If You Know B-Splines Well, You Also Know NURBS! [*]

John Fisher, John Lowther and Ching-Kuang Shene
Department of Computer Science
Michigan Technological University
Houghton, MI 49931
{jnfisher,john,shene}@mtu.edu

ABSTRACT

This paper presents our attempt in designing intuitive and interesting materials for teaching NURBS in an undergraduate course with the help of our tool DesignMentor. This approach does not require tedious mathematics and is based on learning-by-doing and visualization. Our approach was classroom tested and used world-wide in the last seven years.

Categories and Subject Descriptors

I.3.5 [**Computational Geometry and Object Modeling**]: Curve, surface, solid, and object representations; K.3.2 [**Computers and Education**]: Computer science education

General Terms

B-Splines, NURBS

Keywords

B-Splines, NURBS, curves and surfaces

1. INTRODUCTION

B-splines and NURBS (*i.e.*, **N**on-**U**niform **R**ational **B**-**S**plines) were rarely mentioned in a typical graphics course a decade ago. Recently, as the consumer market was flooded with high quality graphics systems that all support NURBS (*e.g.*, 3D Studio Max, Lightwave 3D, Maya and trueSpace) and even after many books on B-splines and NURBS have been published, graphics textbooks and courses still do not cover these topics well [4]. Many textbooks choose a mathematical approach that often blurs the origin and intuitive meaning of NURBS. Textbooks using the programming approach rarely provide students with sufficient information for how to draw and use NURBS and do not supply an environment for students to visualize important properties and algorithms and to practice curve and surface design. To help students learn geometric processing skills that are vital to graphics, visualization and geometric design, we created a junior-level elective course **Introduction to Computing with Geometry** [2] and developed a pedagogical tool DesignMentor Version 2 or DM2.

This paper presents our materials and experience in teaching NURBS in an undergraduate course. Our experience in presenting B-splines was published in [1]. In the following, Section 2 provides a motivation indicating why NURBS are necessary, Section 3 reveals the hidden concept in the definition of NURBS (*i.e.*, a NURBS curve is the projection of a higher dimensional B-spline curve), Section 4 uses NURBSvis, a component of DM2, to help students visualize this projection concept, Section 5 presents two important properties of NURBS based on projection, and Section 6 discusses the unique NURBS shape modification operation achieved by changing weights. Then, we show how to represent conic sections in general and circles in particular in Section 7 and an application in cross-sectional design in Section 8. Finally, Section 9 has our conclusions.

[*]This work is partially supported by the National Science Foundation under grant DUE-0127401. The third author is also supported by an IBM Eclipse Innovation Award 2003.

2. MOTIVATION

We always start our discussion with a challenge: asking students to draw a circle using a B-spline curve. This is impossible and serves as a very good motivation for subsequent discussions. Figure 1 shows four B-spline curves of degree 2, 3, 5 and 7 defined by 8 control points. Even with degree 7, the B-spline curve still does not look like a circle. Hence, we need to find a method that can create circles easily, and this is the merit of discussing and using NURBS.

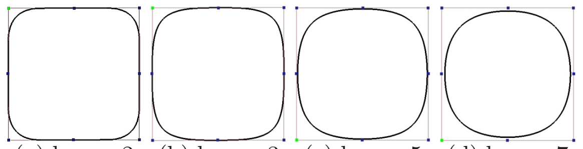

(a)degree 2 (b)degree 3 (c)degree 5 (d)degree 7

Figure 1: B-splines Can Not Represent Circles

Many students were surprised by the fact that the powerful B-splines cannot be used to represent circles. Indeed the unit circle can be represented in a different form, $x = 2t/(1+t^2)$ and $y = (1-t^2)/(1+t^2)$, which is a result discussed in calculus. However, this parametric form is *rational* (*i.e.*, the quotient of two polynomials) rather than polynomial. The subsequent discussion is mainly for finding a rational form and investigating its properties.

