Abstract: C-B-spline curve is an extension of cubic B-spline curve. It has similar properties to cubic B-spline curves and can represent conic curves such as circles, ellipses, hyperbola, etc. This paper presents a new interpolation method that can produce $G^2$-continuous C-B-spline curves without solving global systems of equations, while providing slackness control capabilities. A global slackness parameter controls the distance between the interpolating curve segments and the data segments. The basic idea of the interpolation is to blend a C-B-spline curve with a singularly parametrized polyline, which is dependent on the slackness parameter. With the low-degree polynomials and direct computation of control vertices, the method is computationally simple, and thus useful for interactive interpolation shape design and computer graphics applications.

Key-Words: interpolating, continuity control, slackness control, singular blending, tension

1 Introduction

Polynomial spline interpolation is the most traditional and widespread form of curve and surface modeling [1, 2]. Interpolation addresses the problem of constructing a polynomial spline curve or surface that passes through a set of data points. A typical method involves solving some global systems of linear equations to determine the control vertices of the curve or surface. Barsky and Greenberg presented an approach that solves a linear system and obtains bicubic uniform B-spline surfaces [3]. Based on matrix notation and linear equations, Cohen, Riesenfeld and Elber reviewed several interpolation techniques, including complete spline interpolation, nodal interpolation, and piecewise Hermite interpolation, all using B-spline bases [4]. Ma and Kruth investigated the problem of fitting with NURBS curves and surfaces, and determined the control vertices and weights through homogeneous systems [5]. Many other interpolation methods for geometric modeling have been reported in the literature and applied to various areas of industry and design, but most of them are by virtue of solving global equation systems.

With the increasing prevalence of reverse engineering applications, researchers and engineers have discovered some limitations in interpolation approaches that require solving global systems. First, the geometric models obtained are completely determined by the data points; shape properties, such as slackness, tension, and distance from the given data network, cannot be adjusted. Specifically, although the B-spline representation has local control with respect to the control vertices, the process of finding these vertices for constructing interpolating B-spline curves is global; thus the resulting B-spline curves cannot be locally controlled. This lacks of control limits the use of these interpolation methods in interactive curve and surface design environment. Second, if data points are gathered from a conic curve, a B-spline curve can not easily interpolate them. Third, interpolation produced by solving global systems may exhibit undesirable erroneous oscillations. This undesirable behavior had motivated the development of splines with tension control which usually provide methods for choosing tension parameters to achieve shape-preserving interpolation [6-13]. Finally, solving global systems obviously incurs computational cost, which is significant when the number of data points is large.

With these limitations in mind, the aim of this paper is to present a new C-B-spline curve [13, 14] interpolation method that does not require solving global systems, while provides an extra degree of freedom for users to control its slackness. In other words, by decreasing the slackness, our curve is increasingly tightened until it coincides with the polyline defined by the data points. This capability mimics the procedure of tightening a rope or a cloth around a polygon. Our algorithm uses the idea of singular blending, proposed by Loe [16]. Our interpolating spline is obtained by blending a C-B-spline and a specially parametrized polyline constructed directly from the interpolating constraint.

2 C-curves and C-B-splines
C-curves with basis \( \{ \sin t, \cos t, t, 1 \} \), \( t \in [0, \alpha] \), are an extension of cubic curves [14]. They depend on a parameter \( \alpha > 0 \). C-B-splines are an extension of cubic uniform B-splines [15]. They are analogous to cubic curves and cubic uniform B-splines respectively and can be used widely to represent conic curves [14, 15].

Definition 1 Let \( \alpha \in (0, \pi] \) be a parameter, and define a class of functions
\[
\begin{align*}
B_0(t) &= 0, \\
B_1(t) &= \frac{1}{2\alpha(1-C)}, \\
B_2(t) &= \frac{C - S - 1 + \alpha}{(1+2C) - 2S + 1+2C - 2aC} \sin t, \\
B_3(t) &= \frac{2 + C - S + (1+2C) - \alpha}{(1+2C) - 2S + 1+2C - 2aC} \cos t, \\
\end{align*}
\]
where \( B_j(t), j = 0,1,2,3 \) are basis functions on space \( \{ \sin t, \cos t, t, 1 \} \) and the transform relation is:
\[
\begin{pmatrix}
\sin t \\
\cos t \\
t \\
1
\end{pmatrix} = \frac{1}{S} \begin{pmatrix}
0 & \alpha C & 0 & \alpha \left(2C^2 - 1\right) \\
\alpha S & 0 & \alpha S & 2\alpha S \\
S & S & S & S
\end{pmatrix} \begin{pmatrix}
B_0(t) \\
B_1(t) \\
B_2(t) \\
B_3(t)
\end{pmatrix}.
\]

Fig. 1 depicts images of two classes of basis functions with \( \alpha = \frac{\pi}{4} \) and \( \alpha = \frac{\pi}{3} \) respectively.

Given a set of data points in three-dimensional space: \( \{ P_i \}_{i=1}^n \in \mathbb{R}^3 \), \( P_i \neq P_{i+1}, 1 \leq i \leq n-1, n \geq 4 \), our goal is to construct an interpolating C-B-spline curve without solving any global system to find its control vertices, and its global slackness controlled.

