\}$. After finding such an l, we will set $m = l$ and go back to the beginning of Step 5. Readers may find that for different js in different $P_{m,j}$s, their corresponding ms are not always the same; so in the strictest form, we should denote $P_{m,j}$, $P_{m+1,j}$ as $P_{m(j),j}$, $P_{m(j)+1,j}$. However, for ease of reading, we use the abbreviated notation.

Step 6. Now we have two point sequences $PO_1 = \{PO_{1,1}, \cdots, PO_{1,n_1}\}$ and $PO_2 = \{PO_{2,1}, \cdots, PO_{2,n_2}\}$. Any two points coming from the same point sequence must be on the same side of PP; while points coming from the different point sequences must be on the opposite sides of PP. And any point $PO_{g,h}$ in either of the two point sequences must be among the point sequence CJ_i of a certain curve CUR_i. That is, $PO_{g,h} \in CJ_0 \cup CJ_1 \cup \cdots \cup CJ_n$ ($g = 1, 2$; $h = 1, 2, \cdots, n_g$}. For each point $PO_{g,h}$, we need to find its corresponding point sequence CJ_m and its corresponding closed curve CUR_m. We assume that position is $P_{m,j}$. If the color of the ink is distributed along a certain vector direction, we denote the distance vector spanned by $P_{m,j}$ and the control point on the middle control axis C_m, which is in correspondence to CUR_m, as $P_{m,j}C_m$, and the color variation vector carried by C_m as D_m. Then the color of $P_{m,j}$ can be evaluated by the product of $P_{m,j}C_m$ with D_m.

$$P_{m,j}.\text{color} = (\langle P_{m,j}C_m \cdot D_m \rangle + 1) \times C_m.\text{color}, \tag{1}$$

here, $P_{m,j}.$ color and $C_m.$color are the colors on $P_{m,j}$ and C_m respectively.

If the color of ink is distributed radically, we denote the spatial distance between the current node $P_{m,j}$ and CUR_m's corresponding point C_m as $||P_{m,j} - C_m||$, and denote the color variation scalar carried by C_m as d_m. Then the color of $P_{m,j}$ can be determined according to $||P_{m,j} - C_m||$ and d_m.

$$P_{m,j}.\text{color} = (||P_{m,j} - C_m|| \times d_m + 1) \times C_m.\text{color}. \tag{2}$$

3.3 Interactive painting and calligraphy using a virtual hairy brush

After intersecting the writing primitive with the virtual paper using Algorithm 1, we can now command our virtual hairy brush as an interactive tool for painting and calligraphy, using the following Algorithm 2. Despite what it is called, this algorithm is essentially an algorithm for interactive painting and calligraphy using writing primitives. This is because the major design idea behind our virtual hairy brush model is to use a collection of independently functioning writing primitives to simulate the overall function of a virtual hairy brush.

Algorithm 2. Interactive painting and calligraphy using a virtual hairy brush

Step 1. For the two point sequences obtained by Algorithm 1, i.e. $PO_1 = \{PO_{1,1},\cdots, PO_{1,n_1}\}$ and $PO_2 = \{PO_{2,1},\cdots, PO_{2,n_2}\}$, without loss of generality, we assume $n_1>n_2$. We then connect $PO_{1,i}$ and $PO_{2,j}$ to get LO_i, where $i = 1, \cdots, n_1$, $j = \lfloor i/n_1 \times n_2 \rfloor$ and $\lfloor x \rfloor$ is the minimum integer no less than x. Each of the resultant line segment LO_i ($i = 1, \cdots, n_1$) is intersected with the virtual paper plane PP to get an intersection point S_i. Similarly we also connect $PO_{1,i}$ and $PO_{2,j+1}$ to get LO'_i and intersect the resultant line segment with the virtual paper plane to generate the intersection point S'_i. All the intersection points, i.e. S_i ($i = 1, \cdots, n_1$) and S'_i ($i = 1, \cdots, n_1$), are ordered to form a current ink mark polygon $Poly = \{V_1, \cdots, V_{2\times n_1}\}$. We use this polygon to approximate the irregular cross-section between the writing primitive and the virtual paper plane. Please refer to Fig. 3 for illustration of the wire-frame result and a current ink mark polygon.

