Representing Spheres and Ellipsoids Using Periodic NURBS Surfaces with Fewer Control Vertices

Kaihuai Qin
Dept.of Computer Sci.& Tech.
Tsinghua University
Beijing 100084, CHINA

Wenping Wang
Dept. of Computer Science
The University of Hong Kong
Hong Kong, CHINA

Zesheng Tang
Dept.of Computer Sci. & Tech.
Tsinghua University
Beijing 100084, CHINA

Abstract

A new kind of NURBS representation for periodic curves is introduced. The method for periodic curves is extended to derive a compact and symmetric representation of the sphere or the ellipsoid using a piecewise bi-quadratic NURBS surface with only eight distinct control vertices.

1. Introduction

Nonuniform Rational B-Splines (NURBS) have been widely used in computer aided design and modeling, since NURBS can represent not only free curves and surfaces but also conics and quadrics precisely. A few researchers have studied how to represent the sphere using NURBS, but many parameters such as knots, control vertices and weights etc. are required. Forty-five control vertices are required in Piegl and Tiller's method[5], and sixteen control vertices in Qin's method[6]. In this paper, a new method, which requires only eight distinct control vertices, is presented.

2. Representing the sphere using a periodic NURBS surface

Suppose an NURBS curve of degree p is *g*iven as

$$\mathbf{C}_i(u)=\sum_{j=0}^{p}N_{i+j,p}(u)w_{i+j}\mathbf{V}_{i+j}\Big/\sum_{j=0}^{p}N_{i+j,p}(u)w_{i+j},\ 0\le u=\frac{t-t_{i+p}}{t_{i+p+1}-t_{i+p}}<1$$

where $\mathbf{V}_{i+j}, w_{i+j}>0\ (j=0,1,\cdots,p)$ and $\{t_i\}\left(t_i<t_{i+p+1}, \text{ and } t_i\le t_{i+1}\right)$ are the control vertices, weights and knot vector of the curve, respectively, and $N_{j,p}(u)$ are the normalized B-spline functions[1] defined over $\{t_i\}$. The following theorem can be used to identify whether an NURBS curve $\mathbf{C}_i(u)$ of degree p is a circular arc or not[2]:

Theorem 1 The NURBS curve $\mathbf{C}_i(u)$ of degree p is a circular arc if and only if

I) $\mathbf{V}_{i+j}\ (j=0,1,\cdots,p)$ and $\mathbf{P}_c$ are coplanar;

II) $$\sum_{j=0}^{s}C_p^jC_p^{s-j}\left[\left(\mathbf{U}_{i+s-j}-\mathbf{P}_cH_{i+s-j}\right)\bullet\left(\mathbf{U}_{i+j}-\mathbf{P}_cH_{i+j}\right)-R^2H_{i+s-j}H_{i+j}\right]=0,$$

$$s=0,1,..,2p$$

where

$$\begin{bmatrix}\mathbf{U}_i\\ \mathbf{U}_{i+1}\\ \vdots\\ \mathbf{U}_{i+p}\end{bmatrix}=\mathbf{AM}\begin{bmatrix}w_i\mathbf{V}_i\\ w_{i+1}\mathbf{V}_{i+1}\\ \vdots\\ w_{i+p}\mathbf{V}_{i+p}\end{bmatrix},\quad\begin{bmatrix}H_i\\ H_{i+1}\\ \vdots\\ H_{i+p}\end{bmatrix}=\mathbf{AM}\begin{bmatrix}w_i\\ w_{i+1}\\ \vdots\\ w_{i+p}\end{bmatrix}$$

$$\mathbf{A}=\left[A_{i,j}\right]\ (i,j=0,1,\cdots,p),$$

$$A_{i,j}=\begin{cases}0, i<j;\\ C_i^j\big/C_p^j, i\ge j;\end{cases}\quad C_p^j=\begin{cases}0, j<0 \text{ or } j>p;\\ 1, j=0,p;\\ p!/\left[j!(p-j)!\right], j>0\end{cases}$$

M is the coefficient matrix of B-splines[3], $\mathbf{P}_c$ is the intersection of two different normals to the NURBS curve, or the center of the circular arc; and R is the distance from $\mathbf{P}_c$ to the start point of the curve, or the radius of the arc.