3. FROM B-SPLINE TO NURBS

A B-spline curve requires three elements: **(1)** a set of $n+1$ control points $\mathbf{P}_i$ $(0 \le i \le n)$, **(2)** a knot vector U of $m+1$ knots $0 = u_0 \le u_1 \le u_2 \cdots \le u_{m-1} \le u_m = 1$, and **(3)** a degree p satisfying $m = n + p + 1$. Its equation is:

$$\mathbf{C}(u) = \sum_{i=0}^{n} N_{i,p}(u)\mathbf{P}_i$$

where $N_{i,p}(u)$ is the i-th B-spline basis function of degree p and is defined recursively as follows [1]:

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

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

A NURBS curve adds a weight $w_i \ge 0$ to control point $\mathbf{P}_i$ and has an equation of

$$\begin{aligned} \mathbf{C}(u) &= \frac{1}{\sum_{i=0}^{n} N_{i,p}(u)w_i} \sum_{i=0}^{n} N_{i,p}(u)w_i\mathbf{P}_i \\ &= \sum_{i=0}^{n} R_{i,p}(u)\mathbf{P}_i \end{aligned}$$

where $R_{i,p}(u) = N_{i,p}(u)w_i / \sum_{j=0}^{n} N_{j,p}(u)w_j$, $0 \le i \le n$, are NURBS basis functions. Since all $R_{i,p}(u)$'s are rational functions, NURBS curves are rational.

It is obvious that a NURBS curve becomes a B-Spline curve if all weights are set to 1, and the former can be considered as an extension of the latter. But, the question is: what is the rationale behind the NURBS definition and the use of weights. The key is the *homogeneous coordinate* system. Consider a control point $\mathbf{P}_i = (x_i, y_i, z_i)$ with weight $w_i \ge 0$. Since $\mathbf{P}_i$ has a homogeneous coordinate $\mathbf{P}_i^h = (x_i, y_i, z_i, 1)$ and since multiplying a non-zero value to the homogeneous coordinates of a point does not change its position, $w_i\mathbf{P}_i^h = (w_ix_i, w_iy_i, w_iz_i, w_i)$ is the same point as $\mathbf{P}_i$. If we consider $w_i\mathbf{P}_i^h$ as a 4D point (because it has four coordinate values) and use the same knots and degree p, we have a 4D B-spline curve $\mathbf{C}^w(u)$ of degree p as follows:

$$\mathbf{C}^w(u) = \sum_{i=0}^{n} N_{i,p}(u) \left[w_i\mathbf{P}_i^h \right]$$

The above equation can be expanded:

$$\begin{aligned} \mathbf{C}^w(u) = (& \sum_{i=0}^{n} N_{i,p}(u)w_ix_i, \sum_{i=0}^{n} N_{i,p}(u)w_iy_i, \\ & \sum_{i=0}^{n} N_{i,p}(u)w_iz_i, \sum_{i=0}^{n} N_{i,p}(u)w_i \ ) \end{aligned}$$

Let us reinterpret this 4D point as a point in 3D homogeneous space. Its Euclidean equivalent is obtained by dividing the first three coordinate values by the fourth (*i.e.*, projecting a 4D point to the hyperplane $w = 1$). In 3D Euclidean space, this curve has the following equation:

$$\mathbf{C}(u) = \frac{1}{\sum_{i=0}^{n} N_{i,p}(u)w_i} \sum_{i=0}^{n} N_{i,p}(u)w_i\mathbf{P}_i$$

This is exactly the definition of a NURBS curve!

Thus, a NURBS curve is obtained by lifting 3D control points to 4D using weights, constructing a 4D B-spline curve, and projecting it back to 3D with a central projection (Figure 2). Note that both curves use the same knots and degree, and the weight of each control point serves as the fourth coordinate that "homogenizes" the 3D point to a 4D one.