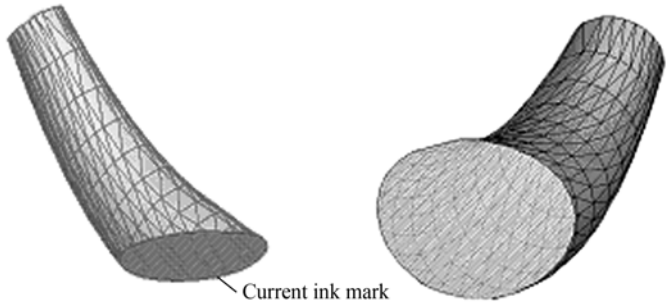

Fig. 3. Wire-frame model of a writing primitive and its current ink mark polygon.

Step 2. For any vertex V_i of the polygon $Poly$, without loss of generality, we assume V_i is obtained by intersecting PP with a line segment whose two end points are $PO_{1,s}$ and $PO_{2,t}$. We can then calculate the ink information $V_i.pro$ on V_i simply using linearly interpolation.

$$V_i.pro = \frac{||PO_{1,s} - V_i|| \times PO_{2,t}.pro + ||PO_{2,t} - V_i|| \times PO_{1,s}.pro}{||PO_{1,s} - PO_{2,t}||}. \quad (3)$$

Here $V_i.pro$ is a vector which contains both the local color information and the wetness around V_i.

Step 3. We then build a mesh over *Poly* using all the horizontal and vertical lines, which passes through at least one of the polygon's vertices V_i. The generated mesh is condensed to a resolution *RR*, which is a distance determined by the current screen space. Denote the *x*, *y* coordinates of V_i as $V_{i\cdot x}$ and $V_{i\cdot y}$. For any *n*, if $||V_{n\cdot x}-V_{n-1\cdot x}||>RR$, we will calculate a new intersection point between the line X = $(V_{n\cdot x}+V_{n-1\cdot x})/2$ and *Poly*. The resultant intersection point is inserted between V_n and V_{n-1}. We repeat the above operations until no more new point needs to be inserted. Similar processing is carried out for the case of $||V_{n\cdot y}-V_{n-1\cdot y}||>RR$.

Step 4. For any grid point *NP* lying on the generated mesh over *Poly*, without loss of generality, we assume this point is obtained by intersecting the line segment V_aV_b with V_cV_d. And then we can compute the ink information $NP.pro$ on the grid point *NP* as

$$NP.pro = \frac{1}{2}\times\left(\frac{||V_a-NP||\times V_b.pro+||V_b-NP||\times V_a.pro}{||V_a-V_b||} + \frac{||V_c-NP||\times V_d.pro+||V_d-NP||\times V_c.pro}{||V_c-V_d||}\right). \tag{4}$$

Step 5. We use $Area_l$ to denote an arbitrary region inside the polygon *Poly*. According to the above mesh establishment process, $Area_l$ is either a triangle or a rectangle. So the vertices of $Area_l$ can be denoted as $P_{l,1}P_{l,2}P_{l,3}$ or $P_{l,1}P_{l,2}P_{l,3}$ $P_{l,4}$. Then for any arbitrary point (i, j) inside $Area_l$, its color information can be computed by applying a bi-linear interpolation according to the color information recorded on all the vertices of this region $Area_l$.

