Shape blending of artistic brushstroke represented by disk B-spline curves*

Cheng Min[1,2] and Wang Guojin[1**]

(1. Department of Mathematics, Zhejiang University, Hangzhou 310027, China; 2. Department of Mathematics, Zhejiang University of Technology, Hangzhou 310032, China)

Accepted on July 11, 2007

Abstract This paper presents an algorithm for automatically generating in-between frames of two artistic brushstrokes. The basic idea of the algorithm is to represent the two key frames of artistic brushstrokes in disk B-spline curves, and then make blending of their geometric intrinsic variables. Given two key frames of artistic brushstrokes, the skeleton curves can be obtained by certain skeleton-based techniques. After disk B-spline representation of the key frames is generated, interpolation of the intrinsic variables of the initial and the target disk B-spline curves is carried out. Examples show that this method can efficiently create in-between frames of artistic brushstrokes.

Keywords: computer aided calligraphy, shape blending, brushstroke, disk B-spline curve, skeleton.

In artistic design and manufacturing, how to efficiently and effectively give the appropriate representation of a brushstroke is an exigent problem often occurring in computer generated calligraphic lettering, painting and animation. Many studies have been carried out in this area. Raster image stack can be used, but large storage space is required and it is not flexible for various manipulations. Another way of representation is to describe the outlines in parametric curves. Bézier or B-spline form is often used to represent these outlines with convenience and simplicity in practice[1—3]. Brushstrokes are always in the pixels concentrated form. If the thicknesses of the brushstroke vary little, the approach of representing their centerline as the datum line and modulating the thickness to describe brushstrokes can achieve acceptable result[4]. Non-uniform B-spline is a good choice to define centerline as a standardization in industry[5]. However, how to efficiently calculate its quasi-offset is still a problem. In addition, interval B-spline is also a selection in the representation of 2D region of brushstroke[6]. But when rotation transformation is needed in a coordinate system, interval splines are not easy to use. In recent years, a novel representation, named disk Bézier curve, has been used in error analysis of shape design and detection of contour line[7]. The basic idea is to represent the center curve in the Bézier curve form and give each point on the curve an outspread distance known as the radius. This kind of curves with variant "fatness" can be used to simulate brushstrokes. It can be obviously seen that if all the radii of the disks are selected to be the same, the obtained curve is just the well-known offset curve. Considering that many brushstrokes do not preserve high order continuity, in order to depict some complex shapes, in this paper, we choose disk B-spline curve (DBSC) instead of simple disk Bézier curve to represent arbitrary strokes. Examples show that this selection is advisable both in rendering quality and time consumption.

Now the problem lies in how to carry out shape blending. During the production process in the fields of mass media, advertising and animation, manual painting is both inefficient and cost consuming, which cannot satisfy the demand of clients. A large number of manual paintings will be needed to show a continuous series of actions. So pure handwork is really incompetent. If this problem can be solved, then much animation work can be carried out automatically, which can simultaneously save the manual work of artists. Here, automatic shape blending techniques are suggested to solve this problem. Shape blending

* Supported by the National Program on Key Basic Research Projects (Grant No. 2004CB719400), and National Natural Science Foundation of China (Grant Nos. 60673031 and 60333010)
** To whom correspondence should be addressed. E-mail: wanggj@zju.edu.cn

deals with the problem of how to efficiently determine the in-between smooth transitional frames between two given shapes, which are respectively referred to as the initial frame and the target frame. It is also named shape averaging, shape interpolation and morphing. Different blending methods lead to different results. Sometimes efficiency and effect are just on the two opposite sides. It is really difficult to guarantee one without sacrificing the other. So how shape blending can be efficiently carried out without severe distortion or diminishing is a problem worth studying. Previous methods for planar shape blending include geometry-based method[8] and quasi-physics-based methods[9,10]. Provided that the shape is represented as a polygon, Sederberg proposes a technique which interpolates the intrinsic parameters—angles and edge lengths of the polygons[8]. This is a simple and effective method for most polygon based blending problems. In Refs. [9, 10], Hu presents a novel technique for shape blending, which is based on strain field interpolation. It allows the shape blending of textured objects by firstly decomposing each object into an isomorphic triangular mesh, and then using strain field interpolation to control the deformation of one mesh into the other. Mapping the texture is straightforward once the polygons can be mapped, e.g. by interpolating values at corresponding barycentric coordinates. The method is robust, and good blending results can be achieved.

