Curve and Surface Interpolation and Approximation: Knowledge Unit and Software Tool *

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 describes a knowledge unit and the use of a software tool, DesignMentor, for teaching a very challenging topic in computer graphics and visualization, namely: curve and surface interpolation and approximation. Topics include global and local interpolation, global approximation, and curve network interpolation. For the past six years, a junior-level course has successfully used this approach.

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, Interpolation, Approximation

Keywords

B-Splines, Interpolation and Approximation

1. INTRODUCTION

Curve and surface construction is an important topic in computer graphics, computer-aided design, and visualization courses. In addition to Bézier, B-spline and NURBS curves and surfaces, interpolation and approximation provide another level of flexibility [6]. In many situations such as surface reengineering and facial movement animation, a designer specifies a set of data points that describes a shape (*e.g.*, the head of a dinosaur model), and seeks a surface that contains the data points (*i.e.*, interpolation). Interpolation is also important in computer animation. An animator specifies a number of key camera positions and interpolates them with a B-spline curve to generate a camera path. While interpolation can produce a curve/surface that contains the given data points, it may oscillate or wiggle its way through every point. Approximation can overcome this problem so that the curve/surface still captures the shape of the data points without containing all of them. Due to its importance, B-splines, interpolation and approximation have also started to have a more important role in graphics textbooks [1, 3, 8].

It is difficult to teach interpolation and approximation because of the necessary mathematical requirements. With our tool DesignMentor Version 2 or DM2, we are able to present this challenging topic in a knowledge unit for our junior-level elective course **Introduction to Computing with Geometry** [5]. We believe interpolation and approximation can also be used in traditional computer graphics or advanced computer graphics courses. In what follows, Section 2 discusses the placement of this knowledge unit in our course, Section 3 presents the differences between the interpolation methods discussed in a numerical methods course and the B-spline approach, and Section 4 shows how to find parameters from a given set of data points. Building upon this basic knowledge, Section 5 and Section 6 present global and local interpolation methods, respectively, Section 7 describes global approximation, and Section 8 provides a discussion of Gordon's method for interpolating a curve network. In each section, the basic concepts and algorithms are presented, followed by the use of DM2 to demonstrate, visualize and compare results. Then, Section 9 discusses a mini-project for this knowledge unit, Section 10 documents some of our experiences in teaching this unit six times, and, finally, Section 11 has our conclusions.

2. COURSE STRUCTURE AND MERIT

"Interpolation/Approximation" is a knowledge unit in our elective course **Introduction to Computing with Geometry**. This course is for students in computer science, mathematics and engineering [5]. It is taught in a computer-equipped classroom and DM2 is used for students to practice and visualize important concepts and properties. In this way, students acquire skills and knowledge gradually with formidable mathematics hidden within DM2. Students learn how to approach geometric problems, representations, and algorithms for Bézier, B-spline and NURBS curves and surfaces [2, 4, 9, 10], followed by cross-sectional design [11]. The interpolation and approximation unit normally takes two weeks near the end of a semester.

*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.

Topics of this unit include parameter selection, global and local interpolations, global approximation, and curve network interpolation. These topics were selected because they are **(1)** the most fundamental; **(2)** simple enough to learn and practice for those students who acquired knowledge of B-spline, calculus and linear algebra; and **(3)** the most widely used skills that can be found in computer graphics, geometric modeling, computer vision, and visualization textbooks, and in all good graphics systems.