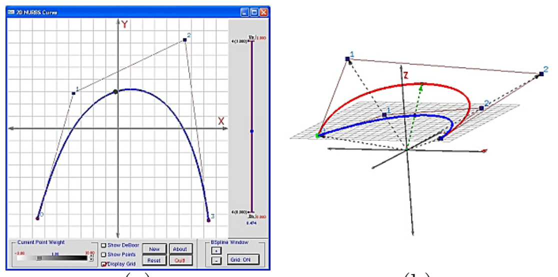
$$\begin{tikzcd}
\text{4D homogeneous coordinates} \arrow[r] & \text{4D B-spline curve} \arrow[d] \\
\text{3D control points with weights} \arrow[u] \arrow[r, dotted] & \text{3D NURBS curve}
\end{tikzcd}$$

Figure 2: The "Lifting" and "Projection" Concept

4. NURBS VISUALIZATION

To help students understand and visualize the "lifting" and "projection" concepts, a visualization system NURBSvis is included in DM2 distribution. NURBSvis is a stand-alone system and can be used without DM2's support. However, since it is difficult to display 4D objects, NURBSvis lifts a set of 2D control points to 3D, constructs a 3D B-spline curve, and projects it back to a 2D NURBS curve.

NURBSvis has two windows: the **2D NURBS Curve** window and the **3D B-Spline Projection** window. A user creates a NURBS curve in the **2D NURBS Curve** window with right-clicks to add control points (Figure 3(a)). Initially, each control point has weight 1 and the curve is a B-spline. A user selects a control point with a left-click and uses left-drag to change its position. The vertical slide is for curve tracing. The lower-right corner has two buttons to zoom in and out the **3D B-Spline Projection** window, and a button to turn on and off the display of the grid in the **2D NURBS Curve** window.



(a) (b)

Figure 3: Windows of NURBSvis

The **3D B-Spline Projection** window shows the relation between 3D B-spline curve and 2D NURBS curve (Figure 3(b)). Since initially the created curve is a B-spline, it is identical to the projection NURBS curve in the $w = 1$ plane. This window supports trackball type rotation for a user to see the relation clearly and easily.

The weight of a selected control point can be changed using the slide in the lower-left corner of the **2D NURBS Curve** window, and the new weight is shown above the slide. If the weight of a control point is not 1, the curve becomes a NURBS curve. As the weight changes, the **3D B-Spline Projection** window shows the corresponding point (xw, yw, w) moving into space. The space curve in red is a 3D B-spline, and its projection 2D NURBS curve in $w = 1$ is in blue. There are lines connecting control points in $w = 1$

and their corresponding 3D points, and there is also a line between $\mathbf{C}(u)$ (*i.e.*, the point on the NURBS curve) and $\mathbf{C}^w(u)$ (*i.e.*, the point on the 3D B-spline curve).

De Boor's algorithm is one of the most important algorithms in B-splines study [1]. It takes a $u \in [0,1]$ and computes the corresponding point on a B-spline curve. Since $\mathbf{C}(u)$ is the projection of $\mathbf{C}^w(u)$, an application of de Boor's algorithm to the 4D B-spline yields $\mathbf{C}^w(u)$, and the projection of all computation steps to $w = 1$ gives de Boor's algorithm for the NURBS curve. Figure 4 shows this computation and the de Boor net. In this way, a user will be able to visualize the relationship between the B-spline version and the NURBS version of de Boor's algorithm. We found that this "proof-without-words" approach is quite effective in explaining the de Boor's algorithm for NURBS. Knot insertion and curve subdivision for NURBS can also be discussed the same way.