First, to make the number of data segments (i.e., line segments connecting the data points) the same as the number of curve segments in the C-B-spline curve to be constructed, we introduce two auxiliary data points if the control polygon is open (\( P_0 \neq P_n \)):
\[
P_0 = 2P_0 - P_2, P_{n+1} = 2P_n - P_{n-1},
\]
or if the control polygon is closed (\( P_1 = P_n \)):
\[
P_0 = P_{n-1}, P_{n+1} = P_2.
\]

For constructing a C-B-spline curve, we assign a knot value to each original point \( P_i \) using the accumulated chord-length method:
\[
t_i = 0, \quad t_i = t_{i-1} + \| P_{i-1} P_i \|, \quad i = 2,3,\cdots,n.
\]

Using \( \{ t_i \}_{i=1}^n \) as the knot vector and \( \{ P_i \}_{i=0}^n \) as the control vertices, we construct the C-B-spline space...
curve as follows:
\[
C(t; \alpha) = C_j(t; \alpha) = \sum_{i=0}^{j} B_i \left( \frac{\alpha - t_j}{t_{j+1} - t_j} \right) P_{i,j-1},
\]

\( t_j \leq t \leq t_{j+1}, j = 1, 2, \ldots, n - 1 \).

\( C(t; \alpha) \) is a continuous \( GC^2 \)-continuous C-B-spline curve [15]. Fig.2 gives two C-B-spline curves which interpolate the endpoints.

\[ f(t) \text{ is a piecewise polynomial of degree 3 and can be constructed as:} \]

\[
\begin{cases}
3 \left( 2 \cos \frac{\alpha}{3} + 1 \right)^2 B_3(t), 0 \leq t \leq \frac{\alpha}{3}, \\
\frac{1}{2 \cos \frac{\alpha}{3} - 1} \left[ 2 \left( 6 \cos^2 \frac{\alpha}{3} + 2 \cos \frac{\alpha}{3} - 1 \right) B_3(t) \right], \\
-2 \left( \cos \frac{\alpha}{3} + 1 \right) B_1(t) + \left( 4 \cos \frac{\alpha}{3} + 1 \right) B_2(t), \frac{\alpha}{3} \leq t \leq \frac{2\alpha}{3}, \\
1 - 3 \left( 2 \cos \frac{\alpha}{3} + 1 \right)^2 B_5(t), \frac{2\alpha}{3} \leq t \leq \alpha.
\end{cases}
\]

3 Singular blending C-B-spline

In order to introduce a shape parameter that enables slackness adjustment, and to avoid solving global equation systems in constructing interpolating spline curves, we blend the above C-B-spline curve \( C(t; \alpha) \) and a singularly parametrized polyline using a blending factor \( \beta \). We will show that the vertices of this polyline can be directly computed from the interpolation conditions. Our design criteria for the interpolating curve \( Q(t; \alpha, \beta) \) are as follows: (1) it must be like the C-B-spline curve \( C(t; \alpha) \); (2) it must maintain the order of continuity of \( C(t; \alpha) \) at the knots; and away from the knots, its order of continuity must be not less than 2. We use a singular blending function to reparametrize the polyline; that is, for each interval \([t_k, t_{k+1}]\), \( k = 1, 2, \ldots, n - 1 \), we construct a 2th-level singular blending function \( S_k(t) \in C^2[t_k, t_{k+1}] \) satisfying the following conditions:

\[
S_k(t_k) = 0, S_k'(t_{k+1}) = 1, \\
S_k'(t_k) = S_k''(t_{k+1}) = 0, \\
S_k''(t_{k+1}) = S_k''(t_k) = 0.
\]
We now use the above singular blending function to construct a reparameterized space polyline, called singular polyline, in which each line segment is parametrized on \([t_j, t_{j+1}]\) and connects the yet-to-be-determined vertices \(V_j\) and \(V_{j+1}\). These vertices are dependent on \(\beta\), thus we express the singular polyline as follows:

\[
L(t; \alpha, \beta) = L_j(t; \alpha, \beta) = \left(1 - S_j(t; \alpha)\right)V_j + S_j(t; \alpha)V_{j+1}, t_j \leq t \leq t_{j+1}, t_j \neq t_{j+1}
\]

It follows that

\[
\frac{\partial}{\partial t} L_j(t_j; \alpha, \beta) = 0, \quad \frac{\partial^2}{\partial t^2} L_j(t_j; \alpha, \beta) = 0;
\]

\(j = 1, 2, \ldots, n\)

Now, by blending the singular polyline \(L(t; \alpha, \beta)\) and C-B-spline \(C(t)\), both defined on \([t_1, t_n]\), using \(\beta\) as the blending factor, we obtain a curve with an additional parameter \(\beta\):

\[
Q(t; \alpha, \beta) = (1 - \beta)C(t; \alpha) + \beta L(t; \alpha, \beta), t_1 \leq t \leq t_n
\]

