# Physics Based Geometric Design

**Hong Qin**[1]

Department of Computer Science, University of Toronto,

10 King's College Road, Toronto, Ontario, M5S 1A4

Email: qin@cs.toronto.edu

**Abstract**

Geometric modeling has proved to be crucial to computer graphics and computer aided geometric design (CAGD). During the past several decades, numerous geometric formulations have been proposed for a large variety of geometric modeling applications. In this paper, we survey the diversity of shape representations ranging from the primitive polynomial to the sophisticated Non-Uniform Rational B-Spline (NURBS). We demonstrate that, among various geometric representations, NURBS have become an industrial standard primarily because of their many superior properties. By reviewing commonly used design paradigms such as interpolation, approximation, interactive modification and variational optimization, we can also show that these conventional geometric design techniques are generally awkward when designers are confronted by complex, real-world objects. This is primarily because they only allow free-form primitives such as NURBS to be indirectly manipulated through numerous degrees of freedom (DOFs). To overcome the disadvantages of this indirect process, we summarize the prior work of physics-based modeling and review dynamic NURBS (D-NURBS) as a physics-based generalization of geometric NURBS for shape design. D-NURBS can unify the features of the industry-standard NURBS geometry with the many demonstrated conveniences of interaction within the new physics-based design framework. We demonstrate that D-NURBS can not only serve as a basis for the future research of physics-based geometric design but also become readily appropriate for a large variety of important applications in graphics, vision, and scientific visualization.

**Keywords:** CAGD, NURBS, Physics-based models, Finite Elements.

[1] The author has moved to the following new address:
Department of Computer & Information Science & Engineering
University of Florida
Gainesville, FL 32611–6120
U.S.A.

# 1 Introduction

Geometric modeling is concerned with the mathematical representation of geometric entities and their application to the design of objects by computer. Geometric modeling is crucial to a variety of fields from CAGD to computer graphics, virtual reality, medical imaging, and computer vision. During the past several decades, numerous geometric formulations ranging from the primitive polynomial to sophisticated rational splines have been proposed for a large variety of geometric modeling applications.

Geometric computation requires that shapes be represented in a precise and unambiguous manner and that the mathematical formulation be flexible for free-form design. Geometric objects can be represented with simple parametric polynomials, piecewise non-rational or rational splines, domain-less recursive subdivision surfaces, and implicit functions. NURBS [109], among frequently used representations, have become an industrial standard for geometric design. This is primarily because they provide a unified mathematical formulation for representing not only free-form curves and surfaces like B-splines, but also standard analytic shapes such as conics, quadrics, and surfaces of revolution.

Many design techniques have been developed for CAGD to achieve various design and manufacturing requirements. First, designers can specify geometric entities by either interpolating or approximating a set of regular data points, scattered data points, or boundary curves. Second, designers can indirectly manipulate the DOFs of the underlying geometric formulation to achieve interactive shape design. Third, through the process of cross-sectional design, they can design surfaces by specifying generator curves. Finally, users can also utilize constraint-based optimization methods to determine free parameters. In general, industrial design requirements can be posed as both quantitative functional and qualitative aesthetic criteria. The latter ones are usually satisfied through the use of optimization techniques.

Traditional geometric design is a kinematic process. It require the designer to achieve desired shapes through the indirect manipulation of many DOFs. This conventional shape modification process can often be clumsy and laborious, because geometry is both abstract and static. Pure geometry does not have the intuitive behavior of precomputational, physical design media such as modeling clay. The design and manufacture process, however, requires a mechanism which can accomplish the interactive modification of geometric information both efficiently and precisely. Unfortunately, traditional design methodology can not offer us such a dynamic and interactive framework for time-varying requirements. To bridge the gap between geometry and the requirements of design and manufacturing, we have developed D-NURBS and physics-based shape design, overcoming the design difficulties of the conventional geometric modeling. As a result, we examine how to associate physical dynamics with abstract geometry. We can demonstrate that D-NURBS, built upon industry-standard geometric NURBS, provide a unified scheme for various shape representations as well as design paradigms.

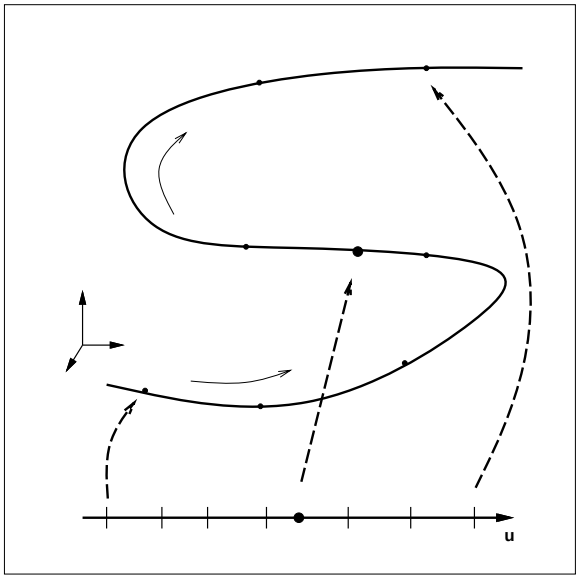



Figure 1: The three-dimensional parametric curve and its parametric domain.

### 1.1 Overview

In Section 2, we review geometric modeling methods. From Section 3 to Section 6, we survey the progression of computer-aided design from the purely geometric to the physics-based paradigms. We start with a review of geometric design (Section 3) and discuss some of its limitations (Section 4). We then review variational design which addresses some of these limitations in Section 5. In Section 6, we summarize prior work in physics-based design which may be viewed as a generalization of variational design. The advantages of the physics-based modeling approach will serve as the motivation for D-NURBS which will be developed in Section 7 for geometric design. Section 8 concludes the paper.

## 2 Geometric Representation

In this section, we review frequently used geometric representations including primitive polynomials, parametric splines, subdivision forms, algebraic functions, and NURBS (more detailed materials can be found in [35, 14, 4, 83, 32]).

### 2.1 Parametric Curves

A parametric curve is defined as a function of one parameter $u$ over an interval domain $I$: $\mathbf{c} = \mathbf{c}(u)$ (see Fig. 1). The parametric formulation supports straightforward geometric computation. It can be easily extended to higher dimensions.

### 2.1.1 Polynomial Forms

From the computational viewpoint, finite-degree polynomials are ideal for representing and approximating geometric entities because polynomials are closed under differentiation, integration, and arithmetic operations. The best known polynomial representation is the monomial form of degree $n$:

$$\mathbf{f}(u) = \mathbf{a}_0 + \mathbf{a}_1 u + \ldots + \mathbf{a}_n u^n. \tag{1}$$

Despite its simplicity, the computational advantage of (1) is counteracted by the fact that the $\mathbf{a}_i$ in (1) do not provide any intuitive insight into the curve shape beyond the point $u = 0$.

In addition, a polynomial curve can be defined by a set of $n+1$ coefficients along with a knot sequence $t_0, \ldots, t_n$

$$\mathbf{f}(u) = \mathbf{a}_0 L_0^n(u) + \ldots + \mathbf{a}_n L_n^n(u) \tag{2}$$

where $L_i^n(u)$ are Lagrange polynomials of degree $n$ satisfying $L_i^n(u_j) = \delta_{ij}$, and $\delta$ is the Kronecker delta. It is apparent that the polynomial curve interpolates all the $\mathbf{a}_i$. However, the interpolating curve exhibits unwanted oscillation, especially for high-order polynomials. Therefore, both the monomial and the Lagrangian forms are not suitable for the construction of piecewise smooth curves.

Alternatively, a modeler can use Hermite polynomials $H_i^n(u)$

$$\begin{aligned}\mathbf{f}(u) = {} & \mathbf{D}_0\mathbf{a}_0 H_0^n(u) + \mathbf{D}_1\mathbf{a}_0 H_1^n(u) + \ldots + \mathbf{D}_{(n-1)/2}\mathbf{a}_0 H_{(n-1)/2}^n(u) \\ & + \mathbf{D}_{(n-1)/2}\mathbf{a}_1 H_{(n+1)/2}^n(u) + \ldots + \mathbf{D}_1\mathbf{a}_1 H_{(n-1)}^n(u) + \mathbf{D}_0\mathbf{a}_1 H_n^n(u)\end{aligned} \tag{3}$$

where $\mathbf{D}_j\mathbf{a}_i = \mathbf{f}^{(j)}(i)$, $i = 0, 1$, $j = 0, \ldots, (n-1)/2$, and $n$ is odd. Even-degree Hermite polynomials require a different treatment (see [35] for the details). Although (3) is geometrically intuitive, the Hermite interpolant has a severe drawback—designers must be provided with high-order derivative information in order to achieve the desired order of continuity across the adjacent curve spans. Unfortunately, derivative information is neither available nor easily derived in most applications.

To overcome prior difficulties, de Casteljau and Bezier independently developed a curve formulation, now called Bezier curves, based on Bernstein polynomials $B_i^n(u)$:

$$\mathbf{f}(u) = \mathbf{a}_0 B_0^n(u) + \ldots + \mathbf{a}_n B_n^n(u) \tag{4}$$

where

$$B_i^n(u) = \left( \begin{array}{c} n \\ i \end{array} \right) (1-u)^{n-i} u^i, i = 0, 1, \ldots, n.$$

Bezier points $\mathbf{a}_0, \ldots, \mathbf{a}_n$ form the Bezier polygon. Unlike the coefficients in (1)–(3), Bezier control points are viewed as design tools to generate a free-form curve. The control polygon is both simple and easy to use mainly because Bernstein basis functions have many important features such as: (i) partition of unity, (ii) positivity, and (iii) recursive evaluation. Feature (i) ensures Bezier points are invariant under affine transformation. Features (i) and (ii) guarantee that the curve segment resides within the convex hull or Bezier polygon and that the Bezier curve has a variation diminishing property. Feature (iii) serves as a basis for the well known de Casteljau algorithm which is used to evaluate a curve through repeated linear interpolation. This recursive evaluation has proven to be both computationally efficient and stable (note that it is slower than Horner's rule). Other important properties of Bernstein polynomials include degree elevation and subdivision (see also [14] for the details about the differentiation and integration of Bernstein polynomials and Bezier curves).

### 2.1.2 Parametric Splines

The above modeling techniques are relatively simple and robust. The only available means for modeling complex shapes, however, is to increase the degree of the basis functions. Higher order polynomials afford greater flexibility at the expense of more complex geometric computation, but they are not quite satisfactory because they tend to introduce spurious undulations between interpolating points. Such oscillations are not implied by the data points themselves. Moreover, designers have to enforce extra continuity requirements where two adjacent polynomials meet. This, in general, complicates the design task tremendously. One alternative is to use piecewise polynomials, or splines.

The mathematical theory of splines was originated by Schoenberg [96] in 1946. A spline curve is a piecewise univariate function satisfying a set of continuity constraints. The spline can be categorized in terms of interpolation/approximation schemes or global/local schemes. In 1972 de Boor proposed B-splines from the standpoint of approximation theory [26]. During 1973 and 1974, Riesenfeld and Gordon proposed B-splines as a powerful and proper generalization of Bernstein polynomials for free-form curve design [91, 48].

A B-spline curve is the combination of a set of piecewise polynomial functions with $n+1$ control points $\mathbf{p}_i$

$$\mathbf{c}(u) = \sum_{i=0}^{n} \mathbf{p}_i B_{i,k}(u) \tag{5}$$

where $u$ is the parametric variable and $B_{i,k}(u)$ are B-spline basis functions. Assuming basis functions of degree $k-1$, a B-spline curve has $n+k+1$ knots $t_i$ in a non-decreasing sequence: $t_0 \leq t_1 \leq \ldots \leq t_{n+k}$.

B-spline basis functions are defined recursively as

$$B_{i,1}(u) = \begin{cases} 1 & \text{for } t_i \leq u < t_{i+1} \\ 0 & \text{otherwise} \end{cases},$$

with

$$B_{i,k}(u) = \frac{u - t_i}{t_{i+k-1} - t_i} B_{i,k-1}(u) + \frac{t_{i+k} - u}{t_{i+k} - t_{i+1}} B_{i+1,k-1}(u).$$

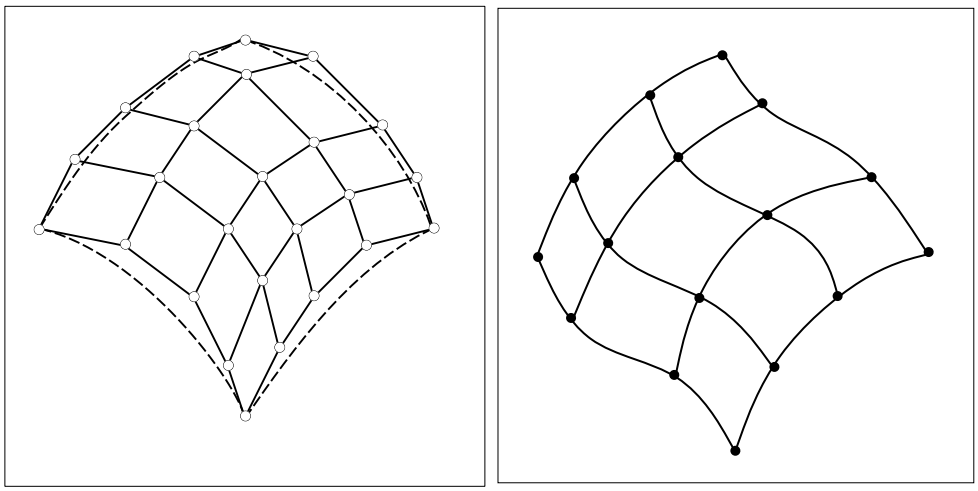
The parametric domain is $t_{k-1} \leq u \leq t_{n+1}$. Similar to Bezier points in (4), de Boor points $\mathbf{p}_i$ form the control polygon. Many attractive and important properties follow immediately including the following:

- Partition of unity, positivity, and recursive evaluation for B-spline basis functions.
- B-splines intrinsically provide various smoothness criteria. The curve is $C^{k-j}$ continuous at a $j$-fold knot.
- B-splines include Bezier curves as their subset. Composite Bezier curves equivalent to a B-spline curve can be easily obtained by means of multiple knot insertion at every old knot until all knots are of multiplicity $k$.
- In contrast to Bezier curves, B-splines have a local control property—the adjustment of one control point affects the curve only locally.
- Invariance under linear transformation.
- Strong convex hull and variation diminishing properties.

A recursive evaluation algorithm that can efficiently evaluate B-spline curves was invented by Mansfield, de Boor, and Cox [26, 22]. B-spline derivative and integration formulas are discussed in detail in [14]. One important advantage of B-splines over Bezier polynomials is that, in order to increase the potential flexibility of B-splines, a modeler need not raise the degree of the basis functions. Instead, this goal can easily be achieved through knot insertion with which the number of B-spline DOFs are increased while the order of the basis function remains unchanged.

## 2.2 Parametric Surfaces

Since the pioneering work by Coons and Bezier in the late 50's, CAGD has been dominated by the theory of rectangular surface patches. Two predominant modeling schemes are control point based (Fig. 2(a)) and transfinite based (Fig. 2(b)) representations. Recently, many different approaches have been derived for representing smooth surfaces. Today, parametric surfaces can be categorized as follows:



(a) (b)

Figure 2: Control point based and transfinite representations.

**Directrix-Generator** The surface is swept out by sliding a generator curve across a set of directrices. Three essential constituents include directrices (longitudinal curves), correspondence rules, and generators. To make the generated surface satisfy continuity requirements, the constituents must be sufficiently smooth. Typical examples are conic lofting surfaces and surfaces of revolution.

**Transfinite Patch** The blending surface interpolates a network of given curves (e.g., the Coons patch).

**Multiple Patch** The surface is a collection of "smaller" patches (e.g., the Bezier patch, and n-sided patches).

**Carpet Method** The surface is a set of multiple patches, and adjacent patches are constrained to maintain certain continuity conditions (e.g., B-splines).

**New Representations** Typical examples are recursive subdivision surfaces, irregular B-spline-like surfaces, multivariate B-splines, and algebraic patches.