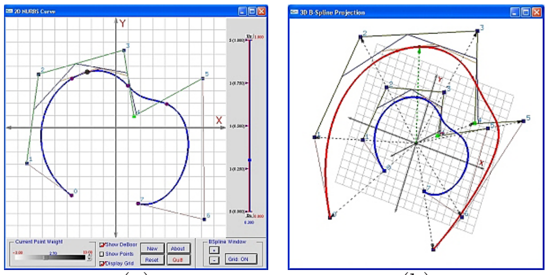

(a) (b)

Figure 4: De Boor's Algorithm

5. NURBS IMPORTANT PROPERTIES

After students have acquired background in projection, additional important properties are discussed. In fact, as long as a B-spline property is not metric related, it also holds for NURBS because a central projection, which is affine, changes metric measure but preserves the relative relation (*e.g.*, ordering and cross-ratio). Two properties that are important to both B-splines and NURBS are discussed: the *strong convex hull* property and *local modification* property. The strong convex hull property of a B-spline curve of degree p states that the curve segment on $[u_i, u_{i+1})$, lies in the convex hull defined by $p+1$ control points $\mathbf{P}_{i-p}$, ..., $\mathbf{P}_i$. This property provides an efficient way of locating a curve segment and guarantees that a selected curve segment or the whole B-spline curve lies in a predictable region. Because the 4D "lifted" B-spline curve satisfies the strong convex hull property and because central projections preserve convexity, the 3D NURBS curve also satisfies this property.

The local modification property states that the B-spline basis function $N_{i,p}(u)$ is non-zero on $[u_i, u_{i+p+1})$. Since $N_{i,p}(u)$ is the coefficient of $\mathbf{P}_i$, if $\mathbf{P}_i$ changes, $N_{i,p}(u)\mathbf{P}_i$ also changes. Since $N_{i,p}(u)$ is non-zero on $[u_i, u_{i+p+1})$, the change of $N_{i,p}(u)\mathbf{P}_i$ only affects the segment on $[u_i, u_{i+p+1})$ and does not affect curve segments elsewhere. With this property, we know that changing the position of a control point only affects a portion of a B-spline curve and the modification is *local*. Thus, modifying control point $\mathbf{P}_i$ of a NURBS curve $\mathbf{C}(u)$ causes the "lifted" control point $w_i\mathbf{P}_i^h$ to change, which, in turn, changes the shape of $\mathbf{C}^w(u)$ on $[u_i, u_{i+p+1})$. Since this curve segment projects to the NURBS curve segment of $\mathbf{C}(u)$ on $[u_i, u_{i+p+1})$, the local modification property holds for NURBS curves.

Figure 5 shows a NURBS curve of degree 4 defined by control points $\mathbf{P}_0$, ..., $\mathbf{P}_{15}$. If $\mathbf{P}_{10}$ is moved from its top position to its new position near the bottom, only the curve segment on $[u_{10}, u_{15})$, shown in light color, is changed. Curve segments at both ends are not affected.

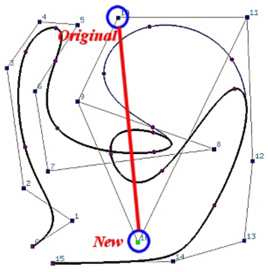


Figure 5: Modifying a Control Point

6. MODIFYING WEIGHTS

In addition to control points, knots and degree, a NURBS curve has weights, one for each control point, and providing one more degree of freedom for shape design. In fact, this simple extension makes NURBS curves more powerful than B-splines. Therefore, the impact of modifying the weight of a selected control point is a must-know property.

Suppose weight w_k of control point $\mathbf{P}_k$ is to be modified. If $w_k = 0$, the term $w_i\mathbf{P}_k$ disappears from the equation of the curve, and control point $\mathbf{P}_k$ has no contribution to the shape of the curve. What if w_k increases from 0 to infinity? Dividing the curve equation by w_k yields:

$$\mathbf{C}(u) = \frac{1}{(\sum_{i=0,i\neq k}^{n}(w_i/w_k)N_{i,p}(u)) + N_{k,p}(u)} \times \left[\left(\sum_{i=0,i\neq k}^{n}\frac{w_i}{w_k}N_{i,p}(u)\mathbf{P}_i\right) + N_{k,p}(u)\mathbf{P}_k\right]$$