This paper deals with the problem of shape blending of artistic 2D brushstroke, which is represented by disk B-spline curves. Given two key frames of artistic brushstrokes, the skeleton curves can be firstly calculated by certain skeleton-based techniques. After the disk B-spline representation of the key frames is generated, interpolation of the intrinsic definitions of the initial and the target frames is carried out. The in-between frames are achieved by the intrinsic variables. This method is essentially a kind of geometry-based method. Compared with the quasi-physics-based methods in Refs. [9, 10], complex solution of equations can be avoided, which directly leads to the decrease of time consumption. But on the other hand, texture mapping is not covered in this paper. Examples show that our blending algorithm, which is based on intrinsic variables of disk B-spline curves, can be applied to computer aided artistic design. It can efficiently create in-between frames of artistic brushstrokes and avoid the problem of distortion and diminishing.

1 Generation of disk B-spline curve representation of artistic brushstroke

1.1 Disk B-spline curve

In order to introduce the definition of disk B-spline curve, some preliminary results for disk arithmetic are presented at first[7].

Let R be the set of real numbers and R^+ be the set of nonnegative real numbers. A real disk in a plane can be defined as the set of points

$$(\boldsymbol{P}_0)_{r_0} := (x_0, y_0)_{r_0} := \{(x, y) \in R^2 \mid (x - x_0)^2 + (y - y_0)^2 \leqslant r_0^2\} \quad (1)$$

where $\boldsymbol{P}_0 = (x_0, y_0) \in R^2$ is the center point and $r_0 \in R^+$ is the radius.

Additive operation between disks and multiplication operation of a real number by a disk are defined respectively as follows. For any two disks, define $(x_i, y_i)_{r_i}$, $i = 1, 2$, define

$$(x_1, y_1)_{r_1} + (x_2, y_2)_{r_2} = (x_1 + x_2, y_1 + y_2)_{r_1 + r_2} \quad (2)$$

$$a(x_i, y_i)_{r_i} = (ax_i, ay_i)_{|a| r_i}, \quad \text{for} \quad a \in R \quad (3)$$

In addition, for the linear combination of n disks, define

$$\sum_{i=1}^{n} a_i (x_i, y_i)_{r_i} = \left(\sum_{i=1}^{n} a_i x_i, \sum_{i=1}^{n} a_i y_i \right)_{\sum_{i=1}^{n} |a_i| r_i}, \quad \text{for} \quad a_i \in R \quad (4)$$

Now it comes to the definition of disk B-spline curves.

Given a knot vector $\boldsymbol{T} = \{t_i\}_{i=-\infty}^{\infty}$, $t_j \leqslant t_{j+1}$, let $N_{i,k}(t)$ be the B-spline basis functions of order k on $\boldsymbol{T}$. Let $(\boldsymbol{P}_i)_{r_i} = (x_i, y_i)_{r_i}$, $i = 0, 1, \cdots, n$ be the $n + 1$ disks given as control disks. A disk B-spline curve can be defined as

$$(\boldsymbol{P})(t) := \sum_{i=0}^{n} (\boldsymbol{P}_i)_{r_i} N_{i,k}(t) \quad (5)$$