Step 6. Denote the average wetness of the writing primitive WP_k at moment *t* as $wet_{k,t}$, its average inner pressure as $pre_{k,t}$, the number of hair threads that WP_k contains as num_k, the average acceleration of WP_k as $dS_{k,t}$, the ink absorption rate of the virtual paper for the virtual ink as *absor*, the wetness on the virtual paper around point $P_{i,j}$ as $Pwet_{t,i,j}$. And now we can use the following equation to estimate the probability, $prob_{t,i,j}$, for choosing the point $P_{i,j}$ for ink deposition at moment *t*.

$$prob_{t,i,j} = (\beta_1\times wet_{k,t}+\beta_2\times Pwet_{t,i,j})\times absor\times pre_{k,t}\times(dS_{k,t})^{-1}\times num_k. \tag{5}$$

If $P_{i,j}$ is selected for ink deposition, its wetness will increase by the amount of $dH_{t,i,j}$.

$$dH_{t,i,j} = \beta_3\times(wet_{k,t}+Pwet_{t,i,j}). \tag{6}$$

In the above equations, β_1, β_2, β_3 are three quality parameters of the virtual hairy brush (please refer to section 4 for discussion on the quality parameters of the virtual hairy brush). After the ink deposition, if the wetness of this point exceeds an upper

boundary *top* , the ink will diffuse into its neighboring locations. Assume that ink is only diffused in eight directions, then each of $P_{i,j}$'s eight neighboring points, $P_{m,n}$, whose current wetness is $Pwet_{m,n}$, will have an increase of wetness by the amount $\Delta wet_{t,m,n}$.

$$\Delta wet_{t,m,n} = random() \times (Pwet_{t,i,j} + dH_{t,i,j} - top) \times facw_{t,i,j}^{-1} \times (1 - Pwet_{t,m,n} / top). \quad (7)$$

Here $m = i-1, i, i+1$; $n = j-1, j, j+1$; $(m, n) \neq (i, j)$; $facw_{t,i,j}$ is a banning factor for ink diffusion at the point $P_{i,j}$, i. e.

$$\begin{aligned} facw_{t,i,j} = 8 - (&Pwet_{t,i-1,j-1} + Pwet_{t,i-1,j+1} + Pwet_{t,i-1,j} + Pwet_{t,i+1,j+1} \\ &+ Pwet_{t,i+1,j-1} + Pwet_{t,i+1,j} + Pwet_{t,i,j-1} + Pwet_{t,i,j+1}) / top. \end{aligned} \quad (8)$$

random() generates a random number in the range of [0,8] with a mathematical expectation being 1. In addition, there is another so called "drying factor of the virtual paper" *Dry*. During each simulation step, the wetness of all the points on the virtual paper will decrease by the amount of *Dry* until it reaches zero.

Step 7. To simulate the drying brush effect and running brush effect better, we further filter the points that are output from Step 6. The idea behind is to set up a mapping to a texture map for candidate filtering under the user's intentional control. If the grey level of the point being mapped is *gr* and the maximum grey level of this texture map is *gr*_max, then the possibility of selecting this point for ink deposition is $\frac{gr}{gr\_\max}$. By this means, we can effectively control the ink distribution through a pre-defined texture map. The computer may also randomly perturb the texture coordinates to be mapped. There is another optional token-bucket process for further improvement of the realism of dry brush effect and running brush effect:

Token-bucket process. First, a segmentation factor *fra* is generated according to the number of hair threads contained in the current writing primitive. Initially, the token number in the bucket is 0. If a point is selected for ink deposition, then the next following *fra* points will all be selected for ink deposition. Meanwhile, we set the token number to be *fra*−1. In the future, when there is a new point to be selected for ink deposition, the ink deposition process will no longer be carried out. Instead, the token number will be reduced by 1. Such an ink deposition selection disabling will continue until the time when the token number returns a 0 again when a new set of *fra* points can be selected for ink deposition.

The complete simulation framework for interactive digital painting and calligraphy using our virtual hairy brush model is shown in Fig. 4. The main points of such a framework have been explained in section 2 and section 3. Note that steps with * will only be executed when their corresponding preconditions are satisfied.