From Theorem 1, it is easy to obtain the necessary and sufficient conditions for an NURBS curve to represent a circular arc[4]. Given the circle as shown in Fig.1, we can represent it by a piecewise NURBS curve of degree *2* with *4* control vertices that form a square.

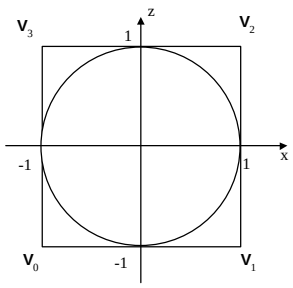

Fig 1. The circle.

$$\mathbf{C}_i(t)=\frac{\sum_{j=0}^{2}w_i\mathbf{V}_{i+j}N_{j,2}(t)}{\sum_{j=0}^{2}w_iN_{j,2}(t)},\quad i=0,1,2,3;\ t\in\left[\frac{1}{3},\frac{1}{2}\right],\quad (1)$$

where $w_0 = 2/3$, $w_1 = 1/3$, $w_2 = 1$, $\{t_0, t_1, t_2, t_3, t_4, t_5\} = \{0, 0, 1/3, 1/2, 1, 1\}$, $\mathbf{V}_j = \begin{Bmatrix} x_j \\ z_j \end{Bmatrix}$, j=0,.,3; and $\mathbf{V}_j = \mathbf{V}_{j-4}$ if $j \geq 4$. Obviously, here only four control vertices are required for representing the circle.

In order to sweep the sphere, we can rotate the hemi-circle C_h, which is composed of $\mathbf{C}_1(t)$ and $\mathbf{C}_2(t)$ in Eq.(1), about the *z*-axis by 360°. In geometry, rotating an NURBS curve need only rotate the control vertices of the curve because of its affine transformation invariance. Thus, we can get 8 control vertices $\mathbf{P}_0 = (-1, -1, -1)$, $\mathbf{P}_1 = (1, -1, -1)$, $\mathbf{P}_2 = (1, 1, -1)$, $\mathbf{P}_3 = (-1,1, -1)$, $\mathbf{P}_4 = (-1, -1, 1)$, $\mathbf{P}_5 = (1, -1,1)$, $\mathbf{P}_6 = (1, 1,1)$ and $\mathbf{P}_7 = (-1, 1, 1)$ of the NURBS surface of degree 2×2. Now using the C pseudocode in Figure 2, we can give Algorithm 1 for an NURBS surface of degree 2×2 with eight vertices to represent the sphere precisely.

```
typedef float *POINT;
typedef float POINT4[4];
typedef POINT *PPOINT;
float Weights[] = {2/3, 1/3, 1};
float  Knots[] = {0, 0, 1/3, 1/2, 1, 1};
PPOINT **pCtrlPts;
POINT4  P[8]={{P0},{P1},{P2},{P3},{P4},{P5},{P6},{P7}};

POINT4  PtsOfSphereByNURBS(I, J, u, v, P)
{ POINT4 Pt;
  pCtrlPts = (PPOINT **) malloc(4*sizeof(PPOINT*));
  for (each i from 0 to 3)
     pCtrlPts[i] = (PPOINT*)malloc(4*sizeof(PPOINT));
  pCtrlPts[0][0]=P[0], pCtrlPts[0][1]=P[2];
  pCtrlPts[0][2]=P[6], pCtrlPts[0][3]=P[4];
  pCtrlPts[1][0]=P[1], pCtrlPts[1][1]=P[3];
  pCtrlPts[1][2]=P[7], pCtrlPts[1][3]=P[5];
  pCtrlPts[2][0]=P[2], pCtrlPts[2][1]=P[0];
  pCtrlPts[2][2]=P[4], pCtrlPts[2][3]=P[6];
  pCtrlPts[3][0]=P[3], pCtrlPts[3][1]=P[1];
  pCtrlPts[3][2]=P[5], pCtrlPts[3][3]=P[7];
  for (each i from 0 to 2)
     for (each j from 0 to 2)
        pCtrlPts[i][j][3] = Weights[i]*Weights[j];
  for (each k from 0 to 3)
  {Pt[k] = 0.;
   for (each i from 0 to 2)
   { s = i+I;
     if (s is greater than 3) s decreases by 4;
     for (each j from 0 to 2)
     {   t = j+J;
         if (t is greater than 3) t decreases by 4;
         if (k is less than 3)   Pt[k] increases by the value of
              Ni,2(u)*Nj,2(v)*pCtrlPts[s][t][k]*pCtrlPts[i][j][3];
         else
              Pt[k] increases by  Ni,2(u)*Nj,2(v)*pCtrlPts[i][j][k];
     }
   }
  }
  for (each k from 0 to 2) Pt[k] is divided by P[3];
  for (each i from 0 to 3) free(pCtrlPts[i]);
  free(pCtrlPts);
  return Pt;
}
```

Figure 2. Algorithm 1 described by the C pseudocode for computing points on the sphere.

As shown in Fig. 3, the sphere is represented by a *symmetric* and *periodic* piecewise NURBS surface of degree 2×2 with eight distinct control vertices

$$\mathbf{S}_{i,j}(t) = \frac{\sum_{k=0}^{2}\sum_{\ell=0}^{2} w_{k,\ell}\mathbf{V}_{i+k,j+\ell}N_{k,2}(u)N_{\ell,2}(u)}{\sum_{k=0}^{2}\sum_{\ell=0}^{2} w_{k,\ell}N_{k,2}(u)N_{\ell,2}(u)},$$

$$i=0,1,2,3;\text{j}=0,1;\ u,v \in \left[\frac{1}{3},\frac{1}{2}\right]$$

where $\mathbf{V}_{i,j}$ ($i, j=0,1,2,3;\ i := i-4$ if $i>3$, and $j := j-4$ if $j>3$) and $w_{k,\ell}$ ($k, \ell = 0,1,2$) are determined as the C pseudocode in Fig. 2.

The ellipsoid can be represented in the way similar to that for the sphere, provided a shear transformation is applied to the control vertices of the sphere[6].

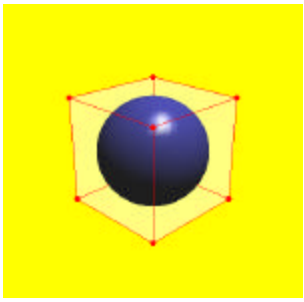

Figure 3. Sphere represented by a piecewise NURBS surface with *8* control vertices.

3. Conclusions

A new method for precisely representing the sphere or the ellipsoid by a periodic NURBS surface of degree 2×2 is proposed. It needs less control vertices than other existing methods that require much more knots, weights and control points for representing the sphere. The method is easy to understand and simple to be implemented.

References

[1] C. de Boor. *A Practical Guide to Splines*. Springer-Verlag, New York, 1978.

[2] K. Qin. Representing conics using NURBS curves of an arbitrary degree. *Tsinghua Science and Technology*. 3(2): 1009-1015, 1998.

[3] K. Qin. General matrix representations for B-splines. Proceedings of Pacific Graphics '98, 1998.

[4] K. Qin, *et al*. Representing conics using NURBS of degree two. *Computer Graphics Forum*. 11(5): 285-291, 1992.

[5] L. Piegl and W. Tiller, Curve and surface constructions using rational B-splines, *CAD*, 19(9):485-496, 1987.

[6] K. Qin. Representing quadric surfaces using NURBS surfaces. *Journal of Computer Science and Technology*. 12(3):210-216, 1997.