Obviously, a disk B-spline curve is determined by two parts. One is a center curve, and the other is the radius function. Thus a disk B-spline curve can equally be written as

$$(\boldsymbol{P})(t) := (x(t), y(t))_{r(t)}, \quad (6)$$

where $\boldsymbol{C}(t) := (x(t), y(t)) = \left(\sum_{i=0}^{n} x_i N_{i,k}(t), \sum_{i=0}^{n} y_i N_{i,k}(t) \right)$ is the center curve in B-spline form, and $r(t) = \sum_{i=0}^{n} r_i N_{i,k}(t)$ is a B-spline scalar function acting as the corresponding radius function. A disk B-spline curve can be regarded as the area swept by the moving circles with the center curve $\boldsymbol{C}(t)$ and the radius function $r(t)$. Two examples of cubic disk B-spline curves are illustrated in Fig. 1.

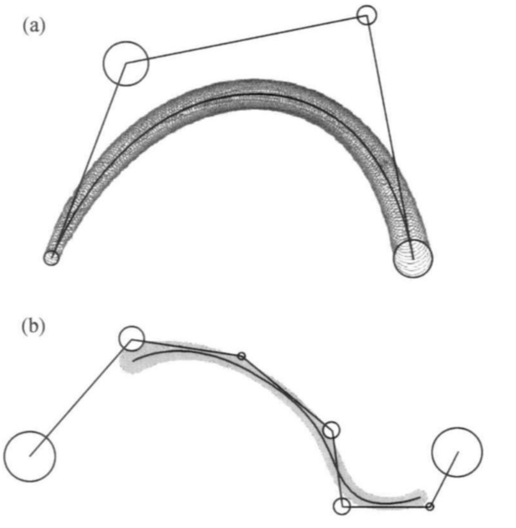

Fig. 1. Cubic disk B-spline curves. (a) A cubic disk B-spline curve with four control disks (Repeated knots are selected on the two ends to preserve endpoints interpolation property. The center curve is rendered in solid line, and the disk B-spline curve is rendered in moving disks); (b) a cubic disk B-spline curve with seven control disks (The center curve is rendered in solid line, and the disk B-spline curve which depicts a 2D brushstroke is rendered in grey).

1.2 Disk B-spline representation of brushstrokes

Because B-spline curve has various specialties, it is one of the most commonly used mathematical models in the design and modeling systems. For example, its shape and its continuity order are both locally adjustable. Therefore selecting B-spline curve to represent the center curve of a given brushstroke is advisable. In addition, a radius function is given to describe the "fatness" of the brushstroke flexibly. As shown in Fig. 1(b), disk B-spline curve can be used to depict a 2D brushstroke. Now the problem lies in how to efficiently give the disk B-spline representation of an arbitrarily given brushstroke. Two steps are suggested in this paper to solve this problem. In the first step, the skeleton curve is obtained from the given brushstroke and being used as the center curve. Here, how to efficiently get the skeleton curve is one problem that needs to be considered. Brushstrokes have their special topological properties. So getting skeleton from brushstrokes is rather easier than that from random scattered data. In this paper, the existing skeleton generation method[11] is suggested. Based on the skeleton curve, its approximating B-spline curve can be quickly achieved by interpolating discrete sampling data. In the second step, radius of each control point is suggested to be calculated through data fitting. Here, normal distance of each control point is approximately selected as the radius.

The skeleton curve will be approximated better when more data are used in interpolation, especially in those segments where the curvature function is critically oscillating. Thus, dynamical sampling can be introduced. Most brushstrokes can be well depicted by disk B-spline curves through the above skeleton-based techniques. Two examples are given in Figs. 2 and 3. Fig. 2 gives an example of Chinese calligraphy, and Fig. 3 is a symbol of athletics for Beijing Olympic 2008. Both of them are simulated by disk B-spline curves.

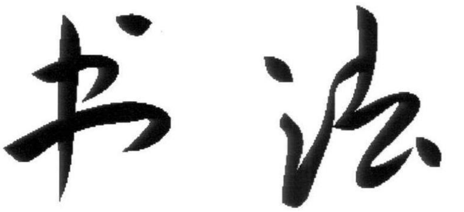

Fig. 2. Chinese calligraphy depicted by disk B-spline curves.

Fig. 3. Symbol of athletics for Beijing Olympic 2008 depicted by disk B-spline curves.

2 Shape blending of two key frames represented by disk B-spline curve

2.1 Intrinsic definitions of 2D polygons and planar disk B-spline curves

Rectangular Cartesian coordinate is often selected in rendering because of intuitive thinking and simplicity in parametric representation. After a polygon is defined, the Cartesian coordinates of the vertices are given instead of those geometric information such as the lengths of the edges and the angles. Each vertex is located independent of any other vertices and depends only on its coordinates. As a result, it lacks the property of geometric invariance. In order to solve this problem, polar coordinates representation is selected in this paper, because it takes full considerations of the geometric information. They are referred to as the "intrinsic variables"[8]. A brief introduction is given as follows.

Let $\boldsymbol{P}_i$, $i = 0, 1, \cdots, n$ be the vertices of a 2D polygon Γ. Denote $\rho_i = \| \boldsymbol{P}_{i-1}\boldsymbol{P}_i \|$, $i = 1, 2, \cdots, n$, and θ_i as the directed angle from the vector $\boldsymbol{P}_{i-2}\boldsymbol{P}_{i-1}$ to the vector $\boldsymbol{P}_{i-1}\boldsymbol{P}_i$, $i = 2, 3, \cdots, n$. Here, anticlockwise angle θ_i is defined to be positive, while clockwise direction be negative. The set $\Omega = \{\{\rho_i\}_{i=1}^n, \{\theta_i\}_{i=2}^n\}$ is called the intrinsic set of the polygon Γ, and the values ρ_i and θ_i are referred to as the intrinsic variables (Fig. 4).

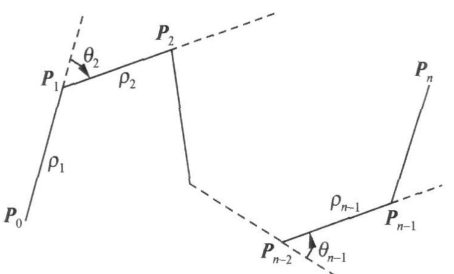

Fig. 4. The intrinsic variables of a 2D polygon.

A polygon can be uniquely determined by the coordinates of its vertices under rectangular Cartesian coordinate. If intrinsic variables are selected to describe a polygon, it can be uniquely determined by the intrinsic set Ω together with the initial point and the initial edge direction.

Let $\boldsymbol{P}_0$ be the initial point and $\boldsymbol{X}_0$ be the initial unit direction of the edge. The Cartesian coordinates of the corresponding vertices can be recursively calculated by

$$\boldsymbol{P}_i = \boldsymbol{P}_{i-1} + \rho_i \frac{\boldsymbol{P}_{i-2}\boldsymbol{P}_{i-1}}{\| \boldsymbol{P}_{i-2}\boldsymbol{P}_{i-1} \|} \cdot \begin{pmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i & \cos\theta_i \end{pmatrix}, \quad i = 2, 3, \cdots, n \tag{7}$$

where $\boldsymbol{P}_0$ and $\boldsymbol{X}_0$ are given in advance, and $\boldsymbol{P}_1 = \boldsymbol{P}_0 + \rho_1 \boldsymbol{X}_0$.

As mentioned above, the mutual conversion between the rectangular Cartesian coordinate representation and the intrinsic variables representation of a 2D polygon can be deducted. This idea, combined with disk operations, can lead to the construction of the intrinsic variables representation of a planar disk B-spline curve.

For a planar disk B-spline curve $(\boldsymbol{P})(t) := \sum_{i=0}^{n} (\boldsymbol{P}_i)_{r_i} N_{i,k}(t)$ as described in (5), if the radius of each control point is set to be zero, it degenerates to a B-spline curve. Applying the intrinsic definitions of a 2D polygon, the set of intrinsic variables Ω of a planar disk B-spline curve can be defined as

$$\Omega = \{\{\rho_i\}_{i=1}^n, \{\theta_i\}_{i=2}^n, \{r_i\}_{i=0}^n\} \tag{8}$$

where $\{\rho_i\}_{i=1}^n$ and $\{\theta_i\}_{i=2}^n$ are the intrinsic variables of the control polygon $\{\boldsymbol{P}_0, \boldsymbol{P}_1, \cdots, \boldsymbol{P}_n\}$, and $\{r_i\}_{i=0}^n$ is the corresponding radius of each control disk.

2.2 Shape blending of planar disk B-spline curves

A curve can be determined by its intrinsic variables. Interpolation between the intrinsic variables of two planar B-spline curves will lead to the shape blending of their shapes. The following algorithm is just based on this idea. In this algorithm, it is supposed that the initial frame and the target frame are represented in disk B-spline forms of the same degree, number of control points and knot vector. Other cases will be discussed in the latter part.

Algorithm 1. Given the disk B-spline representations of the initial and target frames, i.e. the control disks $\{\boldsymbol{P}_i^0\}_{r_i}$, $i = 0, 1, \cdots, n$, disk radius $\{r_i^0\}_{i=0}^n$, $i = 0, 1, \cdots, n$ of the initial key frame, and the control disks $\{\boldsymbol{P}_i^1\}_{r_i}$, $i = 0, 1, \cdots, n$, disk radius $\{r_i^1\}_{i=0}^n$, $i = 0, 1, \cdots, n$ of the target key frame. This algorithm provides a simple way to calculate the control disks of the in-between frame. And undesired distortion can be avoided.

Step 1: Calculate $\Omega^0 = \{\{\rho_i^0\}_{i=1}^n, \{\theta_i^0\}_{i=2}^n, \{r_i^0\}_{i=0}^n\}$ as the intrinsic variables set of the initial key frame, and $\Omega^1 = \{\{\rho_i^1\}_{i=1}^n, \{\theta_i^1\}_{i=2}^n, \{r_i^1\}_{i=0}^n\}$ as that of the target frame.

Step 2: The intrinsic variables set $\Omega^t = \{\{\rho_i^t\}_{i=1}^n, \{\theta_i^t\}_{i=2}^n, \{r_i^t\}_{i=0}^n\}$ of the in-between frame at time t can be calculated by interpolation

$$\rho_i^t = (1 - f(t))\rho_i^0 + f(t)\rho_i^1, \quad i = 1, 2, \cdots, n$$

$$\theta_i^t = (1 - f(t))\theta_i^0 + f(t)\theta_i^1, \quad i = 2, 3, \cdots, n$$

$$r_i^t = (1 - f(t))r_i^0 + f(t)r_i^1, \quad i = 0, 1, \cdots, n$$

where $f(t)$ is a blending function satisfying $f(0)=0$ and $f(1)=1$. In the following examples, the blending function $f(t)$ is selected as the most commonly used linear function $f(t)=t$ for efficiency as well as stability. It can also be specifically selected for a certain purpose.

Step 3: Choose the initial point $\boldsymbol{P}_0^t$ and the initial unit edge vector $\boldsymbol{X}_0^t$. Here, $\boldsymbol{P}_0^t$ is suggested to be selected as a linear blending of $\boldsymbol{P}_0^0$ and $\boldsymbol{P}_0^1$, i.e. $\boldsymbol{P}_0^t = (1-t)\boldsymbol{P}_0^0 + t\boldsymbol{P}_0^1$. Applying linear interpolation, a temporarily used point $\overline{\boldsymbol{P}}_1^t$ can be calculated as $\overline{\boldsymbol{P}}_1^t = (1-t)\boldsymbol{P}_1^0 + t\boldsymbol{P}_1^1$. And the initial unit edge vector $\boldsymbol{X}_0^t$ is selected as $\boldsymbol{X}_0^t = \dfrac{\overline{\boldsymbol{P}}_1^t - \boldsymbol{P}_0^t}{\|\overline{\boldsymbol{P}}_1^t - \boldsymbol{P}_0^t\|}$.

Step 4: With the intrinsic variables set $\Omega^t = \{\{\rho_i^t\}_{i=1}^n, \{\theta_i^t\}_{i=2}^n, \{r_i^t\}_{i=0}^n\}$, the initial point $\boldsymbol{P}_0^t$ and the initial edge vector $\boldsymbol{X}_0^t$, the Cartesian coordinates of the control disks of the in-between frame at time t can be calculated similarly as described in (7). And the radii $\{r_i^t\}_{i=0}^n$ need not to be calculated again. So the in-between frame is generated in disk B-spline form.

2.3 Special cases discussion

If two frames have different degrees, the one with lower degree can be converted to higher degree through degree elevation algorithm. If two frames have different numbers of control points and knot vectors, small adjustment can be applied to solve this problem. Suppose the initial frame has a larger number of control points. Let n be the number of its control points and let $\boldsymbol{T} = \{t_0, t_1, \cdots, t_{n+k}\} = \{0, \cdots, 0, t_k, \cdots, t_n, 1, \cdots, 1\}$ be the knot vector, where k is the order of the curve. Let the initial frame remain unchanged, and calculate points $\boldsymbol{Q}_i$ and radii s_i on the target curve at t_i, $i=k, k+1, \cdots, n$. By interpolating $\boldsymbol{Q}_i$ and s_i, a new approximating target disk curve with n control points and knot vector $\boldsymbol{T}$ can be obtained.

If two key frames consist of several disk B-spline curves, the in-betweens are generated by interpolating the corresponding pairs. The correspondence is established by the order of drawing or specified by the user.

If the two key frame polygons are both closed, i.e. the first control disk coincides with the last control disk, the users usually would like to get the in-between frames which are also closed. However, the blending algorithm mentioned above cannot satisfy this demand. In order to handle this problem, repeated knots are selected on the two ends to preserve endpoints interpolation property. Small alternation to each control point is given. Meanwhile, restrictions to the two end control points are also given. Details of this method can be seen in Ref. [12].

3 Examples

In this section, some examples are presented, and some comparisons are given between our blending algorithm and the traditional linear blending algorithm.

Example 1. In Fig. 5, the initial frame is a little duck represented by a disk B-spline curve, and the target frame depicts an egg. Applying the blending algorithm mentioned above, the in-between frames are generated. Here, the initial frame is rendered in red, the target frame in green, and the in-between frames in blue (same for the following examples). If linear interpolation of RGB color is applied, color blending can similarly be achieved also.

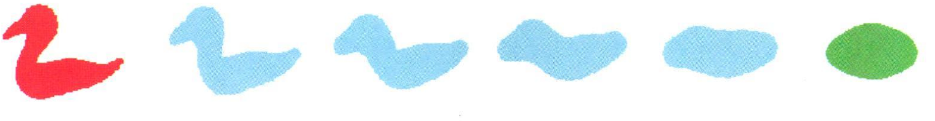

Fig. 5. Duck to egg intrinsic blending.

Example 2. In Fig. 6, a simple example is given, which sufficiently shows the advantage of our algorithm. The initial frame is a water drop, while the target frame is a water drop upside-down. Fig. 6(a) shows the blending results by applying intrinsic variables interpolation, and Fig. 6(b) shows the results of traditional linear blending. It can be obviously seen that by applying the traditional linear blending, unacceptable shrinkage occurs. But blending algorithm of intrinsic variables interpolation can absolutely avoid this problem and reflect the process of water drop upside down in a dynamic balanceable way.

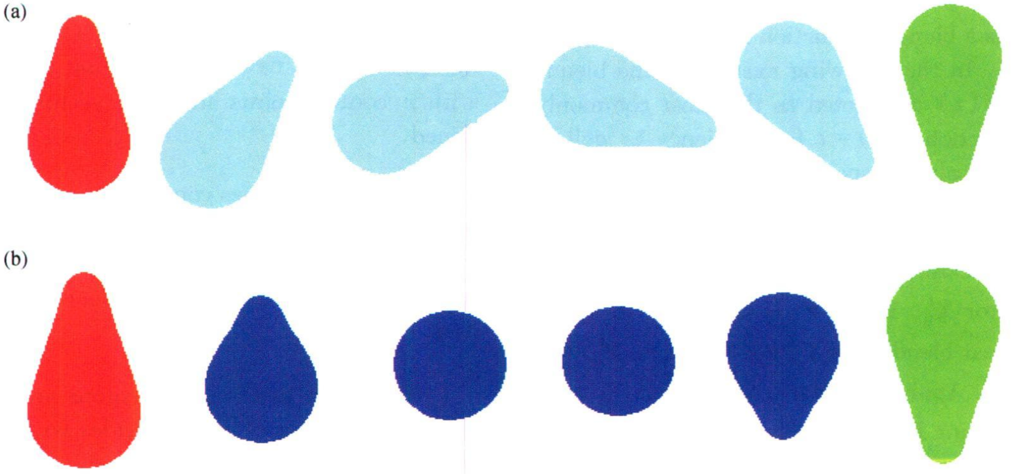

Fig. 6. Shape blending of water drop based on intrinsic variables interpolation (a) and traditional linear interpolation (b).

Example 3. In Fig. 7, the initial frame is the famous Chinese character "jing", which is the representative of capital Beijing. And the target frame is an abstract picture of a running person. Five separate disk B-spline curves are used to depict both the initial and target frames. Some strokes do not preserve high order continuity. That is the reason why disk B-spline curve representation is selected for arbitrary brushstrokes instead of simple disk Bézier curve.

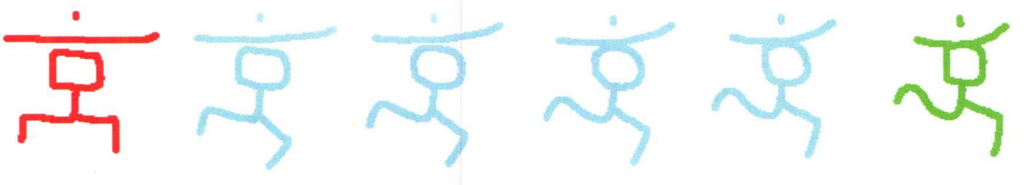

Fig. 7. Shape blending of the famous symbol of Beijing Olympic 2008.

Example 4. In Fig. 8, an example of a person doing exercise is provided. A series of motions are rendered by shape blending. The initial frame and the target frame are both closed, but the generated in-between frames are not closed if simple blending algorithm is used without taking the problem of closeness into consideration (Fig. 8(a)). The small segment which is not closed has been marked with red circle. By giving small alternation to each control point and setting restriction to the two end control points, improvements to solve the problem of closeness are given and thus the closed in-between frames are achieved (Fig. 8(b)).