### 2.2.1 Tensor-Product Surfaces

Among various surface forms, the tensor-product surface is the simplest and most popular scheme because it is a straightforward generalization of its curve predecessor. Given two sets of basis functions $\{F_i(u)\}$ and $\{G_j(v)\}$ for curves, a general tensor-product surface can be constructed as a linear combination of the composite set of basis functions $\{F_i(u)G_j(v)\}$ as follows:

$$\mathbf{f}(u,v) = \sum_i \sum_j \mathbf{c}_{i,j} F_i(u) G_j(v). \tag{6}$$

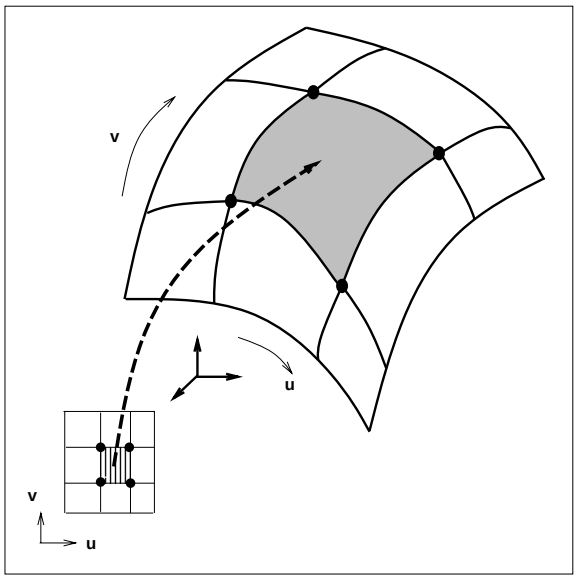



Figure 3: Rectangular surface and its parametric domain.

Fig. 3 illustrates a tensor-product surface and its parametric domain. Because of the special construction of (6), the properties of tensor-product surfaces are usually determined by those of the underlying curve schemes. Tensor-product surfaces of monomial and Lagrangian forms can easily be derived in light of (1) and (2), respectively. Analogous to (4), the tensor-product Bezier surface can be defined as

$$\mathbf{f}(u,v) = \sum_i \sum_j \mathbf{b}_{i,j} B_i^m(u) B_j^n(v). \tag{7}$$

Like Bezier curves, the many nice properties of Bezier surfaces include the following:

- All isoparametric curves are Bezier curves.
- Partial derivatives along the boundary can be formulated explicitly.
- The strong convex hull property.
- Degree elevation and subdivision are carried out along all rows or all columns.
- Affine invariance.

Bezier surface evaluation can be implemented by successively applying the de Casteljau algorithm on relevant rows and columns (note that the evaluation order is irrelevant). In analogy to Bezier curves, complex surfaces can be constructed by stitching together a number of Bezier patches smoothly.

Likewise, a tensor-product B-spline surface is defined over the parametric variables $u$ and $v$ as

$$\mathbf{s}(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{p}_{i,j} B_{i,k}(u) B_{j,l}(v). \tag{8}$$

A B-spline surface has $(m+1)(n+1)$ control points $\mathbf{p}_{i,j}$. Assuming basis functions along the two parametric axes of degree $k-1$ and $l-1$, respectively, the number of knots is $(m+k+1)(n+l+1)$. The non-decreasing knot sequence is $t_0 \leq t_1 \leq \ldots \leq t_{m+k}$ along the $u$-axis and $s_0 \leq s_1 \leq \ldots \leq s_{n+l}$ along the $v$-axis. The parametric domain is $t_{k-1} \leq u \leq t_{m+1}$ and $s_{l-1} \leq v \leq s_{n+1}$. If the end knots have multiplicity $k$ and $l$ in the $u$ and $v$ axis respectively, the surface patch will interpolate the four corners of the boundary control points.

B-spline surfaces generalize Bezier patches. Their primary properties include the following:

- All isoparametric curves are B-spline curves.
- Local control.
- Knot insertion increases the DOFs while keeping the degree of basis functions unchanged.
- Strong convex hull.
- B-splines include Bezier surfaces as their subset. The piecewise Bezier representation of B-splines can be obtained through multiple knot insertion.
- The B-spline control polygon converges to the B-spline surface as the number of knots increases.
- B-spline control points permit interactive manipulation.

### 2.2.2 Transfinite Surfaces

Unlike the previous control point based schemes, the transfinite approach allows surfaces to be determined by a set of prescribed boundary curves. Before the advent of computers, transfinite-based surfaces were generated with the lofting technique. One of the most popular transfinite surfaces is the Coons patch which can be obtained by blending four boundary curves either bilinearly or bicubically. The bilinearly blended Coons patch can be concisely formulated with the Boolean sum

$$(P)\mathbf{f} = (P_1 \oplus P_2)\mathbf{f} = (P_1 + P_2 - P_1 P_2)\mathbf{f} \tag{9}$$

where two operators $P_1$, and $P_2$ are defined in terms of Lagrange polynomials in (2) and four boundary curves

$$(P_1)\mathbf{f} = \mathbf{f}(0,v)L_0^1(u) + \mathbf{f}(1,v)L_1^1(u)$$

$$(P_2)\mathbf{f} = \mathbf{f}(u,0)L_0^1(v) + \mathbf{f}(u,1)L_1^1(v)$$

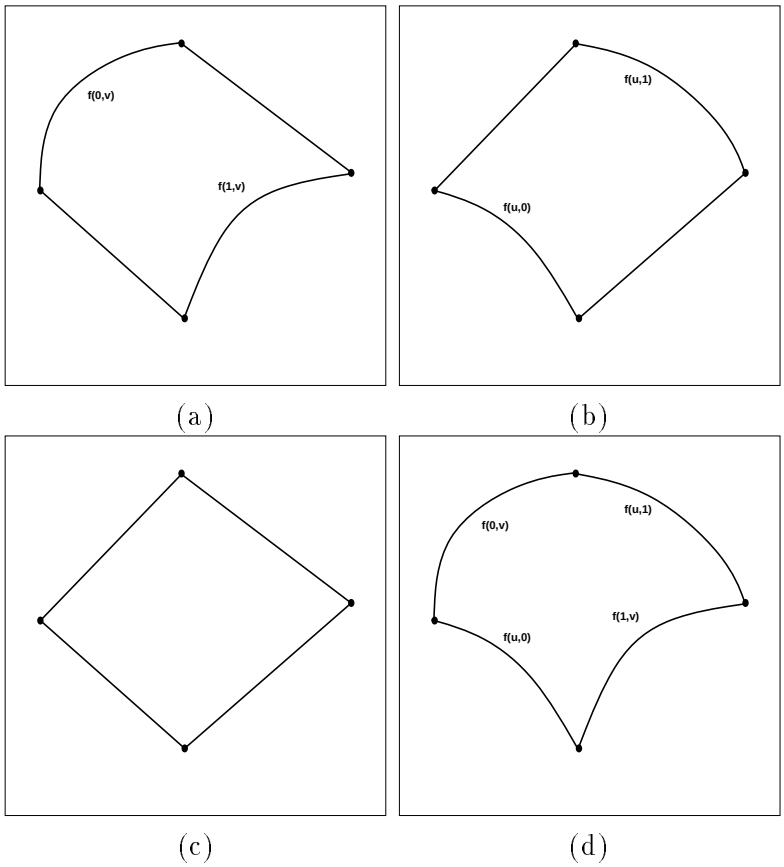



Figure 4: The boolean-sum construction of a bilinearly blended Coons patch.

Fig. 4 illustrates a Boolean sum construction of a bilinearly blended Coons patch. It is apparent that a bilinearly blended Coons patch interpolates four boundary curves. When abutting Coons patches, however, bilinearly blended Coons patches are only $C^0$ across their boundaries. If cross-boundary derivatives are known, bicubically blended Coons patches can be defined using cubic Hermite polynomials (see [14] for the details). Likewise, bicubically blended Coons patches interpolate both boundary curves and prescribed boundary derivatives.

Theoretically, transfinite surfaces may produce high-order continuity across adjacent patches. In practice, however, high-order derivatives are often unavailable during the design process. The *ad hoc* and heuristic approach is to estimate the initial derivative values based on generating curves in the vicinity. Another design difficulty comes from the inconsistency of the cross-derivative (twist) which may influence local surface oscillations. To overcome twist incompatibility in Coons patches, Gregory replaced the incompatible constant twists with variable twists using a convex combination method, which is also known as *Gregory's square*.

Coons' contribution was very significant primarily because it stimulated the development of other surface representations. Generalizing the Coons patch, Gordon constructed a blending surface from a quadrangular network of compatible curves with Lagrange polynomials. The idea of blending was further

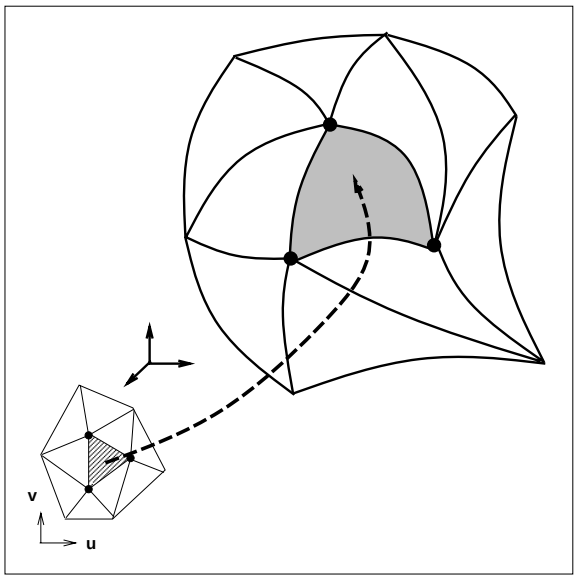



Figure 5: An irregular surface and its parametric domain.

carried over to transfinite interpolation of triangles [5].

Two major schemes have been developed for free-form curve/surface design. The Bezier/B-spline technique emphasizes geometric intuition, while the Coons/Gordon method is based on abstract algebraic concepts such as the Boolean sum. The two types of shape representations differ from each other dramatically in terms of their geometric nature. One apparent distinction is that the boundary curves of a B-spline surface must also be B-splines, whereas the Coons patch allows its boundary curves to be of arbitrary form.

### 2.2.3 Irregular Patches

Despite their simplicity, tensor-product surfaces have drawbacks. The underlying shape (and the parametric domain) of tensor-product surfaces must be "topologically" rectangular. However, when used for representing complex shapes, general $n$-sided patches are more desirable than rectangular patches. Fig. 5 shows a surface composed of triangular patches and its parametric domain.

Historically, de Casteljau considered triangular patches in the late 50's even before he defined tensor-product "Bezier" patches. Nevertheless, it was not until the 70's that triangular schemes started to be widely applied to CAGD. Farin constructed Bezier triangles based on the Bernstein form. Like the univariate Bezier curve and tensor-product Bezier surface, the Bezier triangle has many nice properties such as partition of unity, positivity, and recursive evaluation for basis functions. Detailed discussion about the formulation, derivative computation, degree elevation, subdivision, and continuity condition can be found in [14, 32]. Triangular Bezier patches have been used in various design applications. They can be used as Hermite interpolants which interpolate position and derivative information at vertices and across boundaries. Subsequently, other triangular interpolants have been derived [4]. The main techniques of

triangular patches can be characterized as transfinite, convex combination, and control point approaches.

Significant effort has also been devoted to the derivation of arbitrary $n$-sided patches. Generalizing triangular and tensor-product Bezier surfaces, Loop and DeRose defined S-patches over an arbitrary $n$-sided convex polygonal domain [57]. The key technique is functional composition which embeds a polygonal parametric domain into a higher dimensional simplex. The S-patch is obtained by restricting the Bezier simplex to the embedded polygonal domain. Despite its generalization over regular Bezier surfaces, the S-patch suffers from major drawbacks such as a complicated domain mapping formulation, a time-consuming evaluation algorithm, and the lack of a geometric interpretation for its control polygon.

Significant effort has also been devoted to the derivation of arbitrary n-sided patches. Generalizing triangular and tensor-product Bezier surfaces, Loop and DeRose defined S-patches over an arbitrary n-sided convex polygonal domain [57]. The key technique is functional composition which embeds a polygonal parametric domain into a higher dimensional simplex. The S-patch is obtained by restricting the Bezier simplex to the embedded polygonal domain. Despite its generalization over regular Bezier surfaces, the S-patch suffers from major drawbacks such as a complicated domain mapping formulation, a time-consuming evaluation algorithm, and the lack of a geometric interpretation for its control polygon.

One typical application of the S-patch is to fill a n-sided hole with $G^1$ continuity. Loop and DeRose proposed generalized B-spline surfaces of arbitrary topology based on their n-sided S-patch [58]. Although the technique generalizes biquadratic and bicubic B-spline surfaces, there is no straightforward closed-form formulation for this new representation. In addition, there are strict restrictions placing on the connectivity of the initial control mesh. The construction of the generalized B-spline surface is extremely complicated, and this procedure can only implemented indirectly through multiple steps. Another disadvantage of this B-spline-like scheme is that this algorithm does not provide a unified approach for quadratic and cubic B-splines. In either case, only $G^1$ surfaces are generated.

### 2.2.4 Triangular B-splines

Although widely used, regular B-spline surfaces are incapable of describing surfaces of arbitrary topology. Consequently, triangular B-splines [25] are emerging as a powerful new tool for geometric modeling. They are useful for modeling a broad range of complex objects defined over arbitrary, non-rectangular domains. Using triangular B-splines, designers can benefit from arbitrary parametric domains, non-degeneracy for multi-sided surfaces, and other important features.

The theoretical foundation of triangular B-splines lies in the multivariate simplex spline of approximation theory. Motivated by an idea of Curry and Schoenberg for a geometric interpretation of univariate B-splines, de Boor first presented a brief description of multivariate simplex splines [27]. Since then, their

theory has been explored extensively [62, 23, 24, 53]. The well-known recurrence relation of multivariate simplex splines was first proposed in [62]. Subsequently, Grandine gave a stable evaluation algorithm [49]. Dahmen and Micchelli presented a thorough review of multivariate B-splines [24]. From the point of view of blossoming, Dahmen, Micchelli and Seidel proposed triangular B-splines which are essentially normalized simplex splines [25].

The practical application of multivariate simplex splines is relatively underdeveloped compared to their theory, because of complicated domain partitioning and time-consuming algorithms for evaluation and derivative computation, especially for high dimensional and high order cases. However, it is possible to derive efficient algorithms for a low dimensional domain such as a plane and/or a low order polynomial such as a quadratic or a cubic. Traas discussed the applicability of bivariate quadratic simplex splines as finite elements and derived differentiation and inner product formulas [108]. Auerbach et al. use the bivariate simplex B-spline to fit geological surfaces through scattered data by adjusting the triangulation of the parametric domain in accordance with the data distribution [1]. Recently, the first experimental CAGD software based on the triangular B-spline was developed, demonstrating the practical feasibility of multivariate B-spline algorithms [42].

Triangular B-splines are ideal for geometric design applications because of their many nice properties such as lower polynomial degree and optimum global smoothness with high flexibility. First, their locally defined basis functions are nonnegative piecewise polynomials which sum to unity. Second, polynomial surfaces with degree $n$ can be $C^{n-1}$ continuous if their knots are in general positions. The designer can achieve various smoothness requirements through knot variation. For instance, in quadratic triangular B-splines, the six linearly independent bivariate basis functions are defined over convex hulls spanned by five knots. Making four of the knots collinear renders the corresponding basis function discontinuous along the line. Making three knots collinear leads to a discontinuity of the first derivative. Finally, triangular B-splines have the convex hull property and are affine invariant under standard geometric transformations. Since any piecewise polynomial with degree $n$ over a triangulation can be represented as a linear combination of triangular B-splines [50], they provide a unified representation scheme for polynomial models with arbitrary topology.

## 2.3 Subdivision Forms

Because the planar parametric domain is an open surface, it is almost impossible to model an object of arbitrary genus with a single non-degenerate B-spline surface. Thus, extra boundary continuity constraints must be enforced. Moreover, singularity seems to be inevitable in modeling real-world objects. Although trimming surfaces offer an alternative, it can destroy the compact and concise representation of spline

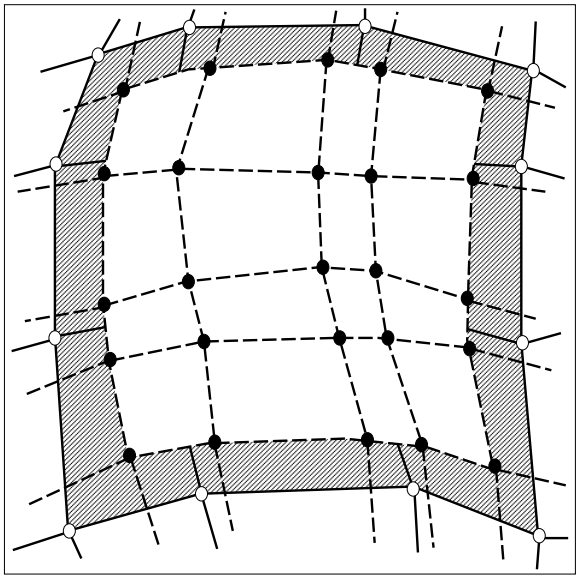


Figure 6: Recursive subdivision surface.

surfaces. The domain-less subdivision schemes (see Fig. 6) have proven to be very promising for modeling extremely complex objects.

In 1974, Chaikin described an efficient algorithm which permits the recursive subdivision of an initial polygon and makes a series of new polygons converge to a smooth curve. In 1978, Catmull and Clark generalized Chaikin's approach and constructed a smooth surface with arbitrary topology through the recursive subdivision of an arbitrarily shaped polyhedron [15]. This surface reduces to a $G^2$ B-spline surface except at a finite number of extraordinary points where only tangent plane continuity is achieved. The essential part of these new schemes is the subdivision rule which can guide the generation of an infinite set of consecutively refined meshes based on an initial polyhedron. The subdivision surface is the limit to which this infinite corner-chopping process converges. Note that, different subdivision rules determine distinct shapes of the converged smooth surface/curve.

The key advantage of subdivision surfaces is that smooth surfaces of arbitrary topology can be obtained based on topologically complex initial meshes. However, recursive subdivision surfaces lack parametric domains, which precludes their immediate pointwise evaluation and hence may limit the applicability of these schemes. In general, subdivision surfaces can be reduced to regular B-spline surfaces except at a finite number of singular points. Since the singular points need special treatment, subdivision surfaces are not equivalent to the previously described standard spline formulations. Instead, they are often referred to as B-spline-like surfaces. The behavior of those singular points are determined by eigenvalues of a set of matrices [29].

Due to the lack of explicit parameterization, the fast evaluation of subdivision surfaces is restrained. Peters developed an algorithm which can generate a bivariate $C^1$ surface with an explicit parameterization

from an irregular control point mesh [76]. His approach produces a refined mesh of points through the use of the subdivision procedure. The new control point mesh is then used for generating a $C^1$ surface consisting of a set of quadratic and cubic patches. The major advantage of this algorithm is that it combines the intuitive subdivision of the irregular meshes with low-degree parameterization. The key constituents of this technique are (i) refining the initial mesh and generating the control points of box spline, (ii) converting the box spline surface into Bezier form, and (iii) filling the remaining holes with cubic triangular patches. Although open or closed surfaces of general topological structure can be obtained, this algorithm is complicated and *ad hoc*. Furthermore, a large number of patches are necessary to describe a complex shape.

## 2.4 Algebraic Functions

Although the most popular representation in CAGD is the parametric form, the traditional representation of geometric entities (in classical analytic geometry) is the implicit function. While parametric forms are well suited for shape editing and rendering, implicit forms are useful for point membership classification. Fig. 7 shows that a two-dimensional implicit function $f(x, y) = 0$ is obtained through a planar cross-section of the three-dimensional scalar function $z = f(x, y)$. The most simple form for implicit functions is the power basis expression of degree $n$

$$\sum_{i,j,k,i+j+k=n} a_{ijk} x^i y^j z^k = 0. \tag{10}$$

Like the monomial form in (1), coefficients in (10) provide neither direct geometric interpretation nor intuitive insight into the underlying shape because the power basis implicit function is algebraic, and not geometric. It is equally difficult to predict the influence on the shape caused by the coefficient perturbation. Also there are no convenient tools for the intuitive shape control of this algebraic surface.

It can be shown that the set of implicit algebraic surfaces is actually larger than that of rational surfaces. This set is also closed under certain geometric operations. Every rational parametric curve/surface can be represented by an implicit algebraic equation, but not vice versa. Despite their representation power, implicit algebraic equations have shortcomings from the perspective of computational geometry. First, digitizing and rendering an implicit function is always difficult. Second, the derived shape may have separate components which are not indicated in the empirical data, therefore, extra polynomial constraints are necessary to ensure no singularities or self-intersections.

Despite these shortcomings, algebraic functions can be used for free-form modeling. Typical applications include forcing an algebraic surface to interpolate a set of points or spatial curves, and using piecewise algebraic patches to form a complex shape satisfying certain continuity requirements across patch boundaries. Sederberg discussed the modeling techniques for cubic algebraic surfaces [97, 98]. Hoffmann systematically

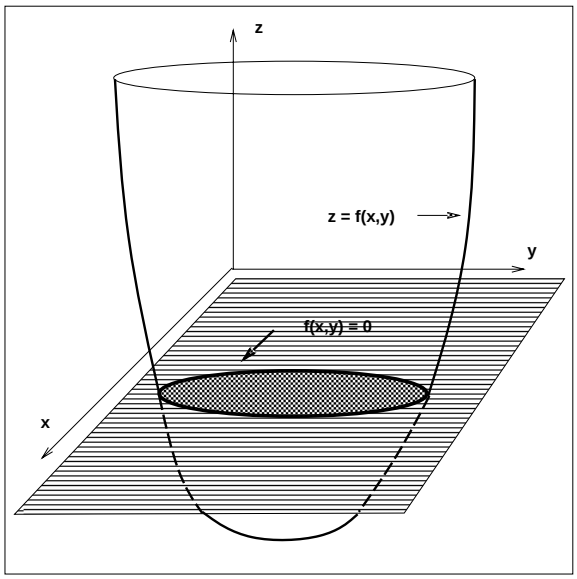



Figure 7: The construction of a two-dimensional implicit function.

reviewed the implicit function including the implicitization, parameterization, and the parametric/implicit conversion in CAGD [52]. Bajaj and Ihm presented an efficient algorithm to implement Hermite interpolation of low-degree algebraic surfaces with $C^1$ or $G^1$ continuity [2]. The interpolation is essentially reduced to a homogeneous linear system where the unknowns are coefficients of the algebraic surface. Note that, neither point nor curve interpolation is an attractive mechanism for defining an implicit surface because it is difficult for designers to predict the surface behavior beyond interpolating curves and points.

The benefits of using implicit functions are due to their low degree and computational efficiency. It is highly desirable to permit the interactive and direct manipulation of implicit algebraic surfaces. A piecewise algebraic surface patch can be defined with trivariate barycentric coordinates using a reference tetrahedron, and a regular lattice of control points and weights can be associated with this bounding tetrahedron. Consider a tetrahedron with noncoplanar vertices $\mathbf{v}_{n000}$, $\mathbf{v}_{0n00}$, $\mathbf{v}_{00n0}$, and $\mathbf{v}_{000n}$. Let $(r, s, t, u)$ denote the local barycentric coordinates of a point $\mathbf{p}$ in the tetrahedron. By definition, we have

$$\mathbf{p} = r\mathbf{v}_{n000} + s\mathbf{v}_{0n00} + \mathbf{v}_{00n0} + u\mathbf{v}_{000n},$$

where $r + s + t + u = 1$. Then, we define a set of $(n + 1)(n + 2)(n + 3)/6$ control points $\mathbf{p}_{ijkl}$ such that

$$\mathbf{p}_{ijkl} = \frac{i\mathbf{v}_{n000} + j\mathbf{v}_{0n00} + k\mathbf{v}_{00n0} + l\mathbf{v}_{000n}}{n},$$

where $i, j, k, l \geq 0$, and $i + j + k + l = n$. A weight $w_{ijkl}$ is assigned to each control point. The algebraic

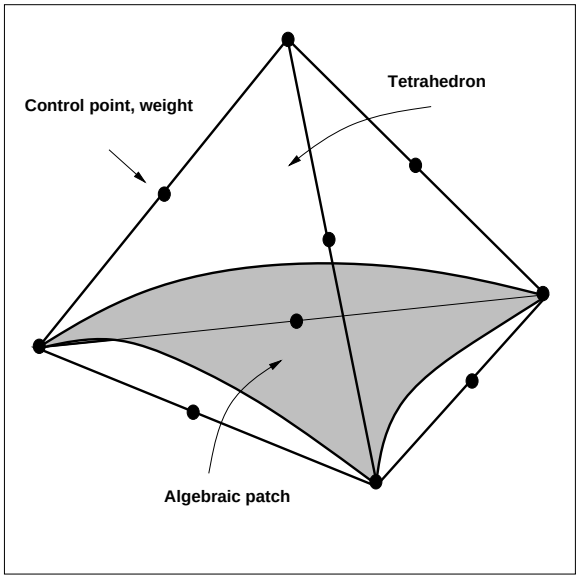


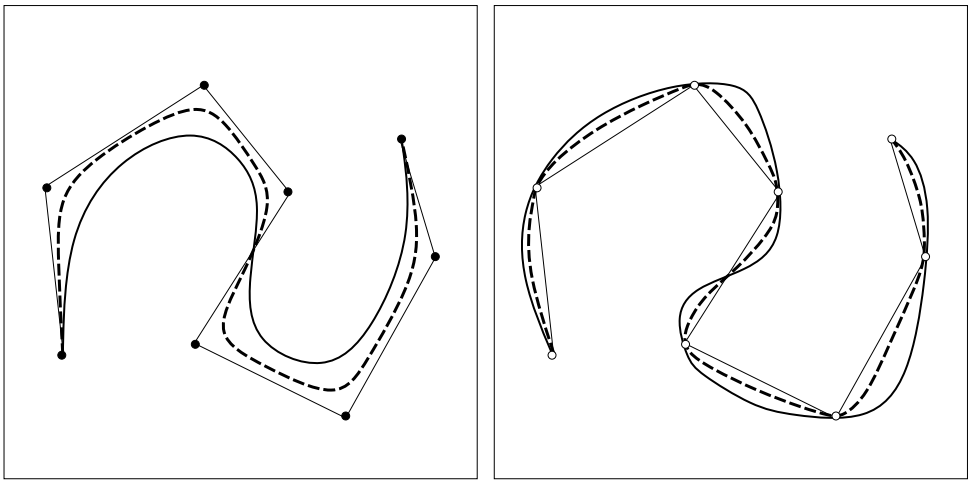

Figure 8: A quadratic algebraic patch.

patch inside the tetrahedron can be formulated using Bernstein-Bezier basis functions as

$$\sum_i \sum_j \sum_k \sum_{l=n-i-j-k} w_{ijkl} \frac{n!}{i!j!k!l!} r^i s^j t^k u^l = 0, \tag{11}$$

where $i, j, k$, and $l$ are non-negative, and $(r, s, t, u)$ represents the local barycentric coordinates of arbitrary vertices on this algebraic patch. Fig. 8 shows the construction of a quadratic algebraic patch in which $n = 2$. The key advantage is that control points and weights provide a meaningful way to control the shape of the patch. Note that, unlike weights of rational splines, the weights of algebraic patches are not independent. Like Bezier curves and surfaces, algebraic patches have some nice properties such as local control of control points and weights, boundary interpolation, gradient control, and self-intersection avoidance (see [97, 98, 2] for the details). Patches can be abutted smoothly for modeling complex shapes. By applying subdivision, additional degrees of freedom are generated, allowing patches to satisfy extra continuity conditions across the boundaries.

Implicit functions have been used in various applications in graphics and CAGD. Generalizing ordinary algebraic modeling methods, Muraki used the "blobby" models for volumetric data fitting [66]. Although this approach allows the automatic extraction of a symbolic shape description from the range data, it is too slow to be of practical importance. The convolution surface [11] is another example of graphical model representation with implicit functions, and this technique is based on skeletons. The convolution surface generalizes the "blobby" model because the continuous integral operator is used to replace the discrete summation of exponential-based operators. Thus, the shape of the convolution surface is smooth and blendings are well behaved. Dutta, Martin and Pratt [30] used piecewise Dupin's quartic cyclides for



(a) (b)

Figure 9: Tension effect on curve shape (a) the curve is generated by control polygon (b) the curve is generated through data interpolation.

free-form surface modeling and blending because of the low algebraic degree, rational parametric form, and simple and intuitive geometric parameters of cyclides.

### 2.5 Geometric and Variational Splines

The main task in CAGD is to smoothly construct complex shapes with a parsimonious number of curve spans or surface patches. For instance, piecewise Bezier curves can be used to construct complex curves satisfying certain continuity requirements at conjunction points. Ordinary parametric continuity, however, has proven to be unnecessarily restrictive. Recently, a weaker continuity condition, geometric continuity, has received a great of deal of attention (see [8, 9] for the details). By definition, two adjacent curves are said to have $G^n$ continuity at a joint if and only if they meet with $C^n$ continuity under arc-length parameterization. Obviously, $G^1$ means unit tangent vector continuity. $G^2$ implies curvature continuity in addition to $G^1$ continuity.

With geometric continuity, it is possible to introduce shape parameters independent of control points, which can provide designers more DOFs for shape design and control. It is always desirable to provide extra flexibility for geometric design. Recently, there has been a great deal of interest in the use of tension parameters as extra design handles beyond control vertices for piecewise polynomial (see Fig. 9 for tension effects on the curve shape). Note that, tension parameters act as a parametric rescaling because the parametric continuity is reduced into geometric continuity. Many special splines involving geometric continuity have been developed during the past twenty years.

The Beta-spline [7], in which adjacent curve spans satisfy geometric continuity, generalizes the uniform cubic B-spline. The uniformly shaped Beta-splines provide only two control parameters (bias and tension)

for the whole curve, and thus, no local control is provided. Nonetheless, this generality allows designers to control the geometric shape with shape parameters in addition to control points. In general, a tension increase will pull the curve towards the control point, and a bias increase will pull the curve span towards the corresponding line segment of the control polygon. In particular, if the bias is one, and the tension is zero, the Beta-spline is reduced to the cubic B-spline. The effect of bias and tension and the necessary $G^2$ continuity conditions are discussed in [47]. Furthermore, a continuously-shaped Beta-spline has been developed [7] where local control of bias and tension are obtained. Other forms of Beta-splines also exist. In [8, 9], $G^1$ and $G^2$ Beta-splines are shown to be equivalent to composite quadratic and cubic Bezier splines, respectively.

DeRose and Barsky discussed a new class of splines, geometrically continuous Catmull-Rom splines [28], which can be expressed as

$$\mathbf{c}(u) = \sum_i \mathbf{a}_i(u) W_i(u), \tag{12}$$

where each $\mathbf{a}_i(u)$ is constructed to interpolate the $k+1$ vertices $\mathbf{v}_i, \ldots, \mathbf{v}_{i+k}$. Unlike the previously discussed spline schemes, $\mathbf{a}_i(u)$ is a vector-valued interpolating function. As a result, geometrically continuous Catmull-Rom splines can either interpolate or approximate data points. It has both control vertices and shape parameters which can be used for shape manipulation. When used in applications, however, its control vertices and shape parameters are often transformed into the equivalent Bezier control polygon to take advantage of various efficient Bezier toolkits.

Many special splines have also been defined via variational forms. In 1974, Nielson generalized the exponential related spline under tension. He proposed the Nu-spline as a $G^2$ (continuous curvature) piecewise polynomial minimizing the following functional over $[t_0, t_n]$:

$$\int_{t_0}^{t_n} \left\| \mathbf{f}''(u) \right\|^2 du + \sum_{i=0}^{n} \nu_i \left\| \mathbf{f}'(u_i) \right\|^2, \tag{13}$$

subject to the interpolation $\mathbf{f}(t_i) = \mathbf{d}_i$ and necessary end conditions. Based on (13), Nielson further developed a geometric representation which allows convenient tension control while maintaining the global $G^2$ continuity [67]. This curve is composed of piecewise planar rational polynomials with tangent direction continuity and zero curvature at each end. Similar to the Nu-spline, it permits local control of tension parameters.

In 1985, Hagen proposed the Tau-spline [51] which is a quintic spline that maintains both curvature and torsion continuity by minimizing

$$\int_{t_0}^{t_n} \left\| \mathbf{f}^{(k)}(u) \right\|^2 du + \sum_{i=0}^{n} \sum_{j=1}^{k-1} \nu_{i,j} \left\| \mathbf{f}^{(j)}(u_i) \right\|^2 \tag{14}$$

subject to interpolatory constraints. Pottmann presented a special spline with tension control [86] which generalizes the Tau-spline and possesses third-order parametric continuity and a smooth torsion plot. Lasser explicitly formulated a Bezier representation for the Tau-spline [56]. Thus, many efficient algorithms of Bezier splines are also applicable to Tau-splines.

In 1986, Nielson extended his Nu-spline to the surface case [68]. The straightforward tensor-product generalization of Nu-spline is not very useful because the arbitrary $n + m$ tension parameters will affect the entire curve network. In contrast to ordinary piecewise splines, Nu-spline provides extra tension parameters. When used for various modeling applications, however, Nu-splines are often converted into piecewise Hermite polynomials to expedite spline evaluation.

Foley presented a weighted Nu-spline interpolant [40] which is the $C^1$ piecewise cubic polynomials by minimizing

$$\sum_{i=0}^{n-1} w_i \int_{t_i}^{t_{i+1}} \left\| \mathbf{f}''(u) \right\|^2 du + \sum_{i=0}^{n} \nu_i \left\| \mathbf{f}'(u_i) \right\|^2, \tag{15}$$

subject to the interpolatory conditions $\mathbf{f}(t_i) = \mathbf{d}_i$. The weighted Nu-spline obtained from (15) can be used as a shape-preserving interpolant. Derivative constraints enforce the piecewise cubic interpolant to preserve local monotonicity. In addition, tension parameters can be used for shape modification. Furthermore, a cardinal basis for univariate weighted Nu-splines is formulated [41]. Unlike cubic Hermite basis functions, cardinal basis of weighted Nu-splines are not local. They can not be evaluated easily and efficiently. These cardinal bases can be further used to construct a weighted Nu-spline surface which is a piecewise bicubic surface. The weighted Nu-spline surfaces can be interactively changed by manipulating the control points (interpolating points) and interval/point tension parameters. Note that, the surface boundaries are also weighted Nu-splines. In particular, when all weights are equal, and all tensions are zero, the weighted Nu-spline is reduced to a $C^2$ cubic spline.

Given $t_0 < t_1 < \ldots < t_n$, the weighted spline interpolant [38] is the $C^1$ piecewise cubic function $f(u)$ that minimizes

$$\int_{t_0}^{t_n} w(u)(f^{(2)}(u))^2 du \tag{16}$$

subject to $f(t_i) = d_i$ and relevant end conditions. Over each interval, weighted cubic splines have piecewise constant tension control $w(u)$. In addition, they generalize cubic B-splines. The B-spline-like basis functions can be formulated explicitly. This can facilitate interactive design. Later, based on (16), Foley derived weighted bicubic splines [39] which minimize the following functional:

$$\int_{t_0}^{t_n} \int_{s_0}^{s_m} w(u,v)(\mathbf{f}_{uu,vv}(u,v))^2 du dv \tag{17}$$

subject to the interpolation constraints $f(u_i, v_j) = d_{i,j}$. This scheme has been used for the interpolation

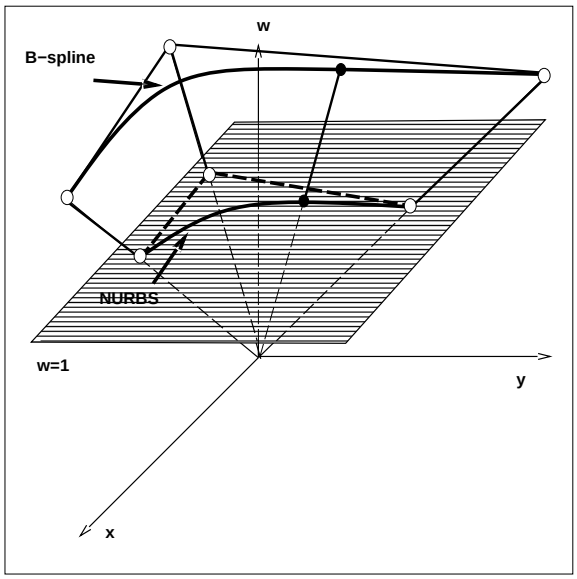



Figure 10: A planar NURBS curve obtained through projection.

of rapidly varying data. The surface is actually a piecewise bicubic Hermite interpolant whose unknown derivatives can be obtained by solving a linear system of equations.

Cohen formulated local basis functions for the locally tensioned splines (LT-spline) which are essentially piecewise $C^0$ cubics having continuous curvatures [21]. It is shown that the LT-splines have the variation diminishing property, the convex hull property, and a straightforward knot insertion algorithm. Both curves and individual basis functions can be easily evaluated. When used for shape modeling, however, LT-splines are often transformed into equivalent B-splines in order to benefit from various algorithms for efficient evaluation, refinement, and hierarchical modeling. Note that, the fact that the B-spline is employed as the underlying formulation for this and other special splines in various applications demonstrates that B-splines are more powerful and flexible schemes.

## 2.6 NURBS Geometry

We now review the formulation of NURBS curves/surfaces and describe their analytic and geometric properties. Intuitively, a spatial NURBS is defined as the projection of a 4D B-spline into a 3D homogeneous space. Fig. 10 illustrates a two-dimensional NURBS obtained through the projection of a three-dimensional B-spline curve.

### 2.6.1 Curves

A NURBS curve generalizes the B-spline. It is the combination of a set of piecewise rational functions with $n+1$ control points $\mathbf{p}_i$ and associated weights $w_i$:

$$\mathbf{c}(u) = \frac{\sum_{i=0}^{n} \mathbf{p}_i w_i B_{i,k}(u)}{\sum_{i=0}^{n} w_i B_{i,k}(u)}, \tag{18}$$

where $u$ is the parametric variable and $B_{i,k}(u)$ are B-spline basis functions. Assuming basis functions of degree $k-1$, a NURBS curve has $n+k+1$ knots $t_i$ in non-decreasing order, $t_0 \leq t_1 \leq \ldots \leq t_{n+k}$. The parametric domain is $t_{k-1} \leq u \leq t_{n+1}$. In many applications, the end knots are repeated with multiplicity $k$ in order to interpolate the initial and final control points $\mathbf{p}_0$ and $\mathbf{p}_n$.

### 2.6.2 Surfaces

A NURBS surface is the generalization of the tensor-product B-spline surface. It is defined over the parametric variables $u$ and $v$ as

$$\mathbf{s}(u,v) = \frac{\sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{p}_{i,j} w_{i,j} B_{i,k}(u) B_{j,l}(v)}{\sum_{i=0}^{m} \sum_{j=0}^{n} w_{i,j} B_{i,k}(u) B_{j,l}(v)}. \tag{19}$$

A NURBS surface has $(m+1)(n+1)$ control points $\mathbf{p}_{i,j}$ and weights $w_{i,j}$. Assuming basis functions along the two parametric axes of degree $k-1$ and $l-1$, respectively, the number of knots is $(m+k+1)(n+l+1)$. The non-decreasing knot sequence is $t_0 \leq t_1 \leq \ldots \leq t_{m+k}$ along the $u$-axis and $s_0 \leq s_1 \leq \ldots \leq s_{n+l}$ along the $v$-axis. The parametric domain is $t_{k-1} \leq u \leq t_{m+1}$ and $s_{l-1} \leq v \leq s_{n+1}$. If the end knots have multiplicity $k$ and $l$ in the $u$ and $v$ axis respectively, the surface patch will interpolate the four corners of the boundary control points.

To evaluate NURBS, the de Boor algorithm can be applied to the numerator and denominator, respectively. A more stable and robust evaluation algorithm which does not make explicit use of 4D projective geometry is described in [33]. The non-uniform knot vector of NURBS offers much better parameterization and greater flexibility in shape design than the previously presented formulations, especially for the fitting of unequally spaced points.

### 2.6.3 Properties

NURBS generalize the non-rational parametric form. In analogy to non-rational B-splines, the rational basis functions of NURBS sum to unity. NURBS inherit many properties of non-rational B-splines such as the strong convex hull property, the variation diminishing property, local support, and invariance under standard geometric transformations (see [34] for more details). Furthermore, they have many additional

properties:

- NURBS offer a unified mathematical framework for common analytic shapes and parametric polynomial forms. For instance, NURBS can be used to precisely express conic segments as well as full conics. NURBS can also represent extruded surfaces, natural quadrics (plane, cylinder, cone and sphere), ruled surfaces, and surfaces of revolution.

- NURBS are infinitely smooth in the interior of a knot span, provided the denominator is not zero, and at a knot they are at least $C^{k-1-r}$ continuous with knot multiplicity $r$. This enables designers to employ NURBS satisfying different smoothness requirements.

- NURBS include weights as extra degrees of freedom which can influence local shape. Weights have a clear geometric interpretation. If a particular weight is zero, the corresponding rational basis function is also zero. Thus, the corresponding control point does not affect the NURBS shape. The spline is attracted toward a control point if the corresponding weight is increased, and repelled from the control point if the weight is decreased (see Fig. 11). By manipulating control points and weights, NURBS provide the greatest flexibility for designing a large variety of shapes.

- Associated with NURBS are a set of powerful geometric toolkits such as knot insertion, refinement, knot removal, and degree elevation. Their evaluation algorithm is fast and computationally stable.

- NURBS are invariant under scaling, rotation, translation, and shear as well as parallel and perspective projections. A linear or rational linear transformation of parameters will not change the shape and order of NURBS. They are the genuine generalization of non-rational B-splines as well as rational and non-rational Bezier forms.

There are various ways to represent a circle using a NURBS curve [84] (Fig. 12 shows one NURBS representation). NURBS can be used to precisely construct conics and rational quadric patches of various types [77]. Spherical patches such as a hemisphere, an octant of the sphere, and a whole sphere can be represented using NURBS [78].

Based on the concept of infinity from projective geometry, Piegl incorporated infinite control points into the NURBS formulation [80, 79]. This can forge a link between the B-spline and Hermite representations. Infinite control points simplify the data set in representing complicated shapes. They can be used to define circles, surfaces of revolution, and general torus patches. Infinite control points can also be used as a design tool which facilitates designers having no expertise of projective geometry.

Piegl systematically discussed NURBS modification through knot insertion and control point/weight based manipulation [81, 82]. Other NURBS geometric design techniques include the interpolation/approximation

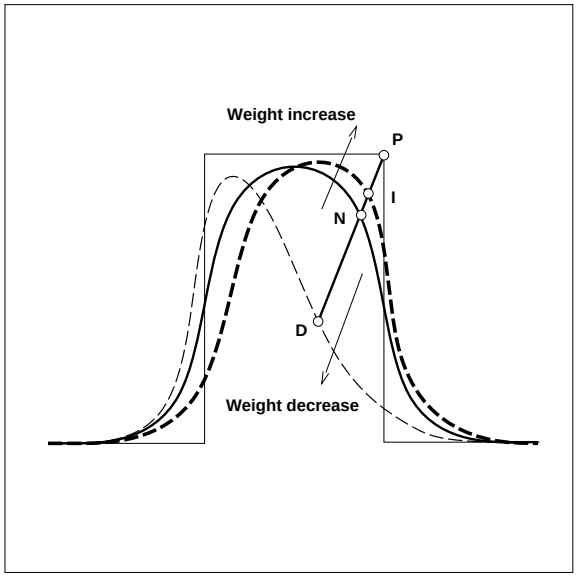



Figure 11: Weight influences on the shape of NURBS curve.

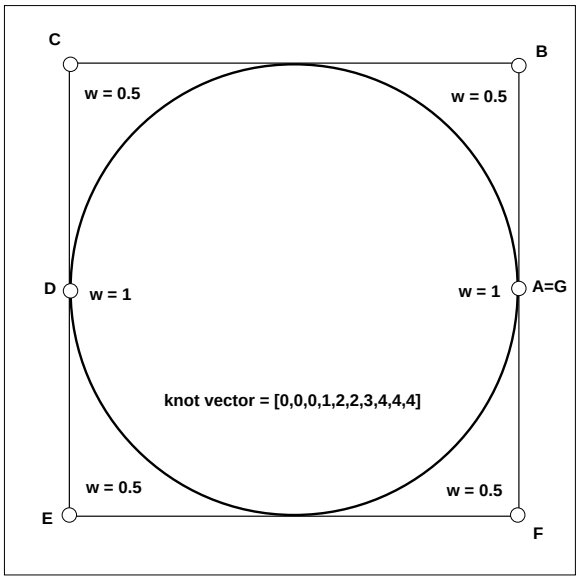



Figure 12: A circle is represented using quadratic NURBS curve with 7 control points.

of a set of data points and cross-sectional design. If weights are provided *a priori*, interpolation and approximation with NURBS is reduced to the solution of a set of linear equations. If weights are not given, however, one often resorts to guessing weights and other heuristics. To fit a large amount of data, especially when the data set is subject to measurement error, least-square approximation is preferable to strict interpolation.

The most frequently used NURBS design techniques are the specification of a control polygon, or interpolation or approximation of data points to generate the initial shape. For surfaces or solids, cross-sectional design including skinning, sweeping, and swinging operations is also popular. The initial shape is then refined into the final desired shape through interactive adjustment of control points and weights and possibly the addition or deletion of knots.

To date, NURBS are the most general parametric representation because they are capable of representing both analytic shapes and free-form parametric forms. They combine a low-degree rational representation of maximal smoothness with a geometrically intuitive variation. The rapid proliferation of NURBS is due partly to their excellent properties and partly to their incorporation into such national and international standards as IGES and PHIGS+. NURBS have been incorporated into a large number of commercial modeling systems.

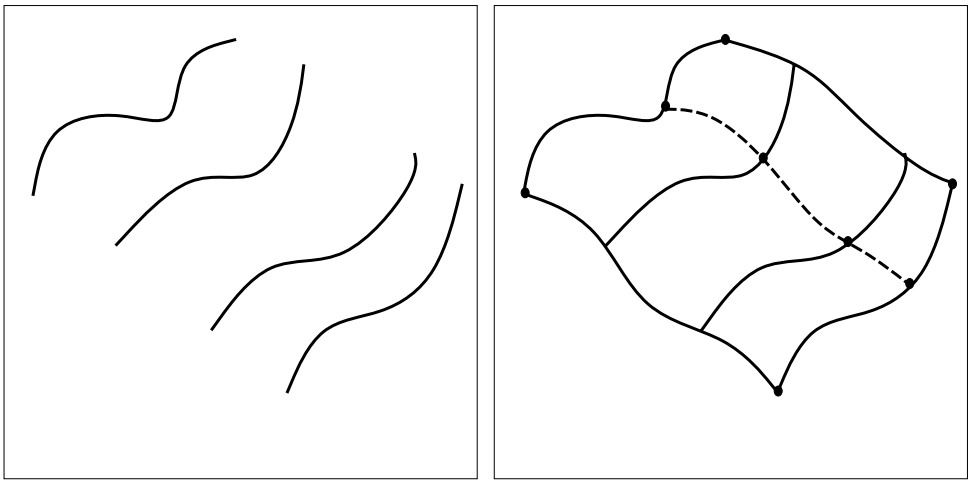
# 3 Geometric Design Paradigms

This section summarizes geometric design techniques including interpolation, approximation, interactive manipulation, and cross-sectional design.

## 3.1 Interpolation

Scientific and engineering applications such as medical imaging, automobile and aircraft design, and geological terrain modeling make use of interpolation techniques. Data interpolation with polynomial splines can be formulated and solved through a set of linear equations. To obtain a unique solution, the number of unknowns must equal the number of independent constraints. For instance, a B-spline curve with $n+1$ control points (see (5)) can be used to interpolate independent $n+1$ data points $\mathbf{d}_i$. Assuming the corresponding parametric values $u_0, \ldots, u_n$ are provided, unknown control points can be obtained by solving

$$\begin{bmatrix} B_{0,k}(u_0) & \cdots & B_{n,k}(u_0) \\ \vdots & \ddots & \vdots \\ B_{0,k}(u_n) & \cdots & B_{n,k}(u_n) \end{bmatrix} \begin{bmatrix} \mathbf{p}_0 \\ \vdots \\ \mathbf{p}_n \end{bmatrix} = \begin{bmatrix} \mathbf{d}_0 \\ \vdots \\ \mathbf{d}_n \end{bmatrix} \tag{20}$$



(a) (a)

Figure 13: The skinning operation (a) isoparametric curves (b) the skinning surface.

Let $\mathbf{C}$ be the coefficient matrix in (20), $\mathbf{p} = [\mathbf{p}_0, \ldots, \mathbf{p}_n]^\top$, $\mathbf{d} = [\mathbf{d}_0, \ldots, \mathbf{d}_n]^\top$, then (20) can be written as

$$\mathbf{Cp} = \mathbf{d}$$

Note that, although the B-spline supports local control, B-spline interpolation is a global scheme. The modification of one data point affects the whole curve. Furthermore, if the parameterization is unknown, (20) is no longer a linear system. The chord-length schemes have been commonly used for the curve parameterization. If only positional data are available, derivative information must be estimated based on the given data to produce a smooth curve. Many automatic methods have been derived for this purpose (see [14] for the details). To achieve data interpolation with tensor-product surfaces, data points must usually conform to a rectangular grid structure.

Another data interpolation approach is the lofting technique which allows a smooth surface to pass through a set of cross-sectional curves. Boolean sum surfaces such as Gordon, Coons, and Ball surfaces are all constructed through the interpolation of a lattice of curves. Among various cross-sectional design techniques, skinning creates a surface interpolating a set of isoparametric curves (see Fig. 13). To interpolate non-isoparametric curves, reparameterization is necessary. Ferguson and Grandine proposed an approach which supports the interpolation of a set of non-constant parameter curves [37].

Much work has been done on spline interpolation. Lounsbery, Mann and DeRose surveyed various interpolation methods over a triangulated polyhedron [59]. Forrest discussed iterative interpolation and approximation of Bezier polynomials [43]. Dyn, Levin, and Gregory used a subdivision surface with tension control to interpolate data points [31]. Geometric splines have become increasingly popular. With geometric continuity, extra degrees of freedom (usually expressed as shape parameters) provide designers additional shape handles. Shape parameters have a clear geometric interpretation. Automatic selection

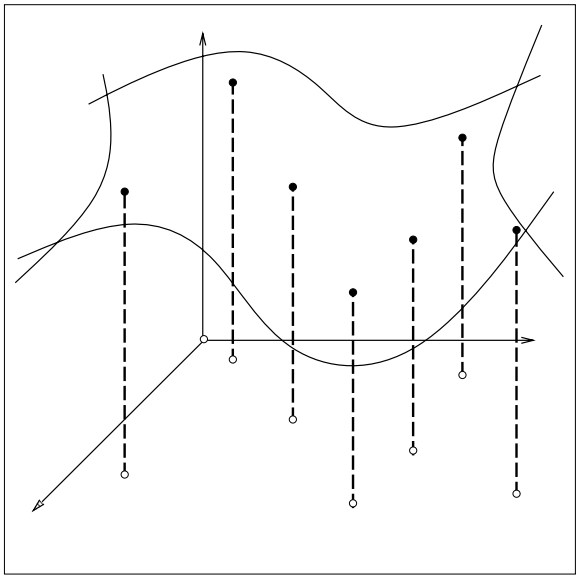


Figure 14: Scattered data interpolation.

of these parameters is highly desirable from designers' point of view. A procedural method [99] exploits piecewise cubic Bezier curves to construct a $G^1$ interpolant with shape parameters. The procedure automatically determines tangent direction and two derivative magnitudes using an intuitive geometric ruled-based approach. Also, the procedural methods can provide pleasing curves for a set of irregular interpolation points.

Scattered data interpolation (see Fig. 14 for an illustration) has also been widely applied especially in scientific computing. Shepard originally formulated the scattered data interpolation as the problem of finding $f(u,v)$ with $f(u_i,v_i) = f_i$, where $(u_i,v_i)$ are irregularly distributed, $i = 0,\ldots,n$. Typical techniques are based on Shepard's formulas and Hardy's multiquadrics [3] which are all distance-weighted and application-dependent interpolants. Shepard's method constructs an interpolant using

$$f(u,v) = \frac{\sum_{i=0}^{i=n} f_i(u,v)/d_i(u,v)^2}{\sum_{i=0}^{i=n} 1/d_i(u,v)^2} \tag{21}$$

where $d_i(u,v)$ is the distance between $(u,v)$ and $(u_i,v_i)$, $f_i(u,v) = f_i + (u-u_i)a_i + (v-v_i)b_i$, $a_i$ and $b_i$ are estimated to approximate the tangent plane at $(u_i,v_i)$. In spite of its simplicity, Shepard's method is a global scheme, it does not reproduce any local shape properties implied by the data because it has local extrema at data sites.

Recently, scattered data interpolation has been extended into the higher dimensional data (e.g. 4D data, surface-on-surface data). Nielson *et al.* investigated the scattered data interpolation over volumetric domains and arbitrarily shaped surface (surface-on-surface) domains [70, 69]. He used Hardy's multiquadric

global interpolant which is defined as

$$f(x,y) = \sum_{i=0}^{n} \alpha_i \sqrt{(x-x_i)^2 + (y-y_i)^2 + R^2} \tag{22}$$

subject to the interpolation constraints: $f(x_i, y_i) = f_i, i = 0, \ldots, n$. The coefficients $\alpha_i$ are computed through a $(n+1) \times (n+1)$ linear system of equations based on interpolatory constraints. Note that the extremely large number of unorganized data can make this system over-determined. It will lead Hardy's method to a least-square approximation. Nielson and Ramaraj also introduced an minimum norm network approach using a set of cubic polynomials to interpolate scattered data points over a spherical domain [71]. The interpolant is determined by functional minimization. The algorithm consists of (i) domain triangulation, (ii) construction of the boundary curve network which satisfies the optimization functional subject to the interpolation constraints, and (iii) patch generation from the curve network to cover the entire domain using triangular interpolants. Step (ii) and (iii) are executed via minimization.

### 3.2 Approximation

Shape approximation is necessary for several reasons.

**Data Exchange:** Different modeling systems often enforce certain limitations such as maximum allowable degree. To exchange geometric data among several design systems, users must approximate higher order geometric entities which can not be precisely represented in the desired system because of the limitations.

**Data Reduction:** It can be formally defined as: given a set of control points with basis functions of degree $n$ (e.g. Bezier curve), find another set of control points with basis functions of degree $m < n$, such that the new representation is the best approximation of the original one. Knot removal is a commonly used technique for data reduction. It removes knots from a spline without perturbing the spline more than a given tolerance. Scattered data (especially noisy data) fitting is another typical example for data reduction.

**Decomposition:** Geometric modeling and data visualization often require a large amount of data. As more and more data are processed by computers, transmission and storage becomes a bottleneck for the geometric process. In order to implement a more economical representation and efficient analysis, multi-resolution techniques are often used.

**Hierarchical Modeling:** The same geometric object is represented at different levels. It starts with a rough global shape defined on a relatively coarse scale. The local details are only modeled on the

refining scale. Hierarchical representation is useful for high speed rendering because it avoids the sampling problem.

**Special Shapes:** When a curve (or surface) has no simple or compact mathematical representation (*e.g.* the intersecting curve of surfaces, or the offset of curves or surfaces), shape approximation is necessary.

If $m$ data points are provided in (20), where $m > n+1$, the coefficient matrix in (20) becomes an $m \times (n+1)$ matrix. Thus, this linear system becomes overdetermined. The interpolation is transformed into approximation. It can be solved as follows

$$\mathbf{C}^\top \mathbf{C}\mathbf{p} = \mathbf{C}^\top \mathbf{d}, \tag{23}$$

where $\mathbf{C}^\top\mathbf{C}$ is the new squared coefficient matrix, $\mathbf{p}$ is the unknown control point vector, and $\mathbf{d}$ is the given data point vector. It can be easily verified that the solution of (23) also minimizes the error functional

$$E = \sum_{i=0}^{m} \|\mathbf{c}(u_i) - \mathbf{d}_i\|^2$$

In general, the whole approximation procedure involves parameterization, least-squares fitting and parameter optimization. Chord-length parameterization is often used for curve approximation. Hoschek applied non-linear parameter optimization to the degree reduction of Bezier splines [55]. Sarkar and Menq presented an algorithm to implement smooth least-square approximation using cubic B-splines, where the parameterization is formulated as a nonlinear minimization problem [95, 94]. In most applications, data points are not regularly distributed. Hoppe *et al.* presented an algorithm which can reconstruct a polygonal approximation from unorganized data points by estimating the tangent plane for each data point [54]. The geometry and topology can be inferred automatically from the data. However, the input data must be dense because it is impossible to recover features where there is insufficient sampling. Very often, a set of data points can not be sampled exactly, Cheng and Barsky presented an algorithm that uses a cubic spline curve to interpolate specified data points at some knots and pass through specified regions at other knots while minimizes the energy [18]. Chou and Piegl used piecewise cubic rational Bezier splines to fit a set of data points and their tangent directions in the least-square sense [19].

Although rational curves and surfaces started to be widely used in CAGD only during the past ten years, rational functions have been studied for many decades in approximation theory. In general, rational approximation provides better accuracy. Nevertheless, it is difficult to reduce rational approximation into a linear system. Recently, Pratt, Goult and Ye proposed an efficient linearized approximation of rational functions [87]. If $\mathbf{g}(u) = \mathbf{p}(u)/q(u)$ is a rational polynomial, where $\mathbf{p}(u) = \sum_{i=0}^{m} \mathbf{p}_i A_i(u)$, $q(u) =$

$\sum_{j=0}^{n} q_i B_i(u)$, the minimization of the error functional

$$E(\mathbf{g}) = \int_0^1 \left\| \mathbf{f}(u) - \frac{\mathbf{p}(u)}{q(u)} \right\|^2 du$$

reduces to a set of nonlinear equations. Although it can be solved by iterative methods in practice, the result largely depends on a good initialization. To avoid the nonlinearity, alternatively, one can minimize the linear functional

$$\hat{E}(\mathbf{g}) = \int_0^1 \|\mathbf{f}(u)q(u) - \mathbf{p}(u)\|^2 du$$

This linear minimization process can be further divided into two steps. First, solve for

$$\frac{\partial \hat{E}}{\partial \mathbf{p}_i} = 0,$$

where $\mathbf{p}_i$ is evaluated in terms of $q(u)$. Second, solve for

$$\frac{\partial \hat{E}}{\partial q_j} = 0,$$

where the unknown $q_j$ of $q(u)$ are determined uniquely. Although it has been shown that this linearized functional produces good results for many applications, it is not the least-squares (best) approximation of the rational function.

### 3.3 Interactive Modification

Due to measurement errors and the subjective judgement of designers, there are no objective *best* shapes. Thus, it is more reasonable to apply interactive techniques. Conventionally, interactive design is accomplished through control polygon manipulation. If geometric continuity is used, extra shape parameters such as tensions can also be exploited to implement interactive control. The interactive procedure must be easy to use. In addition, the underlying mathematical formulation should be transparent to users who may have little mathematical background.

There exists a large variety of spline formulations. Each formulation encourages the implementation of certain styles of interaction. Even though the mathematical power of two formulations may be equivalent, the ease of use from the designer's viewpoint may vary dramatically. For instance, the cubic B-spline without multiple knots is inherently $C^2$ continuous. The cubic Bezier curve, however, possesses $C^2$ continuity only when its control points are subject to certain geometric constraints. Extra design requirements to ensure the piecewise Bezier curve be $C^2$ must be enforced. This may be quite expensive because the constraints have to be maintained throughout the whole design process. Bartels et al. presented a general

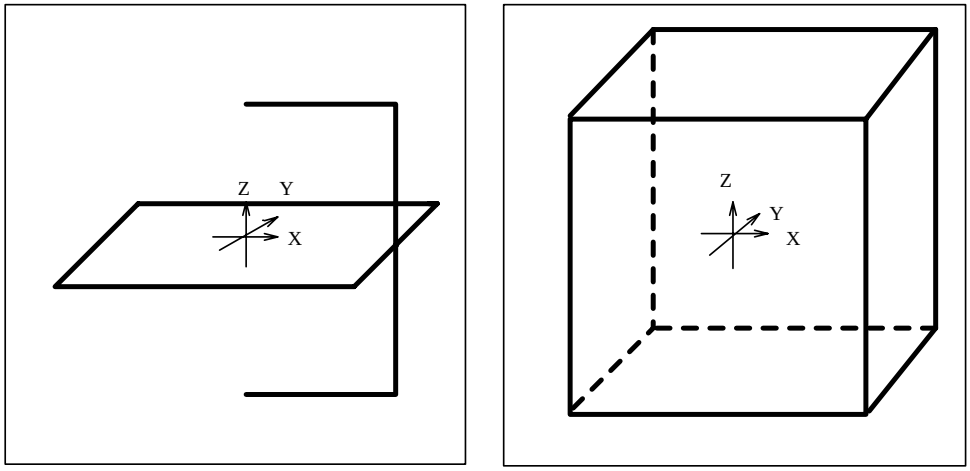
statistical analysis approach to measure and analyze user performance with interactive curve manipulation of a single control point [10]. B-splines, Bezier splines, Catmull-Rom splines, and two other $C^2$ interpolating splines are investigated. It has been demonstrated that there is a significant performance difference among these five formulations. The B-spline formulation is the best in terms of both match quality and time cost of achieving the objective for the curve case.

Cross-sectional design allows control polygons of isoparametric curves to be used as tools for interactive surface refinements. High-level operators such as bends, twists and free-form deformations (FFD) are also applicable. But traditional free-form interaction has long been control point based. Control point manipulation is an indirect, often unnecessarily tedious solution for interactive modeling. Although the B-splines have been very popular because of their interaction convenience, the B-spline formulation does not support direct shape interaction. Direct and precise free-form shape manipulation is often desired. Fowler presented a new technique to kinematically manipulate position and/or normal at arbitrary selected points on the tensor-product B-spline surface [45]. His approach is equivalent to solving an underdetermined linear system of equations. To derive a unique solution, extra geometric constraints are employed to minimize the combined movement of the control vertices.

Deformation is a highly intuitive and interactive operation, it can be used to generate large families of geometric shapes. Barr proposed twisting, bending, and tapering transformation of geometric objects which can be used to create complex objects from simpler ones [6]. Globally and locally defined deformations may be used as hierarchical operations, expanding the CAD/CAM repertoire. Forsey and Bartels presented Hierarchical B-splines which support local refinement and manipulation on B-spline surfaces [44]. In general, refinement is implemented through knot insertion. Knot insertion on the surface will introduce more control points outside the local refining region. Local hierarchical refinement on the B-spline surface can be achieved through the use of an overlay surface and offset referencing of control vertices. Galyean and Hughes presented a new interactive modeling technique for solid sculpting [46]. Controlled by a 3D input device, a sculpting tool can modify the voxel array values which represent the volumetric material. A set of sculpting tools such as cutting and adding is provided. The algorithm allows the modification of the material through purely geometric means.

### 3.4 Cross-Section Design

Many objects of interest, especially manufactured objects, exhibit symmetries. Often it is convenient to model symmetric objects through cross-sectional design by specifying profile curves [36]. Cross-sectional design operations such as skinning, sweeping, and swinging have been widely used for interactive shape modification. The fundamental requirements are: (i) the surface must be visually smooth if all section




(a) (b)

Figure 15: Construction of a cubical NURBS swung surface. (a) NURBS profile curve on x-z plane, NURBS trajectory curve on x-y plane. (b) Cube surface wireframe.

curves are visually smooth, (ii) no unnecessary undulation, and (iii) affine invariance.

Originating in lofting theory, a skinning process generates a surface passing through a series of cross-sectional profiles. Surface design can thus be reduced to the design of (a) a series of compatible cross-sectional profiles, and (b) a path trajectory. Extra effort has to be done to make all section curves compatible. One typical technique to achieve this goal is degree elevation. Because of the lack of 3D interaction techniques and tools, sectional curves are often created as planar profiles. But trial-and-error procedures used for designing fair curves can be laborious. To avoid spatial surface interaction, projection curves are used for interrogation and necessary modification. Other compromises must be made towards the conflicting parameterization demands. Sweeping is a special case of skinning where a set of constant section curves are provided. The construction of the surface also depends on the parameterization of the path (spine) curve.

Barr employed a spherical cross-product to construct superquadrics from profiles [6]. Woodward [113] introduced the swinging operator by extending the spherical cross-product with a scaling factor, and applied it to generate surfaces with B-spline profile curves (see also [20]). Piegl carried the swinging idea over to NURBS curves [83]. He proposed NURBS swung surfaces, a special type of NURBS surfaces formed by swinging one planar NURBS profile curve in the $x$-$z$ plane along a second NURBS trajectory curve in the $x$-$y$ plane. For example, Fig. 15 illustrates the design of a cubical NURBS swung surface from two NURBS profile curves. Swinging generalizes rotational sweeping. Designers can reposition control points, change weights, modify the knot vector, and move the data points of sectional curves in order to modify the surface shape. Several geometric shape design systems, including the recent one in [100], include some form of swinging (or sweeping) among their repertoire of techniques.