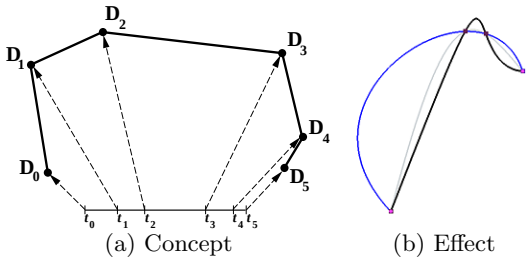
We do not ask students to master algebraic derivations. Instead, we expect our students to be aware of the existence of interpolation and approximation techniques, understand the fundamentals, be able to write programs based on existing algorithms, and recognize the advantages and disadvantages of various methods. Hence, materials are presented in an intuitive and learning-by-doing way with an emphasis on experimenting and visualizing important concepts and algorithms. Students see the effect of each algorithm, compare different algorithms without using formidable mathematics, and practice design skills. Consequently, learning interpolation and approximation is no longer a difficult task.

3. WHAT ARE THE DIFFERENCES?

Traditional interpolation methods (*e.g.*, Lagrange's methods) differ from B-spline methods. **First**, traditional methods use a polynomial $y = p(x)$ where any two data points cannot have equal x-coordinates; however, B-splines use a parametric form. **Second**, in traditional methods the degree of the polynomial is a function of the number of data points, whereas the degree of an interpolating B-spline is an input, permitting the use of lower degree curves/surfaces (*e.g.*, 2 or 3). **Third**, the input to traditional methods is a set of points (x_i, y_i), where x_i is the parameter value corresponding to y_i, whereas the B-spline method only requires a set of points. **Fourth**, traditional methods are global in that changing one data point affects the whole curve; however, local B-spline interpolation restricts the impact to the vicinity of that point.

4. IT ALL BEGINS WITH A SET OF GOOD PARAMETERS

Given a set of $m+1$ *data points* $\mathbf{D}_k$ $(0 \le k \le m)$ and a degree p, we seek a B-spline curve $\mathbf{C}(u)$ that passes through *all* points in the given order. Since parameters are not input, we have to find a set of parameters $t_0 = 0 < t_1 < \cdots < t_m = 1$ so that the interpolating B-spline curve can be "fixed" at these values (*i.e.*, $\mathbf{D}_k = \mathbf{C}(t_k)$ for all k) as in Figure 1(a).



(a) Concept (b) Effect

Figure 1: Parameter Selection

Four methods are presented in class: *equidistant*, *centripetal*, *chord-length*, and *universal*. Each method has its own convincing merit and drawbacks, and the resulting B-spline may be very different. In Figure 1(b), the curves with dark and light colors are computed with the equidistant and centripetal methods, respectively, while the other is obtained with the chord-length method. We pay special attention to the latter because it is easy to understand and perform well when there is no better alternatives. This method computes the parameter t_k with $t_k = d_k/d_m$, where $d_k = \sum_{i=1}^{k} |\mathbf{D}_i - \mathbf{D}_{i-1}|$. Then, a knot vector is generated from the chosen parameters by a simple moving average.

With the help of DM2, we ask students to specify data points, choose a parameter computation method, generate an interpolating B-spline, and compare the results. Thus, students can quickly learn the impact of parameters on a curve *before* they learn interpolation algorithms!

5. GLOBAL INTERPOLATION

The simplest method is *global interpolation*. Since the degree p is an input and the knot vector is computed from the parameters t_k's, the only missing part for the interpolating B-spline curve is $m+1$ control points. Let these unknown control points be $\mathbf{P}_i$ $(0 \le i \le m)$, and let the B-spline curve be $\mathbf{C}(u) = \sum_{i=0}^{m} N_{i,p}(u)\mathbf{P}_i$, where $N_{i,p}(u)$ is the i-th B-spline basis function of degree p [4]. Hence, we have

$$\mathbf{D}_k = \mathbf{C}(t_k) = \sum_{i=0}^{m} N_{i,p}(t_k)\mathbf{P}_i \qquad \text{for all } 0 \le k \le m$$

Let $\mathbf{N} = [N_{i,p}(t_k)]_{(m+1)\times(m+1)}$, $\mathbf{D} = [\mathbf{D}_k]_{(m+1)\times 1}$ and $\mathbf{P} = [\mathbf{P}_i]_{(m+1)\times 1}$. The above equation becomes $\mathbf{D} = \mathbf{N}\cdot\mathbf{P}$. Since $\mathbf{D}$ and $\mathbf{N}$ are known, solving for $\mathbf{P}$ yields all control points and hence the interpolating B-spline curve of degree p.

