Non-Uniform Rational B-Splines Editing Tool

Liane Xu & Wendy Sun
6.4400 Computer Graphics
lianexu@mit.edu, wendysun@mit.edu

Abstract

We built an user-friendly editing tool to work with non-uniform rational B-splines (NURBS) curves and surfaces. NURBS is widely used in real-world applications such as modeling, and our editing tool allows users to conveniently render curves or surfaces they want with NURBS.

1 Motivation

We chose to work with NURBS because of its wide applicability, from both the theoretical and practical aspects.

On the theoretical side, NURBS provides a mathematical basis for representing both analytic shapes and free-form entities (5). Moreover, the local-control feature of NURBS makes it geometrically intuitive to design with NURBS (adding / removing a control point only changes the local shape), enabling local editing of the curve shape. NURBS curves and surfaces are invariant under common transformations. Lastly, in terms of performance, NURBS algorithms are fast and numerically stable.

On the practical side, NURBS is very popular for CAD, visualizations, industry modeling, etc.

2 Background Work

The framework of this project was built on top of Assignment #1, in which we implemented Beizer and B-splines. NURBS is a generalization of Bezier and B-splines, as it weights control points and has rational curves. In particular, it has the extra parameters of weights and non-uniform knots, which are not present in default Bezier or B-splines.

Most of our computations and algorithms are based on *The NURBS Book* by Piegl and Tiller (5), which covers this topic in extensive details. In addition, since one special aspect of NURBS is that it can draw a perfect circle as well as sphere (whereas non-rational Bezier can only give approximations), we referred to "Representing a Circle or a Sphere with NURBS" by Eberly (1) for building our procedural code to draw a sphere.

To double-check that our curves and surfaces are correct, we used an online NURBS calculator (4).

3 Approach

In this section, we lay out how we go from the math formulas of NURBS to implementation in C++.

3.1 NURBS Curve

A pth degree NURBS curve is defined as (5):

$$C(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i \cdot P_i}{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i}, \qquad a \leq u \leq b, \tag{1}$$

where $\{P_i\}$ are control points, $\{w_i\}$ are weights, $\{N_{i,p}(u)\}$ are the pth degree B-spline basis functions defined on the non-periodic and non-uniform knot vector (clamped endpoints here):

$$U = \{a, \cdots, a, u_{p+1}, \cdots, u_{m-p-1}, b, \cdots, b\}, \qquad \text{where } a = 0,\ b = 1.$$

To calculate the B-spline basis function $N_{i,p}(u)$ given any knot vector U and degree, we used a dynamic programming tabular method derived from the Cox-deBoor recurrence formula:

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \leq u < u_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,\,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,\,p-1}(u).$$

3.2 NURBS Surface

A NURBS surface is defined by a control points net, which form NURBS curves in u- and v-directions. A NURBS surface of degree p in u-direction and degree q in v-direction is defined as (5):

$$S(u,v) = \frac{\sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u) \cdot N_{j,q}(v) \cdot w_{i,j} \cdot P_{i,j}}{\sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u) \cdot N_{j,q}(v) \cdot w_{i,j}}, \qquad 0 \leq u, v \leq 1, \tag{2}$$

where the notations are similar to those in Equation (1), except that the surface is in uv-coordinates.

The surface points were able to be implemented by building on top of our curve implementation. However, we also needed to calculate the surface normals for shading. At a high level, the surface normals are the normalized cross product of the partial derivatives with respect to u and v:

$$\frac{\frac{\partial S(u,v)}{\partial u} \times \frac{\partial S(u,v)}{\partial v}}{\left|\frac{\partial S(u,v)}{\partial u} \times \frac{\partial S(u,v)}{\partial v}\right|} = \frac{(S^{(1,0)} \times S^{(0,1)})}{|(S^{(1,0)} \times S^{(0,1)})|},$$

where $S^{(k,l)}(u,v)$ is the (kth, lth) partial derivative of a NURBS surface at point *S(u, v)* with respect to the variables (u,v), which can be derived using the Leibniz rule (the product rule for finding higher order derivatives):

$$\begin{aligned} S^{(k,l)} &= \frac{1}{w^{(0,0)}} \Big(A^{(k,l)} - \sum_{i=1}^{k} \binom{k}{i} w^{(i,0)} S^{(k-i,\,l)} \\ &\quad - \sum_{j=1}^{l} \binom{l}{j} w^{(0,j)} S^{(k,\,l-j)} - \sum_{i=1}^{k} \binom{k}{i} \sum_{j=1}^{l} \binom{l}{j} w^{(i,j)} S^{(k-i,\,l-j)}\Big), \end{aligned}$$

where $A^{(k,l)}$ and $w^{(i,j)}$ are found through the following steps:

1. Find the correct knot vector span index *i* that encompasses u in knot vector U using binary search. Do the same for v in knot vector V.
2. Compute the derivatives of the degree p ($p = p$ for u-direction, $p = q$ for v-direction) B-spline basis functions:

$$N_{i,p}^{(k)}(u) = p \left(\frac{N_{i,\,p-1}^{(k-1)}}{u_{i+p} - u_i} - \frac{N_{i+1,\,p-1}^{(k-1)}}{u_{i+p+1} - u_{i+1}} \right).$$

3. Compute the derivatives of the B-spline surface in u- and v-directions using homogenized coordinates $P_{i,j} = (x, y, z, w)$:

$$D^{(k,l)} = \frac{\partial^{k+l}}{\partial^k u\, \partial^l v} S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}^{(k)}(u)\, N_{j,q}^{(l)}(v)\, P_{i,j}.$$

4. Finally, let $A^{(k,l)} = D^{(k,l)}.xyz$ and $w^{(i,j)} = D^{(i,j)}.w$.

Once we have the coordinates of each surface point and the normals, we render the surface using the subdivison surface method with the number of subdivisions set to 50 for a smooth look.

3.3 Visualize Music Scores with Curves

After we successfully implemented NURBS and built the editing tool, we put it to use by drawing a music score using the curves. Specifically, users could input a file that contains a series of music notes such as "G5," and the GUI will automatically output the music score on the screen.

To achieve this, we first built and saved the music note shape using our NURBS curve editor, and then assigned different notes to different positions / orientations of the music note shape.

4 Results

4.1 GUI for Curve Editing

Our user interface for curve editing includes the following functionalities:

1. Select any control point, and edit its location or weight;
2. Add / remove control point(s), and appropriately recalculate the knot vector;
3. Take in a circle's center coordinates and radius as input, automatically generate the corresponding NURBS control points, and draw the circle.

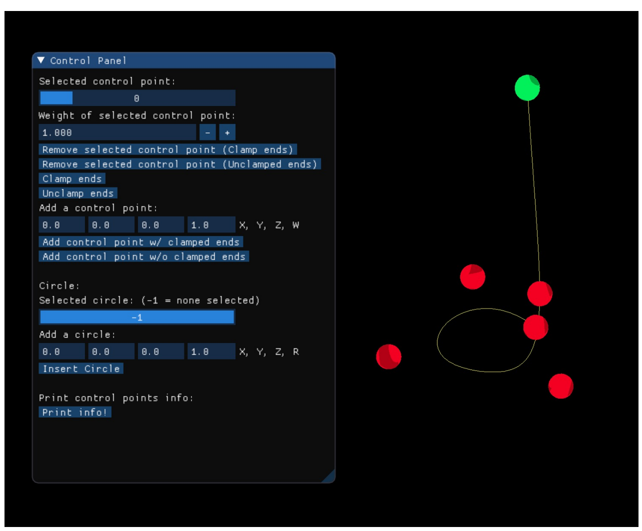
This video is a demo of our curve-editing functionalities (live-captured).

4.2 GUI for Surface Editing

Our GUI for surface editing can also select any control point and edit its location or weight. *This video* is a demo of the surface-editing functionalities (live-captured).

4.3 Music Scores Visualized with NURBS Curves

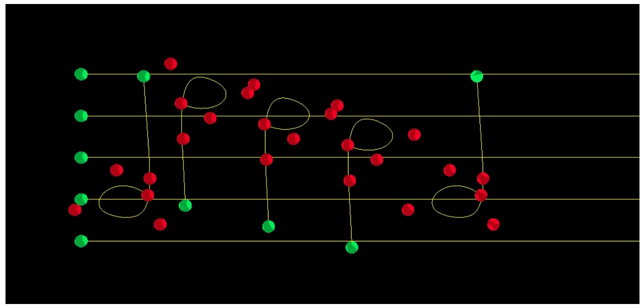
We first saved the music note shape that we built:



Then, we feed in a file that contains music notes. For example,

G4 E5 D5 C5 G4

Our GUI then outputs the following music score onto the screen:



5 Conclusion

5.1 Summary

We built a NURBS curve and surface editing tool, as well as an extension to a sheet music visualizer.

The curve-editing GUI enables users to conveniently edit the look of any NURBS curve and save control point information for later. In addition, the user could also add perfect circles to the scene, with custom center poosition and radius.

For the surface-editing tool, users could move and re-weight control points. The challenging part here during implementation was calculating the surface normals.

Lastly, we applied the tool to visualize input music scores with NURBS curves.

5.2 Future Extensions

There are two potential directions we want to further explore:

1. Implement curve / surface interpolation to given point data, i.e. find the control point locations and weights such that the resulting curve goes through the point data.
2. Since NURBS is the industry standard for modeling and it's fast as well as numerically stable, we want to explore efficient rendering of large-scale NURBS models. We want to replicate the 2023 SIGGRAPH paper by Xiong et al. (6), which uses an elastic tessellation framework for NURBS rendering.

References

[1] Eberly, D. (n.d.). *Representing a Circle or a Sphere with NURBS*.

[2] Haugland, K. (2016, April 24). *Generate and understand NURBS curves*. CodeProject. `https://www.codeproject.com/Articles/1095142/Generate-and-understand-NURBS-curves`

[3] Non-uniform rational B-spline. (2023). In *Wikipedia*. `https://en.wikipedia.org/w/index.php?title=Non-uniform_rational_B-spline&oldid=1189177837#Example:_a_circle`

[4] *NURBS demo — WebGL based online evaluator for NURBS Curves*. (n.d.). Retrieved December 12, 2023, from `https://nurbscalculator.in/`

[5] Piegl, L., & Tiller, W. (1995). *The NURBS Book*. Springer. `https://doi.org/10.1007/978-3-642-97385-7`

[6] Xiong, R., Lu, Y., Chen, C., Zhu, J., Zeng, Y., & Liu, L. (2023). ETER: Elastic Tessellation for Real-Time Pixel-Accurate Rendering of Large-Scale NURBS Models. *ACM Transactions on Graphics*, 42(4), 133:1-133:13. `https://doi.org/10.1145/3592419`