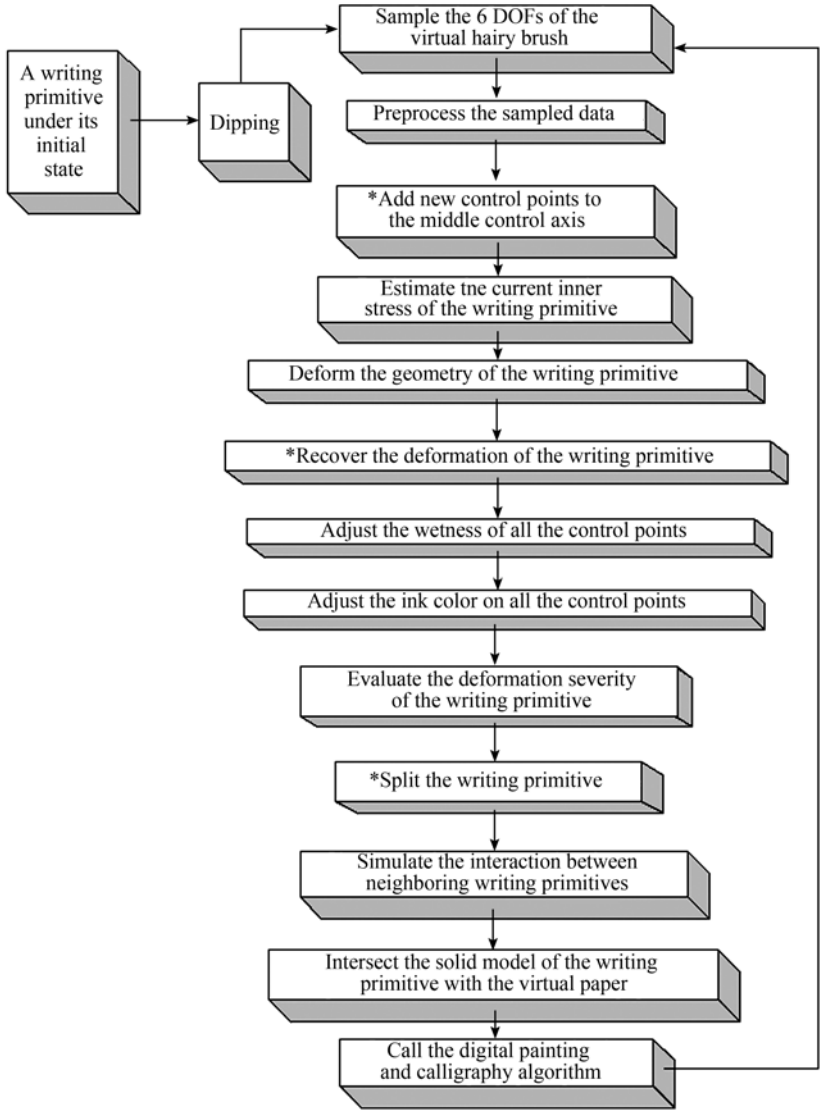

Fig. 4. Simulation framework for painting and calligraphy using a virtual hairy brush.

4 User customization of a virtual hairy brush

In artists' domains, the most commonly used hairy brushes are made up of soft hair. Different brushes embrace different qualities in the process of painting and calligraphy, in particular regarding their dynamics. For example, brushes with thicker hair tend to absorb more ink and their ink saturation capacity is large; brushes with long hair will experience more intensive deformation during painting and calligraphy. To capture the different qualities of real paintbrushes, we introduce many adjustable quality parameters in our virtual hairy brush model, as have been presented in section 2 and section 3. To

support user customization of brush qualities, we set up a configuration database for storing different sets of well-tuned quality parameters, one for a certain kind of brush. Users can choose their most preferred quality parameter configuration for their painting or calligraphy. In addition to the simple choose-and-use option, our system also provides an interactive interface for the user to directly adjust these virtual brush quality parameters. Moreover, our system is equipped with a powerful self-training module for customizing those quality parameters. The working principle of this quality parameter training module is as follows.

Our prototype system carries a certain set of typical strokes from sample paintings and calligraphy artworks. During the training process, a user is expected to imitate these sample strokes using our virtual brush system. Once the user finishes his imitation, we will derive an object function for the machine training module by defining the function to be the distance in RGB space between the imitated strokes and the given sample strokes to be imitated. Under the goal of minimization of this object function, a set of satisfying or most optimized quality parameters of our virtual hairy brush and virtual paper can be identified. The purpose of going through this additional brush quality parameter training process is to customize the most qualified virtual hairy brush and the virtual paper in accordance with an end user's personal preference.

5 Experiment results and discussions

Based on the algorithms proposed in the paper, we have developed a prototype system for digital painting and calligraphy. A snapshot of our running system is shown in Fig. 5. Fig. 6 shows two digital paintings interactively generated using our system. The breakdown of the time spent is as follows: 30% of the time for rendering the brush geometry model, 25% of the time for simulating pigment behavior, that is, the digital painting and calligraphy process in which paint is deposited onto the virtual paper, 20% of the time for brush dynamics, 15% of the time for the initial generation of the brush geometry model and the model update, and the remaining 10% of the time is spent on miscellaneous computations necessary for the simulation.

The solid model based virtual hairy brush described in this paper can simulate many artistic brush painting effects, e.g. dry brush effect and ink diffusion effect. The support for such artistic effects makes our novel virtual hairy brush very expressive for painting and calligraphy. If the rigidity or stiffness of the brush hair is suitably tuned, this system can also be used to produce calligraphy and painting effects achievable by hard hair paintbrushes.

The main advantages of our solid model based virtual hairy brush model and its associated simulation framework for interactive digital painting and calligraphy are as follows:

(1) By introducing the concept of writing primitives and using them as the minimum simulation primitives, much redundant or non-representative behavior of individ-

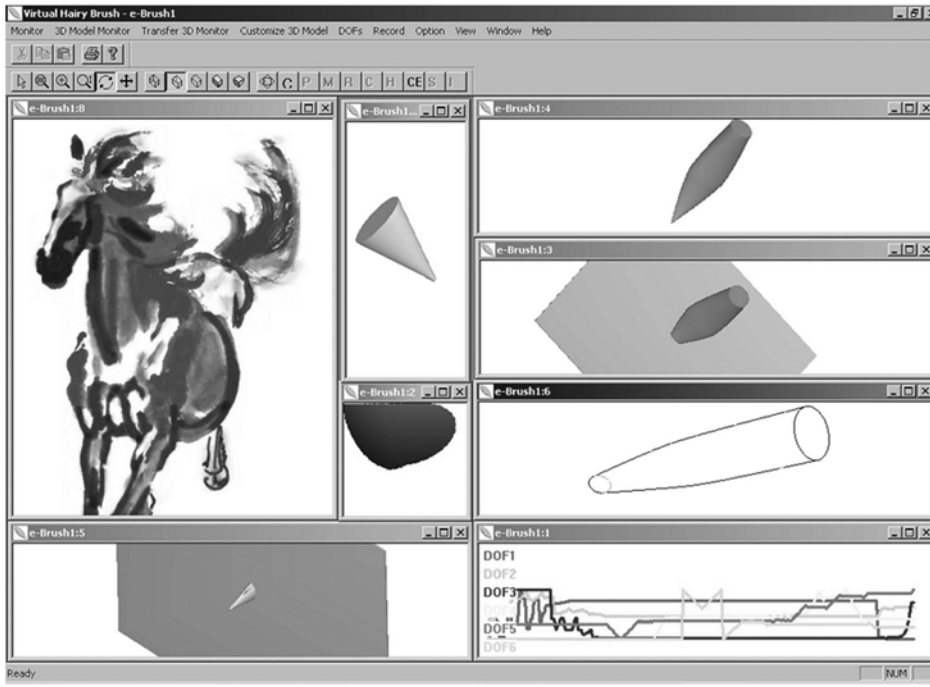

Fig. 5. A snapshot of our prototype system in action.