\(Q(t; \alpha, \beta)\) satisfies interpolation conditions:

\[
Q(t_j; \alpha, \beta) = P_j, j = 1, 2, \ldots, n
\]

that is

\[
\beta V_j = P_j - (1 - \beta)C_j(t_j; \alpha), j = 1, 2, \ldots, n - 1,
\]

\[
\beta V_n = P_n - (1 - \beta)C_n(t_n; \alpha),
\]

Moreover, since the C-B-spline curve \(C(t; \alpha)\) interpolates the endpoints \(P_1, P_n\), we obtain

\[
V_1 = P_1, \quad V_n = P_n.
\]

Substitute \(V_j, j = 1, 2, \ldots, n\) into \(Q(t; \alpha, \beta)\), it follows:

\[
Q(t; \alpha, \beta) = Q_j(t; \alpha, \beta) = (1 - \beta)C_j(t; \alpha) + (1 - S_j(t; \alpha))\left[P_j - (1 - \beta)C_j(t; \alpha)\right] + S_j(t; \alpha)\left[P_{j+1} - (1 - \beta)C_{j+1}(t; \alpha)\right]
\]

Using this formula, we can evaluate singular C-B-spline without explicitly determining the vertices, \(V_j, j = 1, 2, \ldots, n\), that define the singular polyline \(L(t; \alpha, \beta)\). It is clear that, for any blending factor \(\beta\), the resulting curve \(Q(t; \alpha, \beta)\) interpolates the data points \(\{P_j\}_{j=1}^n\). Fig.3 gives two singular C-B-spline curves with their control polygons.

**Theorem 1.** Singular blending C-B-spline curve \(Q(t; \alpha, \beta)\) is 2nd geometric continuous, that is, at every point of \(Q(t; \alpha, \beta)\), there is a unique unit tangent vector (derivative with respect to arc-length parameter) and a unique curvature vector (second derivative with respect to arc-length parameter).

**Proof:**

\[
\frac{\partial}{\partial t} Q(t_j; \alpha, \beta) = Q(t_j; \alpha, \beta) = P_j,
\]

\[
\frac{\partial}{\partial t} Q(t_j; \alpha, \beta) = \frac{1 - \beta}{2(t_{j+1} - t_j)}[P_{j+2} - P_j],
\]

\[
\frac{\partial}{\partial t} Q(t_{j+1}; \alpha, \beta) = \frac{1 - \beta}{2(t_{j+2} - t_j)}[P_{j+2} - P_j],
\]

\[
\frac{\partial}{\partial t} Q(t_j; \alpha, \beta) = \frac{t_j - t_{j-1}}{t_{j+1} - t_j} \frac{\partial}{\partial t} Q(t_j; \alpha, \beta)
\]
\[ \frac{\partial^2}{\partial t^2} Q(t_j^+; \alpha, \beta) = (1 - \beta) C''(t_j^+; \alpha) \]
\[ = \frac{\alpha S(1 - \beta)}{2(1 - C)(t_{j+1} - t_j)^2} \left[ P_{j-1} - 2P_j + P_{j+1} \right], \]
\[ \frac{\partial^2}{\partial t^2} Q(t_j^-; \alpha, \beta) = (1 - \beta) C''(t_j^-; \alpha) \]
\[ = \frac{\alpha S(1 - \beta)}{2(1 - C)(t_j - t_{j-1})^2} \left[ P_{j-1} - 2P_j + P_{j+1} \right], \]
\[ \frac{\partial^2}{\partial t^2} Q(t_j^+; \alpha, \beta) = \left( t_{j+1} - t_j \right)^2 \frac{\partial^2}{\partial t^2} Q(t_j^-; \alpha, \beta). \]

The proof is completed.

In our formulation of the singular C-B spline \( Q(t; \alpha, \beta) \), we have not restricted the range of \( \beta \). But, for the purpose of practical shape modelling, we shall restrict \( \beta \in [0,1] \), and call these curves the standard blending C-B-spline. When \( \beta = 0 \), the blending C-B-spline curve is reduced to the cubic non-uniform B-spline curve \( C(t; \alpha) \), in which case the singular polyline \( L(t; \alpha, \beta) \) is redundant; when \( \beta = 1 \), the singular C-B-spline is reduced to the polyline connecting the data points \( \{P_i\}_{i=1}^{n} \), but reparametrized using the singular blending function \( S_j(t; \alpha) \).

4 Some interpolating examples

In every \( j \)th \((1 \leq j \leq n-1) \) segment of the singular C-B-spline curve, for any fixed parameter \( t \in [t_j, t_{j+1}] \), the distance function decreases uniformly and linearly with \( \beta \) when \( \beta < 1 \), reaches the minimum value 0 when \( \beta = 1 \). Fig.4 and Fig.5 give two interpolating curves.
4 Conclusion

We have presented a new method for constructing singular C-B-spline curves that pass through a set of given data points without solving global equation systems. Slackness control is achieved by adjusting the blending parameter. The additional versatility and control is useful for interactive interpolation design, such as modifying models obtained using reverse engineering techniques. Moreover, the interpolation using singular blending mimics the procedure of tightening a deformable surface around a polygonal mesh, and thus can be used to simulate such animations.

References: