Accelerating Accurate B-spline Free-form Deformation of Polygonal Objects

Jieqing Feng* Qunsheng Peng

State Key Lab. of CAD&CG, Zhejiang University, Hangzhou, 310027, P.R.China

Email: {jqfeng, peng}@cad.zju.edu.cn Tel: +86-571-7951045 Fax: +86-571-7951780

Abstract A previous paper described an algorithm for accurate B-spline free-form deformation of polygonal objects to produce triangular Bézier patches. However that algorithm computed the control points of resulting patches using a generalized de Casteljau algorithm, which is expensive to compute. In this short note, we describe an algorithm that instead uses polynomial interpolation; both theoretical analysis and implementation results show that this new algorithm runs faster than the original.

Key words Free-form deformation, B-spline, Bézier curve, triangular Bézier patch, Polynomial interpolation, Linear equation system

1 Introduction

The accurate B-spline free-form deformation(FFD) of polygonal object was ever proposed in [1] by authors, where the deformation of polygonal objects is represented as a set of triangular Bézier patches[1], whose degree is the summation of three directional degrees of B-spline volume. The method is based on Bernstein polynomial composition and generalized de Casteljau algorithm. Though the accurate FFD solves the sampling problem in deformation of polygonal objects, the computational burden is heavy because generalized de Casteljau algorithm is time-consuming.

In this note, we propose a fast accurate B-spline FFD method. The polynomial interpolation is adopted to expand Bernstein polynomial composition rather than the generalized de Casteljau algorithm. The polynomial interpolation is just a procedure of solving a linear equations system. By properly choosing sampling points, the matrix of linear system and its inverse can be computed and saved in advance, thus processing time is saved.

2 Cost of the de Casteljau algorithm

All notations in the rest of paper is same with those in [1]. It is known that the deformations of line segment and triangular mesh under tensor-product Bézier volume are Bézier curve and triangular Bézier patch, whose degrees are equal to the summation of three directional degrees of Bézier volume. The control points of the result curve and patch can be evaluated through the generalized de Casteljau algorithm. The following analysis shows that it is time-consuming.

To compute a point on a tensor product Bézier volume of degrees n_u, n_v, n_w along three directions, the numbers of ¤oating multiplication($\otimes$), addition($\oplus$) and subtraction($\ominus$) involved in the de Casteljau algorithm are:

$$V(n_u,n_v,n_w)=\frac{3}{2}\left[n_u(n_u+1)(n_v+1)(n_w+1)+n_v(n_v+1)(n_w+1)+n_w(n_w+1)\right](2\otimes+\oplus+\ominus) \quad (1)$$

The computational cost of the procedures `Compute_Auvwijk` (in [1])is just identical with $V(n_u,n_v,n_w)$ except for loops control. The computational cost $Curve^{(de\ Casteljau)}(n_u,n_v,n_w)$ for evaluating all control points of result Bézier curve is:

$$Curve^{(de\ Casteljau)}(n_u,n_v,n_w)=(n_u+1)(n_v+1)(n_w+1)V(n_u,n_v,n_w) \quad (2)$$

*Corresponding author

Here we assume the combinatorial number is pre-computed and saved in a look-up table. Its computational cost is omitted . Similarly, to evaluate all of the control points of result triangular Bézier patch $\mathbf{R}(u, v, w)$, the ¤oating computational cost $Patch^{(de\ Casteljau)}(n_u, n_v, n_w)$ is:

$$Patch^{(de\ Casteljau)}(n_u, n_v, n_w) = \frac{1}{8}\left[(n_u + 1)(n_u + 2)(n_v + 1)(n_v + 2)(n_w + 1)(n_w + 2)\right] V(n_u, n_v, n_w) \tag{3}$$

Formulae (1)-(3) show that the computation cost of the generalized de Casteljau algorithms is very high. In the next section, we will give an interpolation algorithm to compute the control points, which can dramatically decease the computational cost.

3 Computing control points through polynomial interpolation

From numerical analysis, we know that a polynomial of degree n can be reconstructed accurately through interpolating $(n + 1)$ different sampling points in its domain. Interpolation can be implemented by solving a linear equation system, where the matrix is $(n + 1) \times (n + 1)$. In the accurate deformation, the deformation of line and triangle are Bézier curve and patch, which are Bernstein polynomials with 1 and 2 variables respectively. Thus we can compute the control points of curve and patch through polynomial interpolation according to the above statement.

3.1 Reconstruction of Bézier curve

Let Bézier curve $\mathbf{C}(t)$ be the accurate deformation of line segment, whose expression is:

$$\mathbf{C}(t) = \sum_{i=0}^{N} \mathbf{C}_i B_{i,N}(t) \tag{4}$$

where $B_{i,N}(t)$ is Bernstein polynomial of degree N. Our goal is to compute the control points $\{\mathbf{C}_i\}_{i=0}^{N}$. We choose $(n + 1)$ different sampling points $\{t_i\}_{i=0}^{N}$ on it. The points $\mathbf{C}(t_i)$ on the curve can be evaluated through traditional FFD algorithm, *i.e.*, deforming a single point. Then we can establish a linear equation system as follows:

$$\begin{pmatrix} B_{0,N}(t_0) & B_{1,N}(t_0) & \cdots & B_{N,N}(t_0) \\ B_{0,N}(t_1) & B_{1,N}(t_1) & \cdots & B_{N,N}(t_1) \\ \vdots & \vdots & \ddots & \vdots \\ B_{0,N}(t_N) & B_{1,N}(t_N) & \cdots & B_{N,N}(t_N) \end{pmatrix} \begin{pmatrix} \mathbf{C}_0 \\ \mathbf{C}_1 \\ \vdots \\ \mathbf{C}_N \end{pmatrix} = \begin{pmatrix} \mathbf{C}(t_0) \\ \mathbf{C}(t_1) \\ \vdots \\ \mathbf{C}(t_N) \end{pmatrix} \tag{5}$$

We denote the $(N + 1) \times (N + 1)$ matrix in (5) as $\mathbf{B}_2$. The matrix $\mathbf{B}_2$ is non-degenerate if there is no two identical sampling points among $\{t_i\}_{i=0}^{N}$. Of course user can specify the sampling points freely. We will introduce a standard sampling scheme. We use $\left\{\frac{i}{N}\right\}_{i=0}^{N}$ as the sampling points.The control points of the Bézier curve $\mathbf{C}(t)$ can be obtained as follows:

$$\begin{pmatrix} \mathbf{C}_0 \\ \mathbf{C}_1 \\ \vdots \\ \mathbf{C}_N \end{pmatrix} = \mathbf{B}_2^{-1} \begin{pmatrix} \mathbf{C}(t_0) \\ \mathbf{C}(t_1) \\ \vdots \\ \mathbf{C}(t_N) \end{pmatrix} \tag{6}$$

Remember that matrix $\mathbf{B}_2^{-1}$ need to be computed only once. Thus it can be pre-computed and saved for further use.

As described above, the computational cost consists of $(N + 1)$ sampling points on Bézier volume and a multiplication between matrix and vector. The total number of ¤oating computation to compute all of control points of the curve is:

$$Curve^{interpolation}(n_u, n_v, n_w) = \left[(N + 1)V(n_u, n_v, n_w)\right] + \left[(N + 1)^2 \otimes + N(N + 1)\oplus\right] \tag{7}$$

Where $N = n_u + n_v + n_w$. The £rst squarely bracketed term is for evaluating the $(N + 1)$ sampling points through traditional FFD method. The second term is for matrix multiplication.

3.2 Reconstruction of triangular Bézier patch

Let $\mathbf{R}(u,v,w)$ be accurate deformation of a triangular mesh, whose expression is:

$$\mathbf{R}(u,v,w) = \sum_{i+j+k=N} \mathbf{R}_{ijk} B_{ijk}^N(u,v,w) \tag{8}$$

We sort Bernstein polynomials $B_{i,j,N-i-j}^N(u,v,1-u-v)$ in equation (8), which are de£ned on the 2D simplex, in lexicographic order of their subscripts (i,j) as follows:

$$\begin{array}{ccccc}
B_{0,0,N} & B_{0,1,N-1} & B_{0,2,N-2} & \cdots & B_{0,N,0} \\
 & B_{1,0,N-1} & B_{1,1,N-2} & \cdots & B_{1,N-1,0} \\
 & \cdots & \cdots & \ddots & \cdots \\
 & & B_{N-2,0,2} & B_{N-2,1,1} & B_{N-2,2,0} \\
 & & & B_{N-1,0,1} & B_{N-1,1,0} \\
 & & & & B_{N,0,0}
\end{array}$$

Let them be re-noted as $\left\{\tilde{B}_i(\mathbf{u})\right\}_{i=0}^{\frac{(N+1)(N+2)}{2}}$, where $\mathbf{u} = (u,v,1-u-v)$. Any $\frac{(N+1)(N+2)}{2}$ *non-degenerate* sampling points in the patch domain can be used to accurately interpolate the triangular Bézier patch $\mathbf{R}(u,v,w)$. We will not discuss the condition of *non-degenerate*, which is out of scope of this paper. In our implementation, the standard sampling scheme is adopted to interpolate the triangular Bézier patch. The sampling points are just the triples of subscripts of base function divided by N, *i.e.* $\left(\frac{i}{N}, \frac{j}{N}, \frac{N-i-j}{N}\right)$. From the theory of multi-variable spline, we know that it is a valid sampling scheme. Thus it is invertible. The ¤oating computation for all control points of the triangular Bézier patch is:

$$Patch^{interpolation}(n_u,n_v,n_w) = [N_B V(n_u,n_v,n_w)] + \left[N_B^2 \otimes + N_B(N_B-1)\oplus\right] \tag{9}$$

Where $N_B = \frac{(n_u+n_v+n_w+1)(n_u+n_v+n_w+2)}{2}$, which is the number of the control points of patch. Similarly the inverse of matrix related to $\tilde{B}_i(\mathbf{u})$ need also to be computed once.

4 Discussion and implementation

The generalized de Casteljau algorithm is numerically stable, but it takes more time than the interpolation method. Users can choose one between them: if they pursue accuracy, the generalized de Casteljau algorithm is suitable; if they pursue fast interaction, the interpolation algorithm is preferable. Some examples of theoretical comparisons are listed in table 1.

$Curve^{(de\ Casteljau)}(2,2,2) = 2106 \otimes + 1053 \oplus + 1053\ominus$ $Curve^{(interpolation)}(2,2,2) = 595 \otimes + 315 \oplus + 273\ominus$
$Curve^{(de\ Casteljau)}(3,3,3) = 16128 \otimes + 8064 \oplus + 8064\ominus$ $Curve^{(interpolation)}(3,3,3) = 2620 \otimes + 1350 \oplus + 1260\ominus$
$Patch^{(de\ Casteljau)}(2,2,2) = 16848 \otimes + 8424 \oplus + 8424\ominus$ $Patch^{(interpolation)}(2,2,2) = 2968 \otimes + 1848 \oplus + 1092\ominus$
$Patch^{(de\ Casteljau)}(3,3,3) = 252000 \otimes + 126000 \oplus + 126000\ominus$ $Patch^{(interpolation)}(3,3,3) = 16885 \otimes + 9900 \oplus + 6930\ominus$

Table 1: Some examples of calculation comparisons of two methods in theory

We have implemented the proposed algorithm on an SGI Octane workstation. In each example from Figure 1 to Figure 6, Part(a) displays an object to be deformed, represented as triangular meshes. Part (b) gives the result when FFD acts on the vertex of object. The B-spline control lattice is shown by points and dashed lines. Part (c) shows the object after subdivision and re-triangulation. If the B-spline volume is identical with a Bézier volume, no subdivision is performed. The parametric space of B-spline volume is described by dashed lines. Part (d) is the accurate FFD result, where each mesh is a triangular Bézier surface patch. Time comparisons of implementations are listed in table 2.

Figure	Triangles	Degree of B-spline	Patch(sec): $\frac{de\ Casteljau}{interpolation}$	Curve(sec): $\frac{de\ Casteljau}{interpolation}$
1	376	$2 \times 2 \times 1$	$1.25sec/0.24sec = 5.21$	$0.57sec/0.16sec = 3.56$
2	2910	$2 \times 2 \times 2$	$25.36sec/3.25sec = 7.80$	$8.78sec/1.80sec = 4.88$
3	1188	$2 \times 1 \times 1$	$1.37sec/0.39sec = 3.51$	$0.84sec/0.30sec = 2.80$
4	722	$1 \times 2 \times 2$	$2.37sec/0.47sec = 5.04$	$1.10sec/0.30sec = 3.67$
5	6558	$2 \times 2 \times 2$	$57.43sec/7.35sec = 7.81$	$19.84sec/4.07sec = 3.56$
6	1024	$3 \times 3 \times 3$	$87.99sec/4.62sec = 19.04$	$15.81sec/1.79sec = 8.83$

Table 2: Implementation time comparison

5 Conclusion

In this note, B-spline FFD is explored by means of polynomial interpolation. Both theoretical analysis and implementations show that the proposed method is faster than previous one[1].

References

[1] J. Feng, P. Heng *et al.*, “Accurate B-spline Free-Form Deformation of Plygonal Objects”. *Journal of Graphics Tools*, Vol.3, No.3, pp. 11-27, 1998.

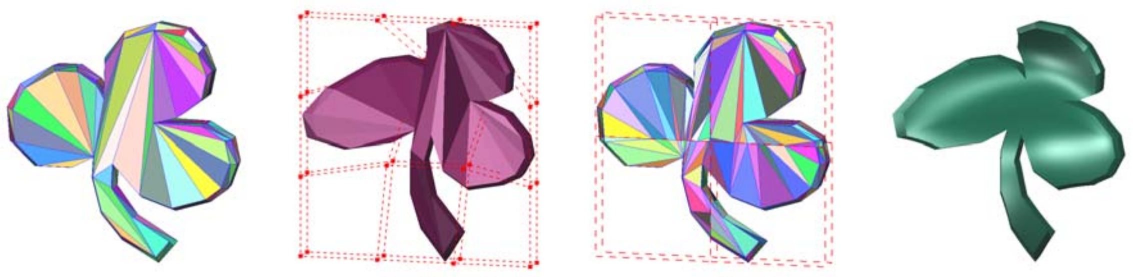

Figure 1: The degree of the B-spline volume is $2{\times}2{\times}1$, with $4{\times}4{\times}2$ control points. Original object contains 200 triangles, after subdivision and re-triangulation it has 376 triangles.

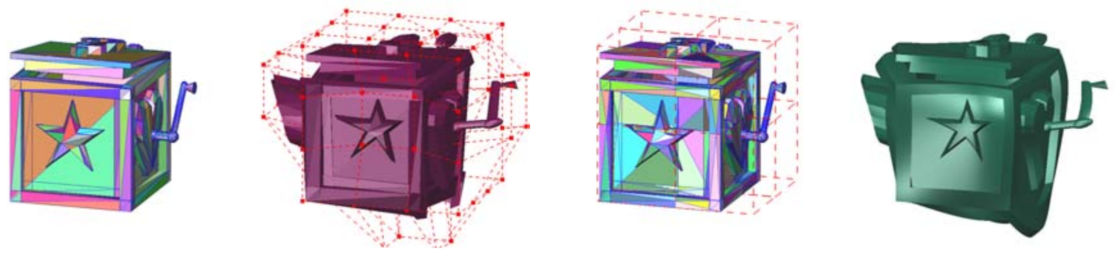

Figure 2: The degree of the B-spline volume is $2{\times}2{\times}2$, with $4{\times}4{\times}4$ control points. Original object contains 1766 triangles, after subdivision and re-triangulation it has 2910 triangles.

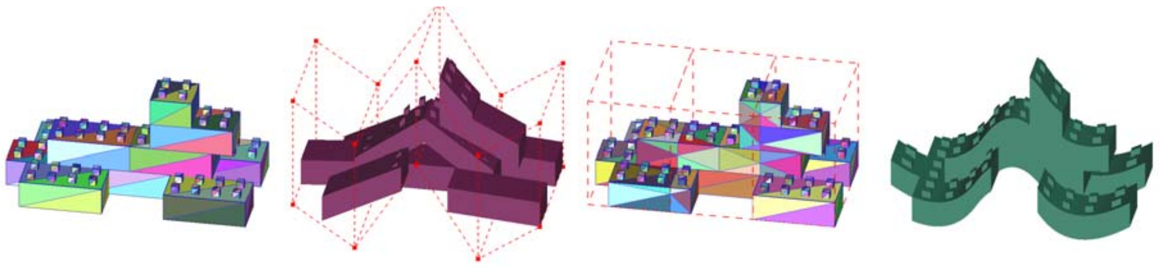

Figure 3: The degree of the B-spline volume is $2{\times}2{\times}1$, with $4{\times}4{\times}2$ control points. Original object contains 1032 triangles, after subdivision and re-triangulation it has 1188 triangles.

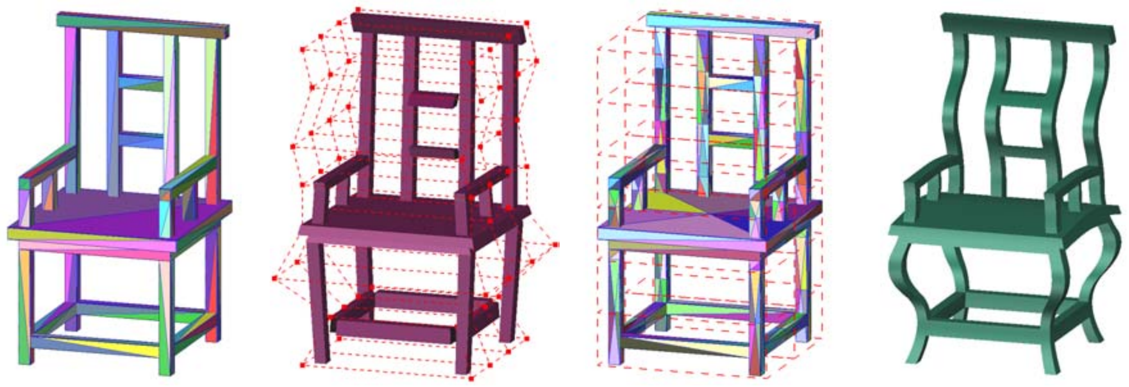

Figure 4: The degree of the B-spline volume is $1{\times}2{\times}2$, with $4{\times}4{\times}2$ control points. Original object contains 214 triangles, after subdivision and re-triangulation it has 722 triangles.

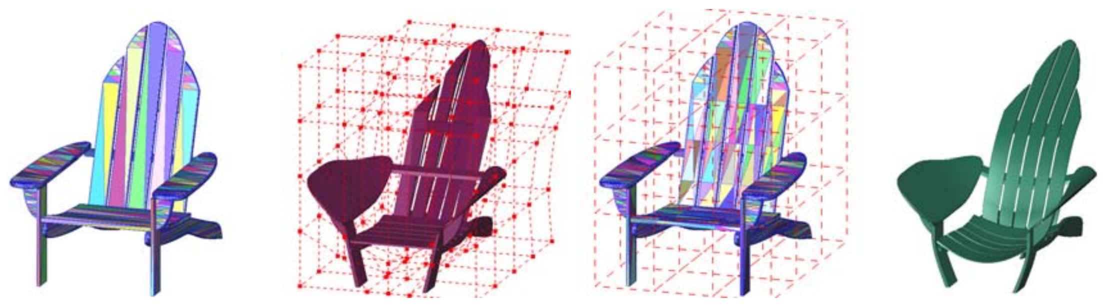

Figure 5: The degree of the B-spline volume is $2{\times}2{\times}2$, with $5{\times}5{\times}5$ control points. Original object contains 3878 triangles, after subdivision and re-triangulation it has 6558 triangles.

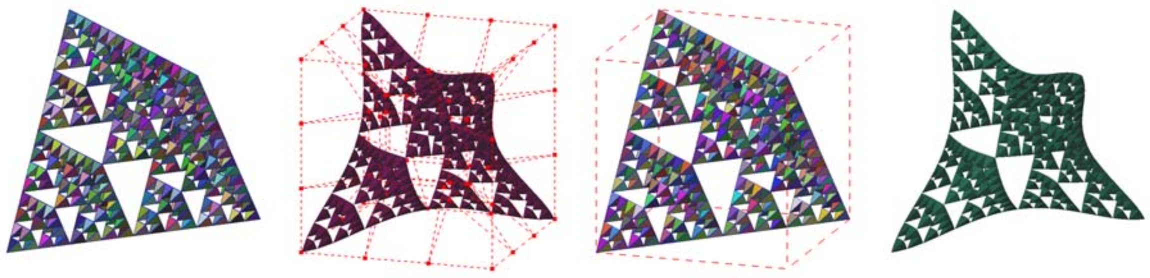

Figure 6: The degree of the B-spline volume is $3{\times}3{\times}3$, with $4{\times}4{\times}4$ control points. Original object contains 1024 triangles.