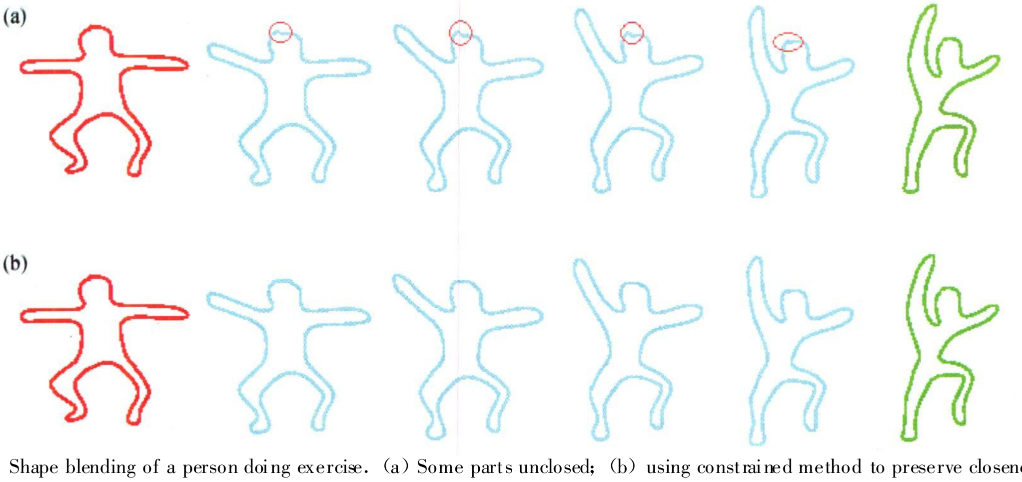

Fig. 8. Shape blending of a person doing exercise. (a) Some parts unclosed; (b) using constrained method to preserve closeness.

4 Conclusion

Because a disk B-spline curve depicts a brushstroke with control points as well as their radius, the key frames and the in-betweens depicted by disk B-spline curves preserve the style of artists' original drawing. By applying blending algorithm of disk B-spline curves, a series of in-between brushstrokes can be generated both efficiently and effectively. The blending algorithm in this paper determines the paths using intrinsic variables rather than considering the vertex paths only. The algorithm determines the intermediate shapes by interpolating the intrinsic definitions of the initial and the target shapes, and the blending results are more satisfactory than those generated by linear paths. Particularly, shrinkage as well as distortion can be avoided, which normally occurs when rotated rigid bodies are linearly blended.

References

1 Strassmann S. Hairy brushes. Siggraph' 86, 1986, 20(4): 225—232

2 Pudet T. Real time fitting of hand-sketched pressure brushstrokes. Computer & Graphics Forum, 1994, 3: 205—220

3 Chua YS. Bézier brushstrokes. Computer Aided Design, 1990, 22(9): 550—555

4 Hsu SC and Lee H. Drawing and animation using skeletal strokes. Proceedings of SIGGRAPH' 94, Computer Graphics, 1994, 28(4): 109—118

5 Pham B. Expressive brushstrokes. CVGIP: Graphical Models and Image Processing, 1991, 53: 1—6

6 Su SL, Xu YQ, Shum HY, et al. Simulating artistic brushstrokes using interval splines. In: The 5th International Conference and on Computer Graphics and Imaging, 2002, 85—90

7 Lin Q and Rokne JG. Disk Bézier curves. Computer Aided Geometric Design, 1998, 15: 721—737

8 Sederberg TW, Gao PS, Wang GJ, et al. 2D shape blending: An intrinsic solution to the vertex path problem. Proceedings of SIGGRAPH' 93, Computer Graphics, 1993, 27(4): 15—18

9 Hu SM, Li CF and Zhang H. Actual morphing: A phsical-based approach for blending two 2D/3D shapes. ACM Symposium on Solid Modeling and Applications, 2004, 11(9): 309—314

10 Yan HB, Hu SM and Martin RR. Morphing based on strain field interpolation. Journal of Computer Animation and Virtual Worlds, 2004, 15(3): 443—452

11 Che WJ. Research on application of distance transformation and medial axis transformation to morphing. Ph. D. thesis (in Chinese), Zhejiang University, 2003

12 Wang GJ, Wang GZ and Zheng JM. Computer Aided Geometric Design (in Chinese). Beijing: CHEP-Springer, 2001