The surface case is similar. Given a grid of data points $\mathbf{D}_{k,l}$ $(0 \le k \le m$ and $0 \le l \le n)$ and degrees p and q, we seek a B-spline surface $\mathbf{S}(u,v)$ of degree (p,q) such that $\mathbf{D}_{k,l} = \mathbf{S}(s_k, t_l)$ for $0 \le k \le m$ and $0 \le l \le n$, where s_k's and t_l's are the chosen parameter values. We have

$$\mathbf{D}_{k,l} = \mathbf{S}(s_k, t_l) = \sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(s_k)N_{j,q}(t_l)\mathbf{P}_{i,j}$$

where $\mathbf{P}_{i,j}$'s $(0 \le i \le m$ and $0 \le j \le n)$ are unknown control points. The above can be rewritten as follows:

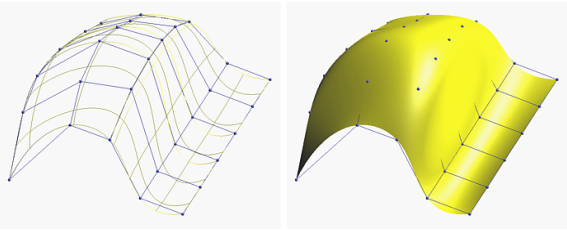
$$\mathbf{D}_{k,l} = \sum_{i=0}^{m} N_{i,p}(s_k)\left[\sum_{j=0}^{n} N_{j,q}(t_l)\mathbf{P}_{i,j}\right] = \sum_{i=0}^{m} N_{i,p}(s_k)\mathbf{R}_{i,l}$$

where

$$\mathbf{R}_{i,l} = \sum_{j=0}^{n} N_{j,q}(t_l)\mathbf{P}_{i,j}$$

Solving for $\mathbf{R}_{i,l}$'s from the first equation using the $\mathbf{D}_{k,j}$'s and $N_{i,p}(s_k)$'s, and solving for $\mathbf{P}_{i,j}$'s from the second using $\mathbf{R}_{i,l}$'s and $N_{j,q}(t_l)$'s yields control points $\mathbf{P}_{i,j}$'s and hence the desired B-spline surface. Therefore, a global interpolating B-spline surface is computed by applying the curve interpolation algorithms twice! Figure 2 shows interpolating a 5×6 grid with a B-spline surface of degree $(3,3)$.

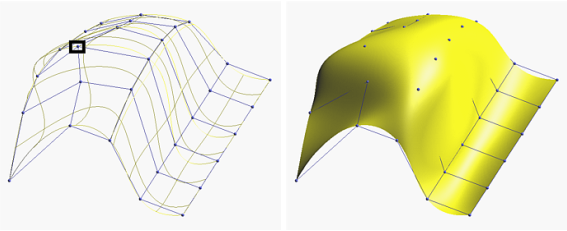
While changing the position of a control point of a B-spline curve only affects a portion of the curve [4], changing the



(a) Isoparametric Lines (b) Rendered Surface

Figure 2: Global Interpolation

position of a data point affects the whole curve. Figure 3 shows the impact of changing the position of the marked data point. The neighborhood of that data point is changed drastically, and the impact even propagates to the right as shown by the isoparametric lines.