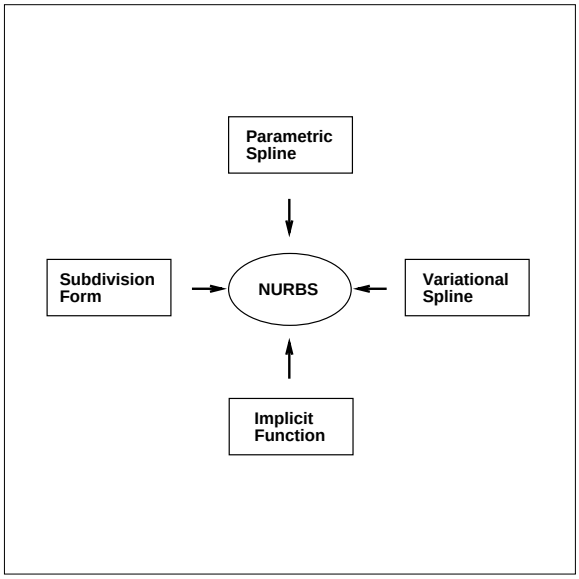



Figure 16: NURBS as an integrated form for various geometric entities.

# 4 Challenges for Geometric Design

NURBS have become a *de facto* standard in commercial modeling systems. Fig. 16 illustrates that NURBS can be used to integrate various geometric forms. Despite the extraordinary flexibility of NURBS, traditional design methodology can not exploit their full potential. This is because NURBS have been viewed as purely geometric primitives, which require designers to interactively adjust many DOFs—control points and associated weights—to obtain desired shapes.

Conventional geometric design techniques previously described are kinematically driven. Despite modern interactive devices, this process can be clumsy and laborious when it comes to designing complex, real-world objects.

Rational splines such as NURBS provide better accuracy than ordinary polynomial functions when used for approximation. Although NURBS can be used to represent many analytical shapes such as conics precisely, no simple and flexible methods are offered to designers for the automatic selection of weights in an intuitive and meaningful way. Conventional methods through user specification of weights are both heuristic and *ad hoc*.

Given a set of empirical 3D data points provided by a scanner, the task of inferring both geometry and topology from this scattered data is extremely difficult. This is because no implicit neighborhood information is available and the data are often subject to measurement error.

Flexible and real-time interaction between designers and the modeling systems is crucial not only for future CAD systems but also for other diverse applications such as architecture design, surgical planning, and robotics. Directly manipulating geometry in an interactive environment requires the integration of virtual reality techniques with well established CAD design and manufacture methodology. In contrast,

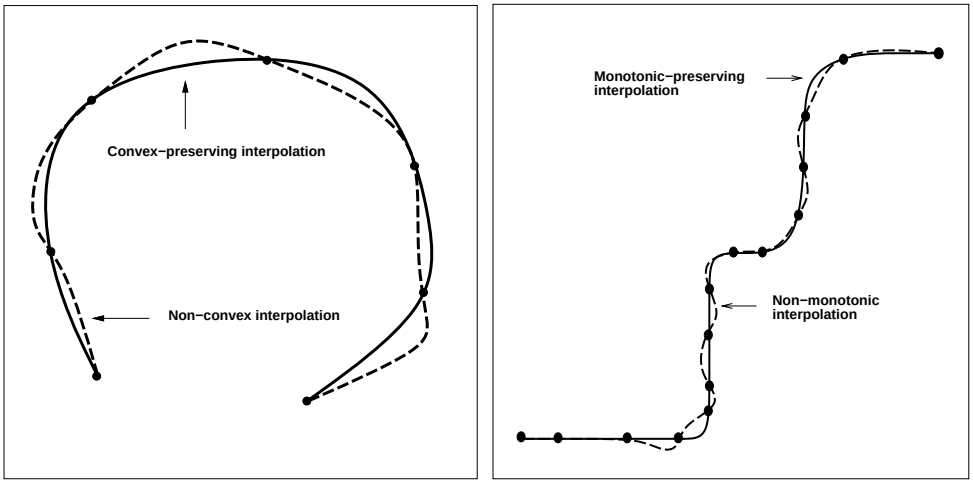



Figure 17: Shape-preserving interpolants (a) convex-preserving (b) monotonic-preserving.

dynamic 3D interaction of free-form geometric shapes in the CAD system has been rarely explored. As a result, the full potential of dynamic interaction is yet to be realized.

The above difficulties are due to the intrinsic features of traditional geometric modeling. Pure geometric representation is abstract. It does not have behavior. The design and manufacture process, however, requires a mechanism which can accomplish the interactive modification of geometric information both efficiently and precisely. It can be demonstrated that traditional design methodology can not offer us such a dynamic and interactive framework for time-varying requirements. Strongly motivated to bridge the large gap between the characteristics of geometry and the design and manufacture process requirements, we investigate how to associate physics and physical behavior with abstract and static geometry.

# 5 Variational Design

Designers not only require the shape to satisfy functional requirements such as interpolation and continuity but also need visually-pleasing and fair shapes implied in the given data points. For instance, if the data are locally monotonic or convex, the interpolation function should also be a shape-preserving interpolant. Shape-preservation requires shape fidelity reflected by the data beyond interpolation/approximation. Fig. 17 illustrates two typical examples of shape-preserving requirements.

In contrast, ordinary interpolating techniques usually generate shape oscillations not implied in data points. To achieve satisfactory design, both functional requirements and aesthetic criteria are needed. One of the most important problems in CAGD is to construct visually-pleasing (fair) splines that can either interpolate or approximate a given set of points. Note that, aesthetic criteria are very subjective because they are often style-dependent. It can be difficult to express such requirements formally. Nevertheless, quantitative judgement has to be formulated even before the design and manufacture process starts. The

variational technique is more appropriate because it can implement qualitative and subjective criteria as quantitative ones.

Variational optimization is also used to determine the interpolant unknowns where only partial information is provided. Once interpolation constraints are satisfied, there are generally surplus degrees of freedom. The values of these free parameters dramatically influences the shape. Local methods apply heuristics to manually set free variables. This will not always achieve satisfactory results because it needs a lot of user intervention. Likewise, manual operation is not efficient for a large number of degrees of freedom. In contrast, global optimization methods allow the primitives to autonomously deform to minimize an energy functional subject to constraints. This approach requires less user input than conventional free-form modeling techniques. Moreover, the fairness criterion acting as an objective function discourages undesired oscillation.

Parkinson interpolated a set of data points with a set of biarcs [72]. Unknown gradients at data points are determined by the efficient linearized approximation of an energy minimization. Wever presented a global cubic $C^2$ interpolant using energy minimization [112]. A non-negative exponential spline is derived as a shape-preserving interpolant for positive data [111], where tension parameters are used in the nonlinear constrained optimization of bending energy. Moreton and Sequin presented a simple mechanism which allows the creation of complex smoothly shaped surfaces of any topological types with piecewise biquintic Bezier patches [65]. Nonlinear optimization techniques are used to minimize a fairness functional based on the variation of curvature in order to determine remaining free parameters unspecified through geometric interpolatory constraints. The use of the variation of curvature functional allows commonly used shapes such as spheres, cylinders and tori to be reconstructed subject to a set of compatible constraints. However, the computational cost of the modeling software is extremely expensive. It prevents this technique from being used for interactive surface design.

In [93], the fairness criterion is defined as the curvature plot which is a piecewise monotonic function with the fewest possible monotone pieces. To generate a fair curve, an interactive fairing process is often used in which users interactively modify control points repeatedly in terms of curvature plots until a curve of acceptable fairness is obtained. Manual adjustment is inefficient and completely incompatible with any automatically refining processes. For instance, when using B-splines to approximate data points, it is possible to exhibit unwanted behavior due to digitizing errors. Therefore, it is inevitable to have an unacceptable curvature plot. Sapidis and Farin developed an algorithm for locally fairing planar B-spline curves by repeatedly removing and reinserting knots of the spline [93]. He further derived a simple geometric condition for the quadratic Bezier curve to ensure monotonic curvatures [92]. This condition also provides a simple criterion which can be used for automatically correcting the curvature plot.

Optimal design requires designers (engineers and stylists) to focus on the shape instead of the abstract geometric representation. For this matter, functional constraints can be expressed in the form of partial differential equations with suitable boundary conditions [60, 13]. Although a wide range of surfaces can be generated based on various PDE requirements and boundary constraints, the PDE-based approach is not compatible with conventional spline techniques. It provides little intuition for shape adjustments. Also, it is difficult to determine how the variation of boundary curves affects the surface interior. Thus, the expertise of designers is necessary to achieve desirable shape refinement. To demonstrate that the PDE surface is a compatible approach with established techniques of mainstream surface design, B-splines have been used to approximate PDE surfaces [12].

# 6 Physics-Based Modeling

The physics-based design paradigm can provide a means to overcome the above drawbacks. Free-form deformable models, initially introduced to computer graphics in [105] and further developed in [104, 85, 101, 75, 102, 61] are particularly relevant. They are also useful for user interfaces, virtual reality, image processing, and geometric modeling. Physics-based models are governed by the mechanical laws of continuous bodies which can be expressed in the form of dynamic differential equations. The dynamic and realistic behaviors can be obtained by solving an associated motion equation numerically. To date, however, researchers in the fields of computer vision and graphics devote most of their endeavors to the investigation of constraints, articulated rigid or nonrigid body synthesis, and complex scene control. Less effort has been applied to free-form dynamic interaction between designers and individual manufactured objects which is especially useful for geometric design.

Physical simulation can be used as an effective interactive tool for building and manipulating a wide range of models. It supports the dynamic manipulation of complex physical models. Terzopoulos and Fleischer demonstrated simple interactive sculpting using viscoelastic and plastic models [104]. Celniker and Gossard developed an interesting prototype system [16] for interactive free-form design based on the finite-element optimization of energy functionals proposed in [104]. The system combines geometric constraints with sculpting operations based on forces and loads to yield fair shapes. However, this approach does not provide interactive mechanisms in dealing with forces and loads. Bloor and Wilson developed related models using similar energies and numerical optimization [13], and in [12] they proposed the use of B-splines for this purpose. Subsequently, Celniker and Welch investigated deformable B-splines with linear constraints [17].

Welch and Witkin extended the approach to trimmed hierarchical B-splines (see also [44]) for interactive modeling of free-form surface with constrained variational optimization [110]. The traditional control point

approach is intuitive by allowing a conceptually simple change. However, to enforce a precise modification, many–even all– the control points have to be repositioned in order to achieve the desired effect.

Motivated by the fact that splines provide insufficient detail for modeling certain natural shapes, such as terrains, and fractals provide insufficient shape control, Szeliski and Terzopoulos proposed constrained fractals, a hybrid of deterministic splines and stochastic fractals which combines their complementary features [101]. Through the use of the energy minimization principles the constrained fractal can be applied to synthesize realistic terrain surface from sparse elevation data. Multiresolution stochastic relaxation is used to compute fractals efficiently.

Thingvold and Cohen proposed a deformable model based on a B-spline surface, whose control points are mass points connected by elastic springs and hinges [107]. The control polygon refinement conserves physical quantities such as mass, spring, and hinge. Pentland et al. used a modal analysis method to decompose non-rigid dynamics into a set of independent vibration modes based on eigenvalues [75, 74, 73]. Discarding high-frequency modes, the number of unknowns can be largely reduced while preserving the accuracy and generality of the formulation. A special global polynomial deformation can be associated with a set of vibration modes and applied to the animation with a superquadric ellipsoid.

Extracting geometric information from scattered data is important for visualization and object recognition. Miller et al. presented a method generating a simple topologically closed geometric model from a volumetric data set [63]. The polyhedron model can expand itself like a balloon until it reaches the volumetric boundary of the scanned object through a relaxation process which also minimizes the prescribed cost function. Global subdivision is needed wherever appropriate for complex shape during the optimization process.

Szeliski and Tonnesen presented a new model of elastic surfaces based on interactive particle systems [102]. New particles are added into the system automatically which enables the surface to stretch and grow. Particle based surfaces can split and join without manual intervention. In spite of the interactive power for free-form modeling, particle based surfaces have some disadvantages, such as the lack of a precise and compact mathematical representation which presumably is vital in engineering applications.

In summary, physics-based models have dynamic behavior which is governed by physical laws. This allows designers to directly manipulate and interactively sculpt shapes using a variety of force-based tools. The energy-based optimization process can be implemented automatically. In addition, physics-based shape design can free designers from making nonintuitive decisions such as assigning weights to NURBS or determining shape parameters in variational splines. Furthermore, with physics-based direct manipulation, non-expert users are able to concentrate on visual shape variation without needing to comprehend the underlying mathematical formulation.

# 7 D-NURBS for Physics-Based Design

By marrying advanced geometric modeling with computational physics, we have developed D-NURBS, a physics-based generalization of geometric NURBS for geometric design. The mathematical development of D-NURBS consists of four related parts: (i) D-NURBS curves [106], (ii) tensor product D-NURBS surfaces [106], (iii) swung D-NURBS surfaces [89], and (iv) triangular D-NURBS surfaces [88, 90]. In the following context, we summarize the significant features of this physics-based design framework. The detailed mathematics and implementation of D-NURBS have been documented in [106, 89, 88, 90]. They are beyond the scope of this survey.

## 7.1 Motivation

Generalizing B-splines to either NURBS or special geometric and variational splines has one feature in common. These generalizations have offered designers extraordinary flexibility when utilized for geometric design. Nevertheless, traditional design methodology can not exploit the full potential of the underlying geometric formulations.

Traditional geometric design is kinematically driven. Designers are faced with the tedium of indirect shape manipulation through a bewildering variety of geometric parameters; i.e., by repositioning control points, adjusting weights, and modifying knot vectors. To sculpt a complex shape, these interactive techniques may take too many trial-and-error procedures. Also, it often requires designers to have specific expertise. Despite the recent prevalence of sophisticated 3D interaction devices, the indirect geometric design process remains clumsy and time consuming in general.

Shape design to required specifications by manual adjustment of available geometric DOFs is often elusive, because relevant design tolerances are typically shape-oriented and not control point/weight oriented. Due to the geometric flexibility of various representations such as NURBS, traditional geometric shape refinement remains *ad hoc* and ambiguous. For instance, to adjust a shape should the designer move a control point, or change a weight, or move two control points,...? Control point and weight dependent manipulation is not natural because these DOFs do not reside on the sculpted geometric entity.

The design requirements of engineers and stylists can be very different. Whereas engineers focus on technical and functional issues, stylists emphasize aesthetically-driven conceptual design. Therefore, typical design requirements may be stated in both quantitative and qualitative terms, such as "a fair and pleasing surface which approximates scattered data and interpolates a cross-section curve." Such requirements may impose both local and global constraints on shape. The incremental manipulation of local shape parameters to satisfy complex local and global shape constraints is at best cumbersome and often unproductive.

Stylists are often interested in geometric "theme variation." This requires geometric entities to be gen-

erated quickly and naturally. Unlike engineers, stylists are concerned more with the geometric shape than with its underlying mathematical description. It is apparent that conventional interpolation/approximation techniques, which often generate computerized models from digitized data, may not be quite suitable to the time-varying requirements of stylists.

## 7.2 The Physics-Based Approach

Physics-based modeling methodology and finite element techniques provide a means to overcome some disadvantages of conventional geometric design.

- The behavior of the deformable model is governed by physical laws. Through a computational physics simulation, the model responds dynamically to applied simulated forces in a natural and predictable way. Shapes can be sculpted interactively using a variety of force-based "tools". The physics-based sculpting is intuitive for shape design and control.

- The equilibrium state of the dynamic model is characterized by a minimum of the potential energy of the model subject to imposed constraints [103]. It is possible to formulate potential energy functionals that satisfy local and global design criteria, such as curve or surface (piecewise) smoothness, and to impose geometric constraints relevant to shape design.

- The physical model may be built upon a standard geometric foundation, such as free-form parametric curve and surface representations. This means that while shape design may proceed interactively or automatically at the physical level, existing geometric toolkits are concurrently applicable at the geometric level.

Physics-based shape design can free designers from the need to make nonintuitive decisions such as assigning weights to NURBS or determining shape parameters in variational splines through the direct and intuitive sculpting of NURBS objects. In addition, with physics-based direct manipulation, non-expert users are able to concentrate on visual shape variation without needing to comprehend the underlying mathematical formulation.

Moreover, geometric design, especially conceptual design, is a time-varying process because designers are often interested in not only the final static equilibrium shape but the intermediate shape variation as well. In contrast to recent variational design approaches, time is fundamental to physics-based modeling. Additional advantages can be obtained through the use of real-time dynamics:

- An "instantaneous" optimizer (if such a thing existed) can produce some kinematics if it were applied at every interaction step to satisfy constraints. But the motion would be artificial and there would

be nothing to prevent sudden, nonsmooth motions (depending on the structure of the constraints) which can be annoying and confusing. By contrast, the dynamic formulation is much more general in that it marries the geometry with time, mass, force, and constraint. Dynamic models produce smooth, natural motions which are familiar and easily controlled.