Clearly, as w_k approaches infinity, w_i/w_k approaches zero and the equation has a limit $\mathbf{P}_k$. Hence, as w_k approaches infinity, the curve is "pulled" toward control point $\mathbf{P}_k$ and eventually passes through it. On the other hand, as w_k reduces to zero, the contribution of $\mathbf{P}_k$ also reduces and the curve is "pushed" away from $\mathbf{P}_k$. Eventually, when w_k reduces to zero, control point $\mathbf{P}_k$ has no contribution to the shape of the curve. But, which curve segment will be affected by this "pulling" and "pushing"? It can easily be analyzed with the projection concept. From the local modification property of B-splines, modifying w_k changes $w_k\mathbf{P}_k^h$, which, in turn, changes the curve segment of the 4D B-spline curve on $[u_k, u_{k+p+1})$. Thus, only the portion of the NURBS curve on $[u_k, u_{k+p+1})$ changes.

With DM2, a user may select a control point and change its weight. As the weight changes, the affected curve segment of the NURBS curve moves toward or away from the selected control point. Figure 6 shows a NURBS curve with control point $\mathbf{P}_5$ selected. The curve segment opposite to $\mathbf{P}_5$ is flat when $w_5 = 0$ because $\mathbf{P}_5$ has no contribution. As w_5 increases, the flat portion moves closer to $\mathbf{P}_5$. Figure 6 shows the curve segments corresponding to w_5 being 0, 0.1, 0.5, 1, 2, 4 and 10. When $w_5 = 10$, the curve is very close to $\mathbf{P}_5$. Moreover, DM2 allows a weight to be negative so that a user can see the impact of a negative weight. In general, when the negative weight is sufficiently small, the strong convex hull property fails. In other word, a portion of the affected curve segment will be outside of the convex hull defined by corresponding control points.

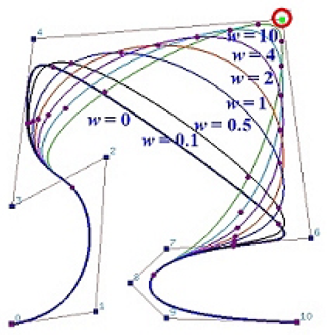


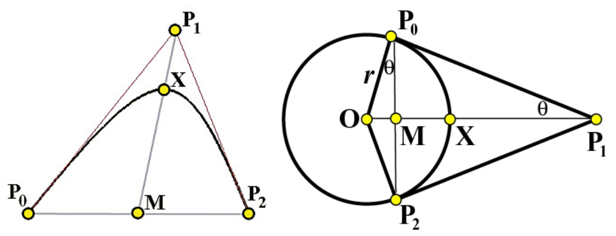
Figure 6: Modifying Weights

7. CONIC SECTIONS AND CIRCLES

We next answer the most basic question: how are conic sections and circles represented. All conic sections are degree 2 curves and can be represented by NURBS curves of degree 2. Thus, we need three control points $\mathbf{P}_0$, $\mathbf{P}_1$ and $\mathbf{P}_2$. Some simple calculations show that the weights of $\mathbf{P}_0$ and $\mathbf{P}_2$ can be set to 1, and only $\mathbf{P}_1$ needs a weight w. Under this condition, the B-spline basis functions are $N_{0,2}(u) = (1-u)^2$, $N_{1,2}(u) = 2u(1-u)$ and $N_{2,2}(u) = u^2$, and the NURBS curve of degree 2 has an equation as follows:

$$\begin{aligned}\mathbf{C}(u) &= \frac{1}{(1-u)^2 + 2u(1-u)w + u^2} \times \\ &\qquad \left[(1-u)^2\mathbf{P}_0 + 2u(1-u)w\mathbf{P}_1 + u^2\mathbf{P}_2\right]\end{aligned}$$

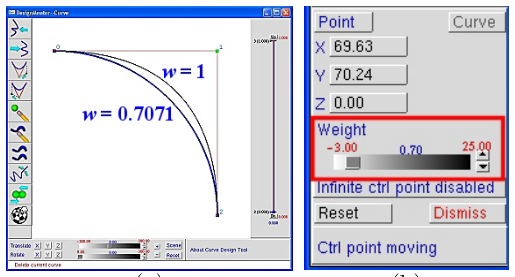
If we place the midpoint $\mathbf{M}$ of $\overline{\mathbf{P}_0\mathbf{P}_1}$ at the coordinate origin, and $\mathbf{P}_0$ and $\mathbf{P}_1$ on opposite sides of the x-axis, then $\mathbf{P}_0 = -\mathbf{P}_2$ (Figure 7(a)). Since $\mathbf{C}(0.5) = \frac{w}{1+w}\mathbf{P}_1$ from the above equation, we learn that the curve $\mathbf{C}(u)$ intersects $\overline{\mathbf{MP}}_1$ at $\mathbf{X} = \mathbf{C}(0.5)$ and $\overline{\mathbf{MX}}/\overline{\mathbf{MP}}_1 = w/(1+w)$. If $w = 1$, the curve is a Bézier curve that represents a parabola and $\mathbf{X}$ is the mid-point of $\overline{\mathbf{MP}}_1$ (*i.e.*, $\mathbf{X} = \frac{1}{2}\mathbf{P}_1$). A result from projective geometry implies that the NURBS curve is an ellipse if $w/(1+w) < 0.5$ (*i.e.*, $w < 1$), and a hyperbola if $w/(1+w) > 0.5$ (*i.e.*, $w > 1$). Consequently, with a NURBS curve of degree 2, one can set the weights of $\mathbf{P}_0$ and $\mathbf{P}_2$ to 1 and use the weight of $\mathbf{P}_1$ to define an ellipse, parabola, or hyperbola curve segment.



(a) (b)

Figure 7: Conics and Circles

How about circles? We learn from geometry $\overline{\mathbf{P}_0\mathbf{P}_1} = \overline{\mathbf{P}_2\mathbf{P}_1}$ (Figure 7(b)). What remains is to compute the weight w for $\mathbf{P}_1$. More precisely, we need to compute the ratio $\overline{\mathbf{MX}}/\overline{\mathbf{MP}}_1 = w/(1+w)$. Let the angle at $\mathbf{P}_1$ be 2θ, and the center and radius of the circle that is tangent to $\overline{\mathbf{P}_0\mathbf{P}}_1$ and $\overline{\mathbf{P}_2\mathbf{P}}_1$ at $\mathbf{P}_0$ and $\mathbf{P}_2$ be $\mathbf{O}$ and r. From $\triangle\mathbf{MOP}_0$, we have $\mathbf{OM} = r\sin(\theta)$ and $\mathbf{MX} = \mathbf{OX} - \mathbf{OM} = r - \mathbf{OM} = r(1-\sin(\theta))$. Since $\tan(\pi-\theta) = \overline{\mathbf{MP}}_1/\overline{\mathbf{MP}}_0$ from $\triangle\mathbf{MP}_0\mathbf{P}_1$ and $\overline{\mathbf{MP}}_0 = r\cos(\theta)$ from $\triangle\mathbf{MOP}_0$, we have $\overline{\mathbf{MP}}_1 = \overline{\mathbf{MP}}_0 \times \tan(\pi-\theta) = r\cos^2(\theta)/\sin(\theta)$. Hence, $\overline{\mathbf{MX}}/\overline{\mathbf{MP}}_1 = \sin(\theta)/(1+\sin(\theta)) = w/(1+w)$ and $w = \sin(\theta)$. If the angle at $\mathbf{P}_1$ is $2\theta = \pi/3 = 60°$, we have $w = \sin(\pi/6) = \frac{1}{2}$ and $\mathbf{X}$ is located at 1/3 of the distance from $\mathbf{M}$ to $\mathbf{P}_1$. If the angle at $\mathbf{P}_1$ is $\pi/2 = 90°$, then $w = \sin(\pi/4) = \sqrt{2}/2$. Figure 8(a) shows a DM2 example. The angle at $\mathbf{P}_1$ is $\pi/2$ and $w = \sqrt{2}/2 \approx 0.7071$. A user may use the slide near the bottom in the **Coords Window** (Figure 8(b)) to modify the weight of the selected control points. The parabola with $w = 1$ is also shown.