Fig. 6. A running horse (left) and parrots (right), both from using our system.

ual hair threads in the virtual brush can be eliminated from the simulation. Instead, some elaborate dynamics and ink diffusion simulation procedures are proposed to capture the macro, representative behavior of clustered brush hair. This can vastly speed up our

painting software, enabling it to have real-time responses with a very high degree of realism. When characters in Kai-font, Li-font and most cases in Xing-font are to be written using our virtual hairy brush, only one writing primitive is enough. To write in Kuang-cao-font, 50 writing primitives are more than enough.

(2) In our virtual hairy brush, writing primitives represented using NURBS surface replace the simple geometry model in the earlier brush models[2,5,6,8]. With this enhanced geometry modelling capability, the interaction area between the writing primitives and the virtual canvas, namely the instant ink mark area, can be of any arbitrary shape rather than having to be an ellipse as in existing approaches. Providing this sort of diversity in the instantaneous brush mark makes our simulation more realistic, the brush mark contour more natural, and the resultant paintings more expressive.

(3) Each control point on the middle control axis can carry its own ink information. Such a mechanism makes it possible to represent a fairly complicated ink distribution, including the variation of ink color and wetness, using only one writing primitive. In addition to the merit of compact representation, simulation of ink diffusion is also benefited in terms of representation capability and the efficiency accessing the data structure of the representation.

(4) The process of ink deposition onto the virtual paper is controlled with probability, which enriches many aesthetic brush painting effects.

(5) During calligraphy and painting, we additionally introduce and simulate an inertial term as a pre-processing step before the brush dynamics simulation for the virtual hairy brush. This improves the sense of naturalness when end users command our brush, allowing the virtual brush to move more fluently.

(6) Since our simulation framework is built on the parametric virtual hairy brush, an end user no longer has to manually specify a large number of brush geometry parameters and brush trajectory parameters for the process of painting and calligraphy. These laborious and tedious tasks are fulfilled automatically in our system by sampling the six degrees of freedoms of the virtual hairy brush and running all the simulation algorithms according to the sampled input. Relieving the end users of such burdens help the users protect and focus on their creativity during digital painting and calligraphy.

(7) Our system also has an interface to customize the quality of the virtual brush through adjusting its quality parameters, either manually or automatically.

(8) Everything inside our digital painting world is parameterized, including the virtual hairy brush model, the six degrees-of-freedom user input, and even the final brush strokes painted. Owing to this parametric nature of representation and simulation, it is no longer necessary to store large quantities of bitmaps. Memory space can be saved substantially. A parametric archiving of the complete digital painting and calligraphy proc-

ess is also possible, which can be used for many meaningful and interesting applications, e.g. animation of paintings based on their parametric representations and high level painting style analysis.

Future work includes developing a computer-aided education environment for on-line painting and calligraphy tutoring based on our current system. In such an environment, study materials are no longer confined to just the final artwork in paper form created by artists; they can also include the detailed and complete information on how the painter/writer manipulates his or her brush throughout the painting or writing process. Such a functionality is impossible to achieve using a traditional physical brush. Developing an intelligent painting and calligraphy generation system is also quite intriguing[18,19]. In the long run, integrating intelligent human-computer interaction techniques into a digital painting system should be a challenging direction which has many practical values.

Acknowledgements We are grateful to the fellows and experts of the Chinese Association of Scientists, the Chinese Academy of Sciences and the Chinese Academy of Engineering for their constructive suggestions and comments on this project in the past four years. The virtual hairy brush project reported in this paper won the First Prize and the "Edison Cup" in GE Fund "Edison Cup" National Collegial Technology Innovation Contest, administered by the International Education Committee (US), and the Outstanding Prize and "Challenge Cup" in the 8th "Challenge Cup" Great China National Collegial Academic, Scientific and Technological Research Project Contest, administered by the Chinese Ministry of Education and Chinese Association of Scientists. This work was supported by the National Natural Science Foundation of China (Grant No. 60402010) and the National "973 Plan" (Grant No. 2002CB312106).