- Dynamics facilitates interaction, especially direct manipulation and interactive sculpting of complex geometric models for real-time shape variation. The dynamic approach subsumes all of the geometric capabilities in an elegant formulation which grounds shape variation in real-world physics. Despite the fact that incremental optimization may provide a means of interaction, pure optimization techniques can easily become trapped in the local minima characteristic of non-linear models and/or constraints. In contrast, real-time dynamics can overcome the difficulty of incremental optimization through the incorporation of inertial properties into the model and the interactive use of force-based tools by the designer.

- Practical design processes span conceptual geometric design and the fabrication of mechanical parts. Physics-based modeling techniques and real-time dynamics integrates geometry with physics in a natural and coherent way. The unified formulation is potentially applicable throughout the entire design and manufacturing process.

D-NURBS are physics-based models that incorporate mass distributions, internal deformation energies, forces, and other physical quantities into the NURBS geometric substrate (Fig. 18). The equations of motion for D-NURBS are formulated systematically through the application of Lagrangian mechanics. Finite element analysis is employed to reduce these equations to efficient algorithms that can be simulated at interactive rates using standard numerical techniques. The dynamic behavior of D-NURBS results from the numerical integration of their nonlinear differential equations in response to the applied forces and constraints to produce physically meaningful, hence intuitively predictable motion and shape variation. The dynamics framework of D-NURBS provides a principled mechanism for automatically determining the values of control points and weights in accordance with design requirements.

D-NURBS generalizes traditional geometric design methodology using geometric NURBS. Within this physics-based shape design framework, elastic functionals can be used to allow the qualitative imposition of "fairness" criteria through quantitative means. Optimal shape design (nonlinear shape optimization) results when D-NURBS achieve static equilibrium subject to global or local geometric constraints. Moreover, geometric design, especially conceptual design, is a time-varying process because designers are often interested in not only the final static equilibrium shape but the intermediate shape variation as well. Consequently, designers can interactively sculpt complex shapes to satisfy required specifications not only in the

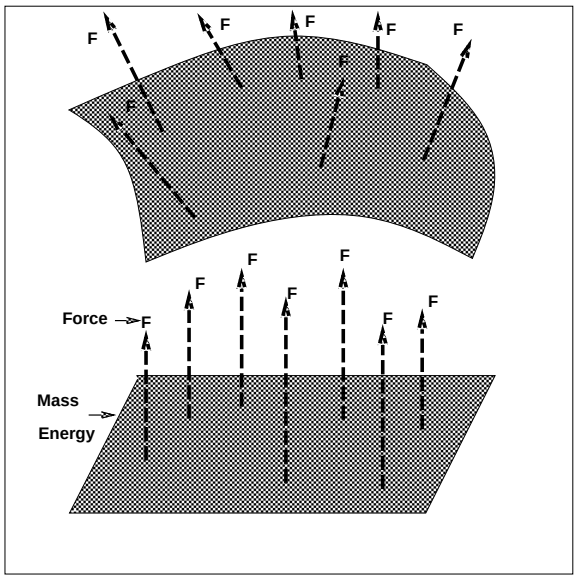



Figure 18: D-NURBS as new Physics-based models.

traditional indirect fashion, by adjusting control points, but also through direct physical manipulation, by applying simulated forces and local and global shape constraints. The key advantage of this physics-based design framework is that existing geometric toolkits continue to be applicable at the basic geometry level, while it also affords designers new force-based toolkits that support dynamic manipulation and interactive sculpting at the physics level (see Fig. 19). More importantly, the two type of toolkits are compatible with each other. Designers are free to choose either one or both to achieve design requirements. The significant advantage of D-NURBS is that it integrates force-based dynamic manipulation and interactive sculpting with all existing NURBS geometric features and toolkits.

### 7.3 Modeling Applications

We have implemented a D-NURBS prototype modeling environment. We use this system to demonstrate that D-NURBS are effective tools in a wide range of applications, including interactive sculpting through force-based direct manipulation tools, shape blending, and scattered data fitting.

D-NURBS provide a natural solution to the solid rounding problem. In contrast to the geometric approach, the D-NURBS can produce a smooth fillet with the proper continuity requirements by minimizing its internal deformation energy. Additional position and normal constraints may be imposed across the boundary of the surface. The dynamic simulation automatically produces the desired final shape. Fig. 20 demonstrates edge rounding using D-NURBS surfaces. In Fig. 20(a1), we round an edge at the intersection of two planar faces. The faces are formed using quadratic D-NURBS patches with $8 \times 5$ control points. The D-NURBS rounds the corner as it achieves the minimal energy equilibrium state shown in Fig. 20(a2). Fig. 20(b1) illustrates the rounding of a trihedral corner of a cube. The corner is represented using a

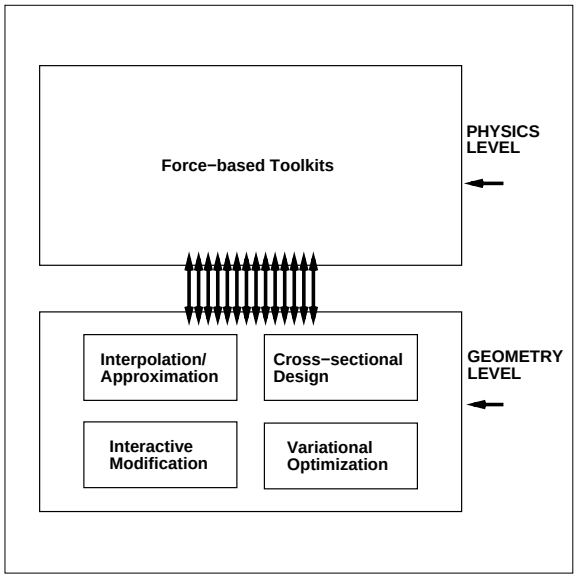



Figure 19: Two-level Physics-based design paradigm.

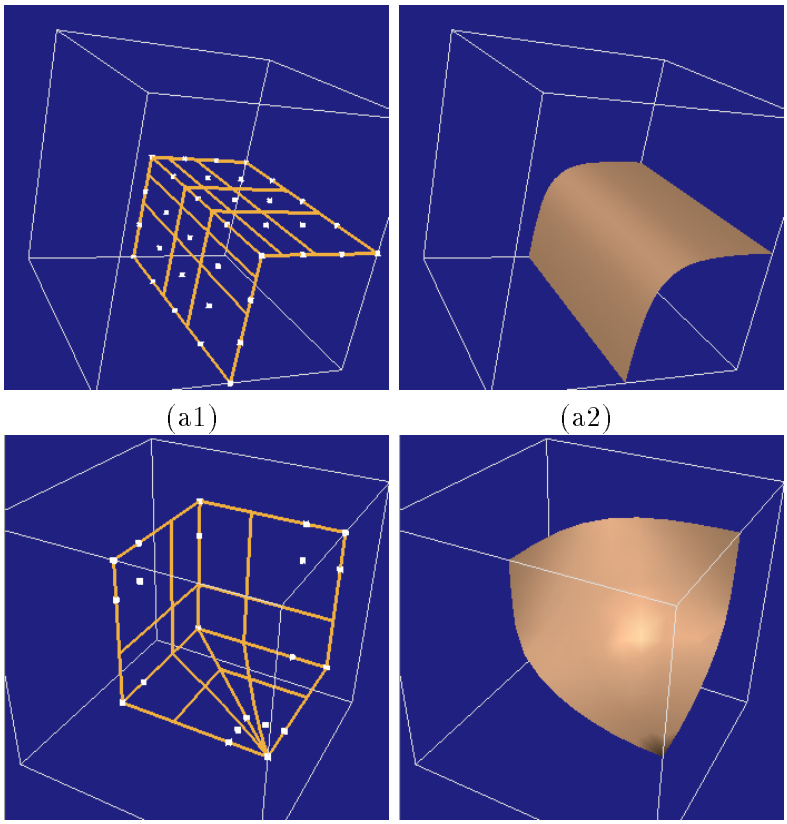



Figure 20: Solid rounding: (a) rounding an edge between polyhedral faces; (b) rounding a trihedral vertex. (a1) Initial configuration of control points and patches. (a2) Rounded D-NURBS surface in static equilibrium. (b1) Initial configuration of control points and patches. (b2) Rounded D-NURBS surface. In both examples, the control points along edges have multiplicity 2.

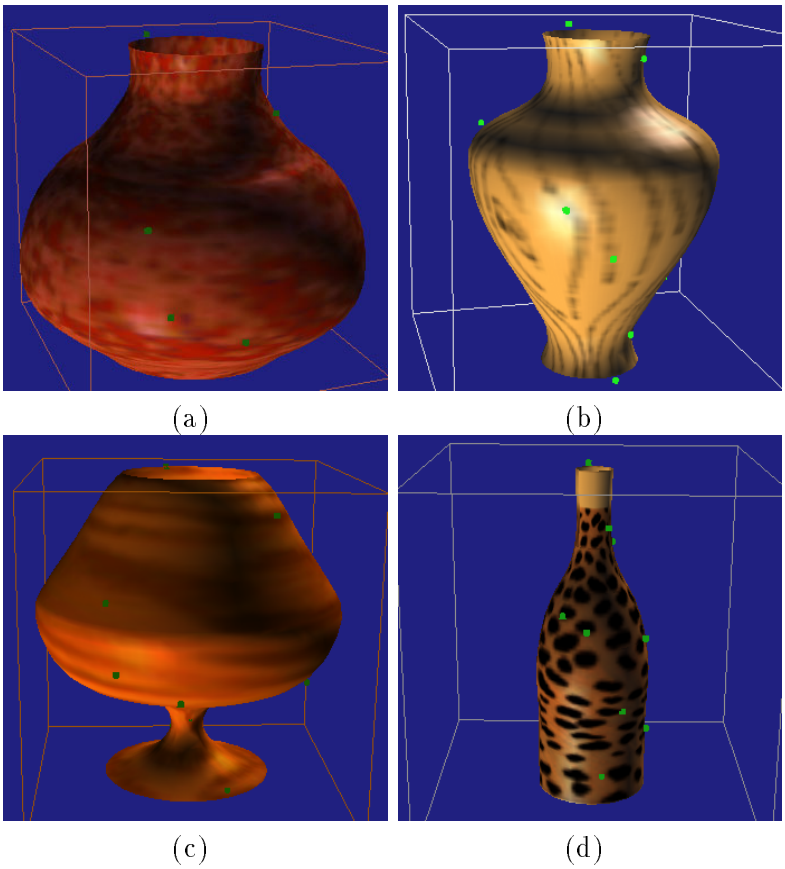



Figure 21: Four fitted shapes. (a) Pot. (b) Vase. (c) Glass. (d) Bottle.

quadratic D-NURBS surface with $6 \times 5$ control points. The corner is rounded with position and normal constraints along the far boundaries of the faces (Fig. 20(b2)).

D-NURBS is also useful for scattered data fitting. Interesting situations arise when there are fewer or more data points than there are degrees of freedom in the model, leading to underconstrained or overconstrained fitting problems. The data interpolation problem is amenable to common constraint techniques [64]. Approximation can be achieved by physically coupling the D-NURBS surface to the data through Hookean spring forces. Spatial data points are often associated with corresponding nearest material points of the model. We use a D-NURBS swung surface with 70 control points to recover a pot, a vase, a bottle, and a wine glass generated from synthetic data. The number of randomly sampled data from objects are 10, 13, 14, and 17, respectively. Fig. 21(a-d) shows the final reconstructed shapes.

The physics-based modeling approach is ideal for interactive sculpting of surfaces. It provides direct manipulation of the dynamic surface to refine its shape through the application of interactive sculpting tools in the form of forces. Fig. 22 illustrates four shapes sculpted using spring forces. The initial open surface is generated using a quadratic triangular B-splines with 24 control points.

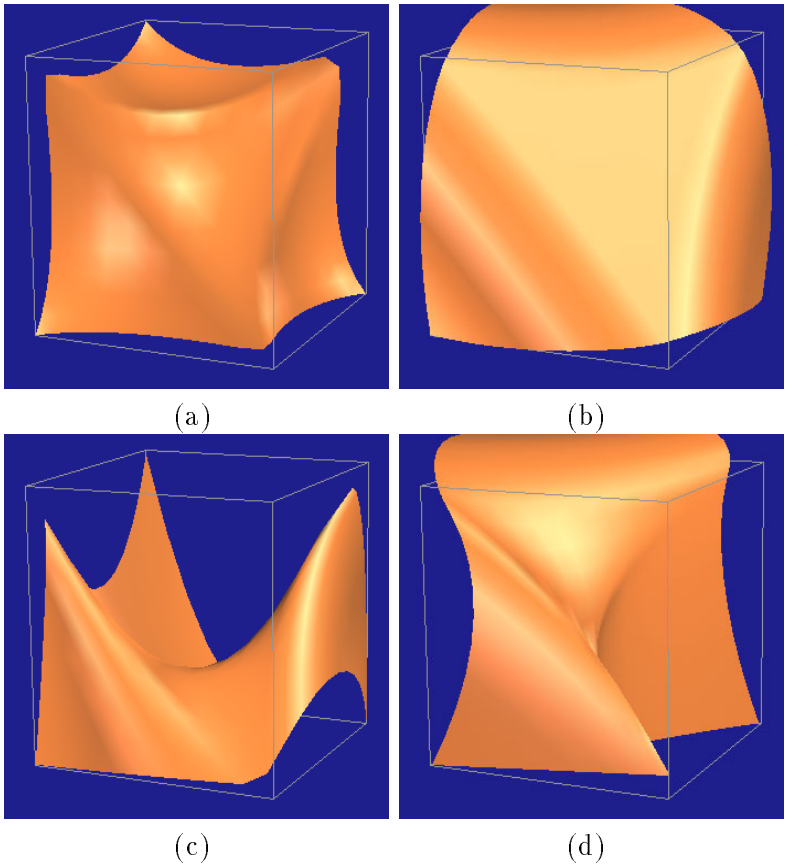



Figure 22: Interactive sculpting of an open quadratic surface into four different shapes (a–d).

# 8 Conclusion

We have reviewed geometric modeling and demonstrated that NURBS, among many shape representations, become an industrial standard in geometric design systems because of their ability to unify both free-form parametric splines and commonly used algebraic functions. We also survey various shape design techniques and discuss some of the limitations of geometric design. Furthermore, we summarize the major advantages of D-NURBS — a newly developed physics-based shape design framework. It has been shown that D-NURBS can be used to overcome the drawbacks of established geometric design methodology. Moreover, the D-NURBS physics-based framework furnishes designers not only the standard geometric toolkits but powerful force-based sculpting tools as well. It provides mechanisms for automatically adjusting unknown parameters to support user manipulation and satisfy design requirements.

The application experiments obtained from the D-NURBS interactive modeling environment illustrate that D-NURBS can provide a promising approach for a variety of CAD and graphics modeling problems such as constraint-based optimization for surface design and fairing, automatic settings of weights in surface fitting, and interactive sculpting through applied forces. In addition, D-NURBS promise to serve as a basis for future research endeavors in physics-based CAGD, graphics, virtual reality, vision, and scientific

visualization.

Since our models are built on the industry-standard NURBS geometric substrate, designers working with them can continue to make use of the existing array of geometric design toolkits. With the advent of high-performance graphics systems, the physics-based framework is poised for incorporation into commercial design systems to interactively model and sculpt complex shapes in real-time.

## References


[1] S. Auerbach, R. Gmelig Meyling, M. Neamtu, and H. Schaeben. Approximation and geometric modeling with simplex B-splines associated with irregular triangles. *Computer Aided Geometric Design*, 8(1):67–87, 1991.

[2] C. Bajaj and I. Ihm. Algebraic surface design with Hermite interpolation. *ACM Transactions on Graphics*, 11(1):61–91, 1992.

[3] R. Barnhill. A survey of the representation and design of surfaces. *IEEE Computer Graphics and Applications*, 3(7):9–16, 1983.

[4] R. Barnhill. Surfaces in computer aided geometric design: A survey with new results. *Computer Aided Geometric Design*, 2(1):1–17, 1985.

[5] R. Barnhill, G. Birhoff, and W. Gordon. Smooth interpolation in triangles. *J. Approximation Theory*, 8:114–128, 1973.