(a) (b)

Figure 8: A Circular Arc

Several circular arcs can be strung together to form a NURBS representation of a circle. Figure 9(a) shows the inscribed circle of an equilateral triangle. It is defined by 7 control points $\mathbf{P}_0, \ldots, \mathbf{P}_6 = \mathbf{P}_0$ ($n = 6$). Except for $\mathbf{P}_1$, $\mathbf{P}_3$ and $\mathbf{P}_5$ that have weight $\frac{1}{2}$, all other control points have weights 1. This NURBS curve of degree 2 has knots 0, 0, 0, 1/3, 1/3, 2/3, 2/3, 1, 1, 1. A circle can also be inscribed in a square. The circle has four circular arcs as shown in Figure 9(b). This NURBS circle of degree 2 is defined by 9 control points $\mathbf{P}_0, \ldots, \mathbf{P}_8 = \mathbf{P}_0$ ($n = 8$). The weights of $\mathbf{P}_1$, $\mathbf{P}_3$, $\mathbf{P}_5$ and $\mathbf{P}_7$ are $\sqrt{2}/2$ and the weights of the remaining are 1. This curve has knots 0, 0, 0, 1/4, 1/4, 1/2, 1/2, 3/4, 3/4, 1, 1, 1. See [1, 3] for the details.

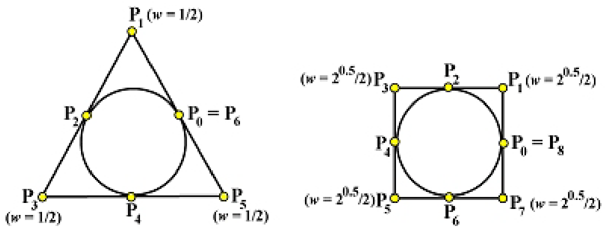
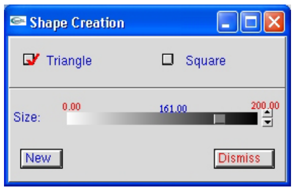


Figure 9: Complete Circles

When DM2 is asked to generate a circle, a small window appears (Figure 10(a)) for a user to choose the equilateral triangle version or square version and use the slide to set a radius. The desired circle with center at the origin is shown on-the-fly as the radius changes (Figure 10(b)).

8. CROSS-SECTIONAL DESIGN

Why are circles necessary? There are two reasons: (1) a circle is the simplest curve and (2) circles are used fre-



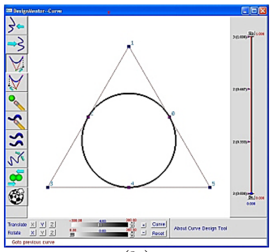
(a)

(b)

Figure 10: DM2 Circle Generation

quently (*e.g.*, in generating surfaces of revolution). DM2 supports a special surface design technique for generating commonly used surfaces, the cross-sectional design [5]. In cross-sectional design, a user specifies a *profile* curve and a *trajectory* curve so that the former will follow the latter to sweep out a surface. The result, in most cases, is a NURBS surface although both curves are in general B-splines.

Suppose we wish to design a vase shape. The first step is to design a B-spline profile curve as shown in Figure 11(a). Because this is a surface of revolution, the trajectory curve is a circle. DM2 generates this trajectory circle automatically. Once this circle, represented as a NURBS curve, is in hand, the revolving process involves the determination of all control points based on the circle representation. Figure 11(b) shows the wireframe version of generated vase in which the circles and their control points are clearly show, and Figure 11(c) shows the rendered result.