(a) Isoparametric Lines (b) Rendered Surface

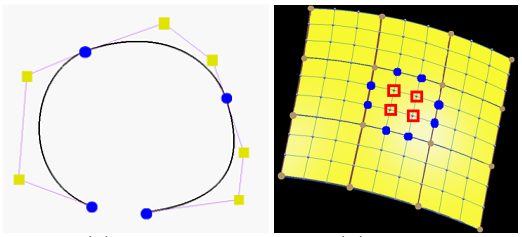
Figure 3: Changing the Position of a Data Point

DM2 is used to create curves and surfaces with different parameters, and to visualize the impact of changing the position of a data point. DM2 also helps in understanding the drawbacks of "global" interpolation.

6. LOCAL INTERPOLATION

Local interpolation techniques are introduced to overcome the unpleasant result in Figure 3 and other problems. Given $m+1$ data points $\mathbf{D}_k$ $(0 \le k \le m)$, a Bézier curve of degree 3 is constructed for each pair of adjacent data points such that two adjacent Bézier curves are joined with C^1-continuity. Since only two end-points are known, two internal control points must be found. In Figure 4(a), the four blue dots are the given data points, while the six yellow squares are the constructed internal control points. The surface case is more involved. A Bézier surface of degree $(3, 3)$ requires $16 = 4 \times 4$ control points as shown in Figure 4(b). From the four corner data points, eight additional "border" control points are constructed (dots) as in the curve case, two on each border. The four internal control points (squares) are computed from the given data and "border" control points. Since the theory is too complex, only a computation scheme is discussed in class.

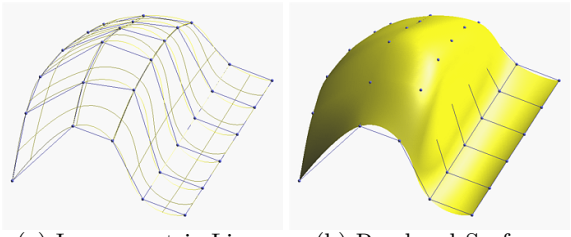
After all necessary control points are constructed, we have a number of Bézier surfaces joining along their common boundary curves with C^1-continuity. The desired Bézier surface is obtained by removing control points along common boundaries. Figure 5 shows the result of applying local interpolation to the same data points used for global interpolation. Compared the isoparametric lines in Figure 2(a) and Figure 5(a), we see that the local version is closer to the



(a) Curve (b) Surface

Figure 4: Local Interpolation Construction

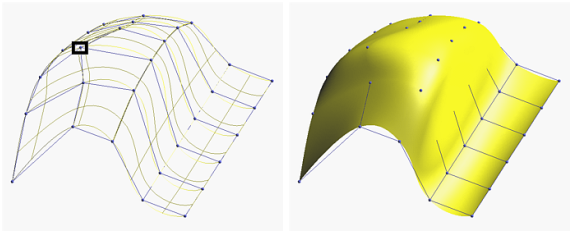
data grid than the global one.



(a) Isoparametric Lines (b) Rendered Surface

Figure 5: Local Interpolation

In general, given the same degree, results from local and global interpolations are not very different. The real advantage of using local interpolation is for *local control*. Figure 6 is the results of changing the position of the marked point. The impact is not so drastic as in Figure 3(a). Moreover, local interpolation does not produce a large bulge, and the right-end of the surface is unaffected.



(a) Isoparametric Lines (b) Rendered Surface

Figure 6: Changing the Position of a Data Point

We start with the discussion of the drawbacks of global interpolation as a motivation for local interpolation. Then, DM2 is used to show construction of a surface using local interpolation, and to demonstrate the *local control* property. However, students are asked to interpolate three points by hand calculation to reinforce their understanding and compare the differences between global and local interpolations.

7. GLOBAL APPROXIMATION

In many applications, requiring a curve/surface to contain all data points may be too restrictive, and a curve/surface that can *follow* the shape of the data points closely may be sufficient. This leads to the discussion of *global approximation*. Note that a B-spline curve defined by the data points and a given degree also *approximates* and follows the shape of the data points; however, this is not a good solution.

Given a set of $m+1$ data points $\mathbf{D}_k$ and a degree p, we seek a B-spline curve $\mathbf{C}(u)$ of degree p that approximates the data points in the least square sense. Let this B-spline curve be $\mathbf{C} = \sum_{i=0}^{m} N_{i,p}(u)\mathbf{P}_i$, where n is a user selectable value and $\mathbf{P}_i$'s are $n+1$ unknown control points. For each parameter t_k, $|\mathbf{C}(t_k) - \mathbf{D}_k|$ is the distance between the "computed" point and the actual point. A good approximation method should minimize the sum of squared "error" distance:

$$f(\mathbf{P}_0, \ldots, \mathbf{P}_n) = \sum_{k=0}^{m} |\mathbf{C}(t_k) - \mathbf{D}_k|^2$$

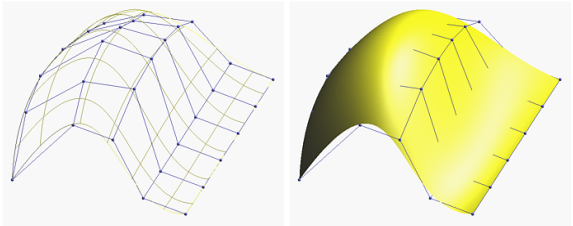
The $\mathbf{P}_i$'s that minimize $f()$ are the desired control points. To ensure a unique solution, we set $\mathbf{D}_0 = \mathbf{P}_0$ and $\mathbf{D}_m = \mathbf{P}_n$ to "clamp" the curve at both ends. Then, computing and setting partial derivatives to 0's yields a system of linear equations whose solution is the desired result.

The surface case is similar. Given a set of $(m+1) \times (n+1)$ data points $\mathbf{D}_{k,l}$ $(0 \le k \le m$ and $0 \le l \le n)$ and a desired degree (p, q), we seek a B-spline surface $\mathbf{S}(u, v)$ that approximates the $\mathbf{D}_{k,l}$'s in the sense of least square. More precisely, find control points $\mathbf{P}_{i,j}$ $(0 \le i \le e$ and $0 \le j \le f)$ such that the following is minimized:

$$f(\mathbf{P}_{i,j}\text{'s}) = \sum_{k=0}^{m} \sum_{l=0}^{n} |\mathbf{S}(s_k, t_l) - \mathbf{D}_{k,l}|^2$$

where e and f are user selectable values, and s_k's and t_l's are parameters. A simplified method is presented in class. Curve approximation is applied to each column l of $\mathbf{D}_{k,l}$ $(0 \le k \le m)$ to obtain a set of $(e+1) \times (n+1)$ "intermediate" control points $\mathbf{R}_{i,l}$ $(0 \le i \le e)$. Then, curve approximation is applied again to each row i of $\mathbf{R}_{i,l}$'s to obtain the final control points $\mathbf{P}_{i,j}$ $(0 \le j \le f)$. Thus, an approximating B-spline surface is constructed with $(n+1)+(m+1)$ curve approximations. Although the resulting surface may not be optimal, it is simple enough for students to understand.

Figure 7 is the result of applying surface approximation to the same data points used before. The surface does not contain all points, and looks "smoother" than those constructed with global and local interpolations.

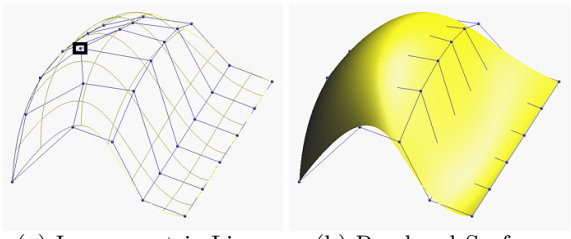


(a) Isoparametric Lines (b) Rendered Surface

Figure 7: Approximation

This approximation method is also global, but the impact of changing the position of a data point is not as drastic as that of global interpolation as shown in Figure 8.

Just as in the global interpolation case, students are asked to interpolate and approximate the same set of data points, compare the difference to see the impact of relaxing the requirement of "passing through every point," and visualize the "global" impact of moving one data point.

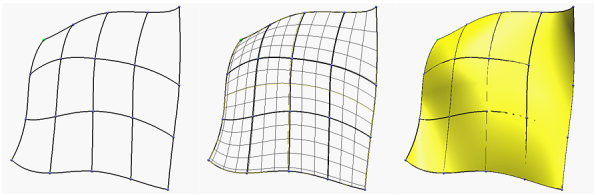


(a) Isoparametric Lines (b) Rendered Surface

Figure 8: Changing the Position of a Data Point

8. CURVE NETWORK INTERPOLATION

Sometimes designers may prefer to use a curve network rather than a set of data points. A *curve network* consists of two families of B-spline curves, $\mathbf{F}_k(u)$ $(0 \le k \le K)$ and $\mathbf{G}_l(v)$ $(0 \le l \le L)$ such that the intersection of two curves, one from each family, consists of exactly one point (Figure 9(a)). Let the intersection point of curve $\mathbf{F}_k()$ and curve $\mathbf{G}_l()$ be $\mathbf{D}_{l,k} = \mathbf{F}_k(u_l) \cap \mathbf{G}_l(v_k)$ for some u_l and v_k. An interpolating surface for this curve network is a B-spline surface $\mathbf{S}(u, v)$ such that $\mathbf{F}_k(u)$'s are isoparametric curves in one direction (*i.e.*, $\mathbf{S}(u, v_k) = \mathbf{F}_k(u)$) and $\mathbf{G}_l(v)$'s are isoparametric curves in the other (*i.e.*, $\mathbf{S}(u_l, v) = \mathbf{G}_l(v)$).



(a) Network (b) Lines (b) Surface

Figure 9: Gordon Surfaces

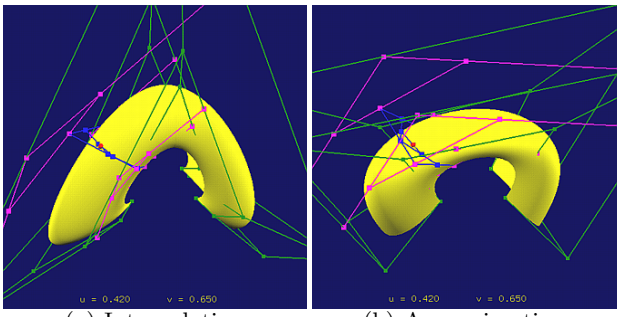
Gordon's elegant solution to this problem is discussed in class. It consists of three simple surfaces. The first $\mathbf{S}_1(u, v)$ contains all $\mathbf{G}_l(v)$'s, the second $\mathbf{S}_2(u, v)$ contains all $\mathbf{F}_k(u)$'s, and the third $\mathbf{S}_3(u, v)$ contains all intersection points $\mathbf{D}_{l,k}$'s. The first two are skinned surfaces that can be constructed with the skinning technique [11], which is discussed before interpolation and approximation. The third one can be constructed with global or local interpolation. Then, the desired surface is simply $\mathbf{S}(u, v) = \mathbf{S}_1(u, v) + \mathbf{S}_2(u, v) - \mathbf{S}_3(u, v)$. Note that there are an infinite number of surfaces that contain the given curve network, and the constructed Gorgon surface is only a possible, maybe non-optimal, solution. Figure 9(b) and Figure 9(c) are the constructed Gordon surface for the curve network in Figure 9(a) in isoparametric and rendered forms, respectively.

Students are asked to use DM2 to design a curve network, and interpolate it with a Gordon surface using their own implementation. The curve network is extracted from a set of isoparametric curves of a B-spline surface. Then, students are asked to compare the constructed Gordon surface with the original using a "goodness-of-fit" measure. In this way, they not only practice curve network design, but also learn

to investigate the concept of goodness-of-fit to determine if the resulting surface is good enough.