References

1. Li Zehou, Trilogy of Aesthetics, History of Aesthetics (in Chinese), Vol. I, Hefei: Anhui Literature Press, 1999.
2. Strassmann, S., Hairy brush, in Proceedings of SIGGRAPH'86, Dallas, USA, New York: ACM Press, 1986, 225—232.
3. Hsu, S. C., Lee, I. H. H., Drawing and animation using skeletal strokes, in Proceedings of Siggragh'94, Orlando, USA, New York: ACM Press, 1994, 109—118.
4. Schlechtweg, S., Raab, A., Rendering line drawings for illustrative purpose, in Abstraction in Interactive Computational Visualization: Exploring Complex Information Space (eds. Strothotte, T., Wagener, H.), Berlin: Springer-Verlag, 1997.
5. Lee, J., Simulating oriental black-ink painting, IEEE Computer Graphics and Applications, 1999, 19(3): 74—81.
6. Wong, H. T. F., Ip, H. H. S., Virtual brush: a model-based synthesis of Chinese calligraphy, Computers & Graphics, 2000, 24(1): 99—113.
7. Baxter, B., Scheib, V., Lin, M. et al., DAB: interactive haptic painting with 3D virtual brushes, in Proceedings of SIGGRAPH'01, Los Angeles, USA, New York: ACM Press, 2001, 461—468.
8. Chu, N., Tai, C., An efficient brush model for physical-based 3D painting, in Proceedings of Pacific Graphics'02, Beijing: IEEE Press, 2002.
9. Greene, R., The drawing prism: a versatile graphic input device, in Proceedings of SIGGRAPH'85, San Francisco, USA, New York: ACM Press, 1985. 103—109.
10. Lee, J., Diffusion rendering of black ink paintings using new paper and ink models, Computers & Graphics, 2001, 25(2): 295—308.[DOI]

11. Xu, S, Tang, M., Lau, F. C. M. et al., A solid model based virtual hairy brush, in Proceedings of Eurographics'02, Saarbrucken, Germany, Oxford: Blackwell Publishers, 2002, 299—308.
12. Xu, S., Tang, M., Lau, F. C. M. et al., A solid model based virtual hairy brush, Technical Report, HKU-CSIS-TR-2002-04, Department of Computer Science, The University of Hong Kong, 2002.
13. Xu, S., Lau, F. C. M., Tang, F. et al., Advanced design for a realistic virtual brush, Proceedings of Eurographics'03, Granada, Spain, Oxford: Blackwell Publishers, 2003, 533—542.
14. Shao, L., Zhou, H., A new contour fill algorithm for outlined character image generation, Computers & Graphics, 1995, 19(4): 551—556.[DOI]
15. Shamir, A., Rappoport, A., Quality enhancements of digital outline fonts, Computers & Graphics, 1997, 21(6): 713—725.[DOI]
16. Wacom Technology Co., http://www.wacom.com/.
17. Bernd, F., John, P., Jurgen, W. et al., Cubic-mouse-based interaction in virtual environments, IEEE Computer Graphics & Applications, 2000, 20(4): 12—15.
18. Xu, S., Lau, F. C. M., Cheung, K. W. et al., Automatic generation of artistic Chinese calligraphy, in Proceedings of the Nineteenth National Conference on Artificial Intelligence and the Sixteenth Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI), San Jose, USA, AAAI Press/The MIT Press, 2004, 937—942.
19. Xu, S., Lau, F. C. M., Pan, Y., Automatic artistic calligraphy generation, Technical Report, HKU-CSIS-TR-2003-02, Department of Computer Science, The University of Hong Kong, 2003.