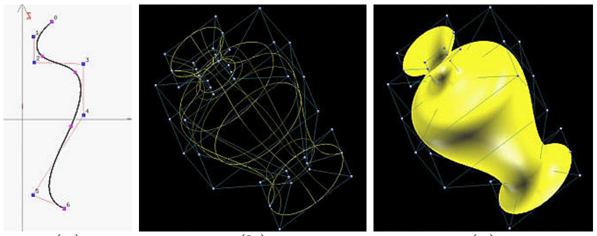

(a) (b) (c)

Figure 11: A Surface of Revolution

Circles may also be used as profile curves. A user may select a number of circles with various size (Figure 12(a)) for the cross-sectional system module of DM2 to compute a NURBS surface that contains all of them (Figure 12(b)). A surface that “interpolates” a set of curves is referred to as a *skinned* surface. Details are given in [5].

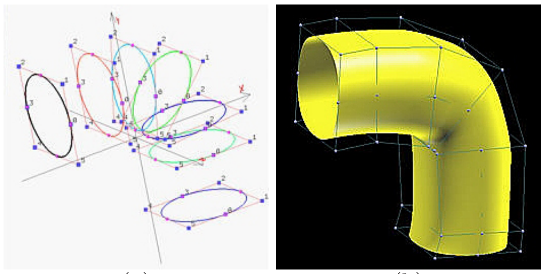

(a) (b)

Figure 12: A Skinned Surface

9. CONCLUSIONS

We have presented our approach of teaching the fundamentals of NURBS to undergraduate students in an elective course **Introduction to Computing Geometry**. In this course, we spend two weeks on B-splines followed by one week on NURBS. Student reactions in the past seven years have been very positive. Students especially like DesignMentor because it helps them understand the concepts and visualize the algorithms. In a breadth-first course, one may survey important concepts and use DM2 and NURBSvis to demonstrate the inner-working of important algorithms and to practice curve and surface design skills. Preliminary course evaluation results using pre- and post- tests and attitude survey were published in [1, 2]. The success and effectiveness of our materials and DesignMentor are also justified by the number of adaptations. There are many universities world-wide using our materials. A partial list includes MIT, Ohio State, Technische Fachhochschule Berlin, University of Alaska, University of Alberta, University of Manchester, University of Melbourne, University of Paris-South and Verona University. There are more than 2,500 downloads from our site and many universities have their own regional download servers. Moreover, the more than 44,000 visitors to our tutorial site, most from off campus, also demonstrated the usefulness of our materials.

Since the use of NURBS has become a basic design tool in virtually all graphics systems and is widely used in many interdisciplinary areas, it is the time for computer science educators to seriously consider incorporating this important topic into a typical curriculum. We hope this paper may serve as a starting point. We are finalizing DM2 for public release and are continually developing DesignMentor to support more features. Interested readers may find more about our work, web-based textbook, user guides, DesignMentor and other tools, and future announcement at `www.cs.mtu.edu/~shene/NSF-2`.

10. REFERENCES

[1] J. L. Lowther and C.-K. Shene, Teaching B-Splines Is Not Difficult!, *ACM 34th Annual SIGCSE Technical Symposium*, 2003, pp. 381–385.

[2] J. L. Lowther and C.-K. Shene, Computing with Geometry as an Undergraduate Course: A Three-Year Experience, *ACM 32nd Annual SIGCSE Technical Symposium*, 2001, pp. 119–123.

[3] C.-K. Shene, *Introduction to Computing with Geometry Notes*, available at `www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/`

[4] R. Wolfe, A Syllabus Survey: Examining the State of Current Practice in Introductory Computer Graphics Courses, *Computer Graphics*, Vol. 33 (1999), No. 1 (February), pp. 32–33.

[5] Y. Zhao, Y. Zhou, J. L. Lowther and C.-K. Shene, Cross-Sectional Design: A Tool for Computer Graphics and Computer-Aided Design Courses, *29th ASEE/IEEE Frontiers in Education*, Vol. II (1999), pp. (12b3-1)-(12b3-6).