[6] A. Barr. Superquadrics and angle preserving transformations. *IEEE Computer Graphics and Applications*, 1(1):11–23, 1981.

[7] B. Barsky. Local control of bias and tension in Beta-splines. *ACM Transactions on Graphics*, 2(2):109–134, 1983.

[8] B. Barsky and T. DeRose. Geometric continuity of parametric curves: three equvilent characterization. *IEEE Computer Graphics and Applications*, 9(6):60–68, 1989.

[9] B. Barsky and T. DeRose. Geometric continuity of parametric curves: constructions of geometrically continuious plines. *IEEE Computer Graphics and Applications*, 10(1):60–68, 1990.

[10] R. Bartels, J. Beatty, K. Booth, E. Bosch, and P. Jolicoeur. Experimental comparison of splines using the shape-matching paradigm. *ACM Transactions on Graphics*, 12(3):179–208, 1993.

[11] J. Bloomenthal and K. Shoemake. Convolution surfaces. *Computer Graphics*, 25(4):251–256, 1991.

[12] M.I.G. Bloor and M.J. Wilson. Representing PDE surfaces in terms of B-splines. *Computer-Aided Design*, 22(6):324–331, 1990.

[13] M.I.G. Bloor and M.J. Wilson. Using partial differential equations to generate free-form surfaces. *Computer-Aided Design*, 22(4):202–212, 1990.

[14] W. Bohm, G. Farin, and J. Kahmann. A survey of curve and surface methods in CAGD. *Computer Aided Geometric Design*, 1(1):1–60, 1984.

[15] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10(6):350–355, 1978.

[16] G. Celniker and D. Gossard. Deformable curve and surface finite elements for free-form shape design. *Computer Graphics*, 25(4):257–266, 1991. (Proc. ACM Siggraph'91).

[17] G. Celniker and W. Welch. Linear constraints for deformable B-spline surfaces. In *Proceedings, Symposium on Interactive 3D Graphics*, pages 165–170, 1992.

[18] F. Cheng and B. Barsky. Interproximation: interpolation and approximation using cubic spline. *Computer-Aided Design*, 23(10):700–706, 1991.

[19] J. Chou and L. Piegl. Data reduction using cubic rational B-splines. *IEEE Computer Graphics and Applications*, 12(3):60–68, 1992.

[20] E. Cobb. *Design of Sculptured Surfaces Using the B-spline Representation.* PhD thesis, University of Utah, 1984.

[21] E. Cohen. A new local basis for designing with tensioned splines. *ACM Transactions on Graphics*, 6(2):81–122, 1987.

[22] M. Cox. The numerical evaluation of B-splines. *J. Institute of Mathematics and its Applications*, 10:134–149, 1972.

[23] W. Dahmen and C. Micchelli. On the linear independence of multivariate B-splines, I. triangulations of simploids. *SIAM J. Numer. Anal.*, 19(5):993–1012, 1982.

[24] W. Dahmen and C. Micchelli. Recent progress in multivariate splines. In C.K. Chui, L.L. Schumaker, and J.D. Ward, editors, *Approximation Theory IV*, pages 27–121. Academic Press, New York, 1983.

[25] W. Dahmen, C. Micchelli, and H.-P. Seidel. Blossoming begets B-spline bases built better by B-patches. *Mathematics of Computation*, 59(199):97–115, 1992.

[26] C. de Boor. On calculating with B-Splines. *Journal of Approximation Theory*, 6(1):50–62, 1972.

[27] C. de Boor. Splines as linear combinations of B-splines. In G. Lorentz, C. Chui, and L.L. Schumaker, editors, *Approximation Theory II*, pages 1–47. Academic Press, New York, 1976.

[28] T. DeRose and B. Barsky. Geometric continuity, shape parameters and geometric constructions for Catmull-Rom splines. *ACM Transactions on Graphics*, 7(1):1–41, 1988.

[29] D. Doo and M. Sabin. Behaviour of recursive division surfaces near extraordinary points. *Computer-Aided Design*, 10(6):356–360, 1978.

[30] D. Dutta, R. Martin, and M. Pratt. Cyclides in surface and solid modeling. *IEEE Computer Graphics and Applications*, 13(1):53–59, 1993.

[31] N. Dyn, D. Levin, and J. Gregory. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Transactions on Graphics*, 9(2):160–169, 1990.

[32] G. Farin. Triangular Bernstein-Bezier patches. *Computer Aided Geometric Design*, 3(2):83–127, 1986.

[33] G. Farin. Trends in curve and surface design. *Computer-Aided Design*, 21(5):293–296, 1989.

[34] G. Farin. *Curves and Surfaces for Computer aided Geometric Design: A Practical Guide.* Academic Press, second edition, 1990.

[35] R. Farouki and J. Hinds. A hierarchy of geometric forms. *IEEE Computer Graphics and Applications*, 5(5):51–78, 1985.

[36] I.D. Faux and M.J. Pratt. *Computational Geometry for Design and Manufacture*. Ellis Horwood, Chichester,UK, 1979.

[37] D. Ferguson and T. Grandine. On the construction of surfaces interpolating curves: I. A method for handling nonconstant parameter curves. *ACM Transactions on Graphics*, 9(2):212–225, 1990.

[38] T. Foley. Local control of interval tension using weighted splines. *Computer Aided Geometric Design*, 3(4):281–294, 1986.

[39] T. Foley. Weighted bicubic spline interpolation to rapidly varying data. *ACM Transactions on Graphics*, 6(1):1–18, 1987.

[40] T. Foley. A shape perserving interpolant with tension controls. *Computer Aided Geometric Design*, 5(2):105–118, 1988.

[41] T. Foley and H. Ely. Surface interpolation with tension controls using cardinal bases. *Computer Aided Geometric Design*, 6(2):97–109, 1989.

[42] P. Fong and H.-P. Seidel. An implementation of triangular B-spline surfaces over arbitrary triangulations. *Computer Aided Geometric Design*, 3-4(10):267–275, 1993.

[43] A. Forrest. Interactive interpolation and approximation by Bezier polynomials. *Computer-Aided Design*, 22(9):527–537, 1990.

[44] D.R. Forsey and R.H. Bartels. Hierarchical B-spline refinement. *Computer Graphics*, 22(4):205–212, 1988.

[45] B. Fowler. Geometric manipulation of tensor product surfaces. In *Proceedings, Symposium on Interactive 3D Graphics*, pages 101–108, 1992.

[46] T. Galyean and J. Hughes. Sculptinng: An interactive volumetric modeling technique. *Computer Graphics*, 25(4):267–274, 1991.

[47] T.N.T. Goodman and K. Unsworth. Manipulating shape and producing geometric continuity in Beta-spline curves. *IEEE Computer Graphics and Applications*, 6(2):50–56, 1986.

[48] W. Gordon and R. Riesenfeld. B-spline curves and surfaces. In R. Barnhill and R. Riesenfeld, editors, *Computer Aided Geometric Design*, pages 95–126. Academic Press, 1974.

[49] T. Grandine. The stable evaluation of multivariate simplex splines. *Mathematics of Computation*, 50(181):197–205, 1988.

[50] G. Greiner and H.-P. Seidel. Modeling with triangular B-splines. *IEEE Computer Graphics and Applications*, 14(2):56–60, 1994.

[51] H. Hagen. Geometric spline curves. *Computer Aided Geometric Design*, 2(1-3):223–227, 1985.

[52] C. Hoffmann. Implicit curves and surfaces in CAGD. *IEEE Computer Graphics and Applications*, 13(1):79–88, 1993.

[53] K. Hollig. Multivariate splines. *SIAM J. Numer. Anal.*, 19(5):1013–1031, 1982.

[54] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *Computer Graphics*, 26(2):71–77, 1992.

[55] J. Hoschek. Approximate conversion of spline curves. *Computer Aided Geometric Design*, 4(1-2):59–66, 1987.

[56] D. Lasser. Two remarks on Tau-splines. *ACM Transactions on Graphics*, 9(2):198–211, 1990.

[57] C. Loop and T. DeRose. A multisided generalization of Bezier surfaces. *ACM Transactions on Graphics*, 8(3):204–234, 1989.

[58] C. Loop and T. DeRose. Generalized B-spline surfaces of arbitrary topology. *Computer Graphics*, 24(4):347–356, 1990.

[59] M. Lounsbery, S. Mann, and T. DeRose. Parametric surface interpolation. *IEEE Computer Graphics and Applications*, 12(5):45–52, 1992.

[60] T.W. Lowe, M.I.G. Bloor, and M.J. Wilson. Functionality in blend design. *Computer-Aided Design*, 22(10):655–665, 1990.

[61] D. Metaxas and D. Terzopoulos. Dynamic deformation of solid primitives with constraints. *Computer Graphics*, 26(2):309–312, 1992. (Proc. ACM Siggraph'92).

[62] C.A. Micchelli. On a numerically efficient method for computing with multivariate B-splines. In W. Schempp and K. Zeller, editors, *Multivariate Approximation Theory*, pages 211–248. Birkhauser, Basel, 1979.

[63] J. Miller, D. Breen, W. Lorensen, R. O'bara, and M. Wozny. Geometrically deformed models: A method for extracting closed geometric models from volume data. *Computer Graphics*, 25(4):217–226, 1991.

[64] M. Minoux. *Mathematical Programming*. Wiley, New York, 1986.

[65] H.P. Moreton and C.H. Sequin. Functional optimization for fair surface design. *Computer Graphics*, 26(2):167–176, 1992. (Proc. ACM Siggraph'92).

[66] S. Muraki. Volumetric shape description of range data using Blobby Model. *Computer Graphics*, 25(4):227–235, 1991.

[67] G. Nielson. A locally controllable spline with tension for interactive curve design. *Computer Aided Geometric Design*, 1(3):199–205, 1984.

[68] G. Nielson. Rectangular $\nu$-splines. *IEEE Computer Graphics and Applications*, 6(2):35–40, 1986.

[69] G. Nielson. Scattered data modeling. *IEEE Computer Graphics and Applications*, 13(1):60–70, 1993.

[70] G. Nielson, T. Foley, B. Hamann, and D. Lane. Visualizing and modeling scattered multivariate data. *IEEE Computer Graphics and Applications*, 11(3):47–55, 1991.

[71] G. Nielson and R. Ramaraj. Interpolation over a sphere based upon a minimum norm network. *Computer Aided Geometric Design*, 4(1-2):41–57, 1987.

[72] D. Parkinson. Optimisec biarc curves with tension. *Computer Aided Geometric Design*, 9(3):207–218, 1992.

[73] A. Pentland and B. Horowitz. Recovery of nonrigid motion and structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):730–742, 1991.

[74] A. Pentland and S. Sclaroff. Closed-form solutions for physically based shape modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):715–729, 1991.

[75] A. Pentland and J. Williams. Good vibrations: Modal dynamics for graphics and animation. *Computer Graphics*, 23(3):215–222, 1989.

[76] J. Peters. Smooth free-form surfaces over irregular meshes generalizing quadratic splines. *Computer Aided Geometric Design*, 10(3-4):347–361, 1993.

[77] L. Piegl. Representation of quadric primitives by rational polynomials. *Computer Aided Geometric Design*, 2(1-3):151–155, 1985.

[78] L. Piegl. The sphere as a rational Bezier surface. *Computer Aided Geometric Design*, 3(1):45–52, 1986.

[79] L. Piegl. Infinite control points–A method for representing surfaces of revolution using boundary data. *IEEE Computer Graphics and Applications*, 7(3):45–55, 1987.

[80] L. Piegl. On the use of infinite control points in CAGD. *Computer Aided Geometric Design*, 4(1-2):155–166, 1987.

[81] L. Piegl. Modifying the shape of rational B-splines. part 1:curves. *Computer-Aided Design*, 21(8):509–518, 1989.

[82] L. Piegl. Modifying the shape of rational B-splines. part 2:surfaces. *Computer-Aided Design*, 21(9):538–546, 1989.

[83] L. Piegl. On NURBS: A survey. *IEEE Computer Graphics and Applications*, 11(1):55–71, Jan. 1991.

[84] L. Piegl and W. Tiller. A menagerie of rational B-Spline circles. *IEEE Computer Graphics and Applications*, 9(5):48–56, 1989.

[85] J. Platt and A. Barr. Constraints methods for flexible models. *Computer Graphics*, 22(4):279–288, 1988.

[86] H. Pottmann. Smooth curves under tension. *Computer Aided Design*, 22(4):241–245, 1990.

[87] M. Pratt, R. Goult, and L. Ye. On rational parametric curve approximation. *Computer Aided Geometric Design*, 10(3-4):363–377, 1993.

[88] H. Qin and D. Terzopoulos. Dynamic manipulation of triangular B-splines. In *Proceedings of Third ACM/IEEE Symposium on Solid Modeling and Applications*, pages 351–360, Salt Lake City, May 1995. ACM Press.

[89] H. Qin and D. Terzopoulos. Dynamic NURBS swung surfaces for physics-based shape design. *Computer Aided Design*, 27(2):111–127, 1995.

[90] H. Qin and D. Terzopoulos. Triangular NURBS and their dynamic generalizations. *Computer Aided Geometric Design*, 1996. in press.

[91] R. Riesenfeld. *Applications of B-spline Approximation to Geometric Problems of Computer-aided Design*. PhD thesis, Syracuse University, 1973.

[92] N. Sapidis. Controlling the curvature of a quadratic bezier curve. *Computer Aided Geometric Design*, 9(2):85–91, 1992.

[93] N. Sapidis and G. Farin. Automatic fairing algorithm for B-spline curves. *Computer-Aided Design*, 22(2):121–129, 1990.

[94] B. Sarkar and C. Menq. Parameter optimization in approximating curves and surfaces to measurement data. *Computer Aided Geometric Design*, 8(4):267–290, 1991.

[95] B. Sarkar and C.-H. Menq. Smooth-surface approximation and reverse engineering. *Computer-Aided Design*, 23(9):623–628, 1991.

[96] I.J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions. *Quarterly of Applied Mathematics*, 4:45–99, 1946.

[97] T. Sederberg. Techniques for cubic algebraic surfaces. *IEEE Computer Graphics and Applications*, 10(4):14–25, 1990.

[98] T. Sederberg. Techniques for cubic algebraic surfaces. *IEEE Computer Graphics and Application*, 10(5):12–21, 1990.

[99] L. Shirman and C. Sequin. Procedural interpolation with geometrically continuous cubic splines. *Computer-Aided Design*, 24(5):267–277, 1992.

[100] J. Snyder and J. Kajiya. Generative modeling: A symbolic system for geometric modeling. *Computer Graphics*, 26(2):369–378, 1992.

[101] R. Szeliski and D. Terzopoulos. From splines to fractals. *Computer Graphics*, 23(3):51–60, 1989.

[102] R. Szeliski and D. Tonnesen. Surface modeling with oriented particle systems. *Computer Graphics*, 26(2):185–194, 1992.

[103] D. Terzopoulos. Regularization of inverse visual problems involving discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(4):413–424, 1986.

[104] D. Terzopoulos and K. Fleischer. Deformable models. *The Visual Computer*, 4(6):306–331, 1988.

[105] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. *Computer Graphics*, 21(4):205–214, 1987.

[106] D. Terzopoulos and H. Qin. Dynamic NURBS with geometric constraints for interactive sculpting. *ACM Transactions on Graphics*, 13(2):103–136, 1994.

[107] J.A. Thingvold and E. Cohen. Physical modeling with B-spline surfaces for interactive design and animation. *Computer Graphics*, 24(2):129–137, 1990. Proceedings, 1990 Symposium on Interactive 3D Graphics.

[108] C. Traas. Practice of bivariate simplicial splines. In W. Dahmen et al, editor, *Computation of Curves and Surfaces*, pages 383–422. Kluwer Academic Publishers, 1990.

[109] K.J. Versprille. *Computer-Aided Design Applications of the Rational B-Spline Approximation form.* PhD thesis, Syracuse University, 1975.

[110] W. Welch and A. Witkin. Variational surface modeling. *Computer Graphics*, 26(2):157–166, 1992. (Proc. ACM Siggraph'92).

[111] U. Wever. Non-negative exponential. *Computer-Aided Design*, 20(1):11–16, 1988.

[112] U. Wever. Optimal parameterization for cubic splines. *Computer-Aided Design*, 23(9):641–644, 1991.

[113] C. Woodward. Cross-sectional design of B-spline surfaces. *Computers and Graphics*, 11(2):193–201, 1987.