9. A MINI-PROJECT

Interpolation/approximation is a difficult topic. To make sure students understand the subject, we gave them a comprehensive and challenging two-week mini-project. They are provided with a linear equation solver and a scaled-down version of DM2 for implementing global interpolation and approximation using various parameters chosen from a predefined menu. Also, they must use de Boor's algorithm to trace the constructed B-spline surfaces and display the de Boor nets correctly. Figure 10(a) and Figure 10(b) show the interpolation and approximation screen shots of a typical student program, respectively.



(a) Interpolation (b) Approximation

Figure 10: Student Mini-Project

10. EXPERIENCE

This interpolation and approximation unit was taught six times and was evaluated with pre- and post- tests and an attitudinal survey. Twenty-two questions were used to assess student learning level of course topics in a scale from 0 (none) to 4 (excellent). Two questions were about curve and surface interpolation and approximation. The curve (*resp.*, surface) question has pre- and post- test averages 0.7 and 1.9 (*resp.*, 0.5 and 2.0) with standard deviation 0.7 and 0.8 (*resp.*, 0.6 and 0.8). The Kolmogorov-Smirnov test shows that the difference between pre- and post- test is significant. Thus, students did learn interpolation and approximation. The gain is smaller in the curve case because students thought this topic is similar to what they learn in a numerical methods class and rated themselves higher in the pre-test. We did not expect a large gain because this is a difficult topic. Judged by exam scores and the mini-project, we are confident that our students did know how to implement global interpolation and global approximation provided proper tools are given. This is exactly our expectation. Even more encouraging is that many students felt that doing such a challenging project improved their programming skills and the understanding of the subject.

11. CONCLUSIONS

We have presented knowledge unit, a mini-project, and the use of DM2 for teaching important concepts, algorithms, and design skills in interpolation and approximation using B-splines. Combined with our web-based tutorial, students not only learn the fundamentals in an intuitive and learning-by-doing way, hut also practice design skills interactively. While many may argue that presenting such a difficult topic is virtually an impossible mission in a computer graphics course, as demonstrated by our work we strongly believe, with proper tools, good course materials and presentation skills, it can be done well. DesignMentor is under continual development and will soon have a full set of features. Interested readers may find more about our work, including our web-based textbook, user guides, DesignMentor and other tools, and future announcements, at the following site:

`www.cs.mtu.edu/~shene/NSF-2`

Acknowledgment

The interpolation and approximation component of DesignMentor was implemented by Yan Zhou, our former graduate students. Former graduate student Budirijanto Purnumo started the implementation of DM2.

12. REFERENCES

[1] Samuel R. Buss, *3-D Computer Graphics*, Cambridge University Press, 2003.

[2] John Fisher, John Lowther and Ching-Kuang Shene, If You Know B-Splines Well, You Also Know NURBS! *ACM 35th Annual SIGCSE Technical Symposium*, 2004, pp. 343-347.

[3] J. D. Foley, A. van Dam, S. K. Feiner and J. F. Hughes, *Computer Graphics: Principles and Practice*, Second Edition, Addison-Wesley, 1990.

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

[5] 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.

[6] L. Piegl and W. Tiller, *The NURBS Book*, Springer-Verlag, 1995.

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

[8] P. Shirley, *Fundamentals of Computer Graphics*, A K Peters, 2002.

[9] Y. Zhao, J. L. Lowther and C.-K. Shene, A Tool for Teaching Curve Design, *ACM 29th Annual SIGCSE Technical Symposium*, 1998, pp. 97–101.

[10] Y. Zhao, Y. Zhou, J. L. Lowther and C.-K. Shene, Teaching Surface Design Made Easy, *ACM 30th Annual SIGCSE Technical Symposium*, 1999, pp. 222–226.

[11] 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).