Physics-Based Modeling Framework for Graphics, Computer-Aided Design, and Visualization

Hong Qin

Department of Computer Science

State University of New York at Stony Brook

Stony Brook, NY 11794–4400

E-mail: qin@cs.sunysb.edu

Tel: (516)632-8450

Fax: (516)632-8334

Abstract

Geometric modeling is of significance to a wide variety of fields such as graphics, computer-aided design (CAD), visualization, medicine, and virtual environment. This paper presents a novel modeling methodology — physics-based geometric modeling paradigm, which can combine geometric objects with physical and material properties, thus allowing intuitive interaction with geometric objects by means of force. Conventional modeling techniques, which are based on pure geometric primitives and operations, often require users to painstakingly adjust numerous geometric parameters to achieve modeling requirements. This indirect design process can be laborious and oftentimes clumsy. By contrast, physics-based geometric modeling treats these parameters as generalized coordinates which evolve automatically in response to simulated forces and geometric constraints according to the principles of Lagrangian mechanics. The integration of standard geometric primitives with intuitively meaningful physical behavior provides a systematic and unified approach to many important applications on graphics, CAD, and visualization. Typical examples include shape blending, constraint-based and parametric design, the fitting of unstructured data, and solid rounding with geometric and physical constraints. Physics-based modeling method also support direct manipulation and interactive sculpting of shapes using force-based tools.

Keywords:

Graphics, CAD, Visualization, Geometric Modeling, Dynamics, Finite Elements, Interactive Techniques.

1 Introduction

Geometric modeling is of fundamental importance to a large variety of research areas including computer graphics, computer-integrated engineering and manufacturing, scientific visualization, robotics, and other disciplines. It centers on methods and algorithms for shape representation, approximation, manipulation, and computation. During the past three decades, numerous shape representations have been developed for geometric modeling applications. Among them, non-uniform rational B-splines, or NURBS, [40] have become an industry-standard representation [24]. NURBS are prevalent in commercial modeling systems primarily because they provide a unified mathematical representation for free-form curves and surfaces and for standard analytic shapes such as conics, quadrics, and surfaces of revolution. More specifically, NURBS generalize the non-rational parametric form. They inherit many of the properties of non-rational B-splines, such as the strong convex hull property, variation diminishing property, local support, and invariance under standard geometric transformations. Moreover, they have some additional properties. NURBS can be used to satisfy different smoothness requirements. They include weights as extra degrees of freedom that influence local shape. Most importantly, NURBS offer a common mathematical framework for implicit and parametric polynomial forms. In principle, they can represent analytic functions such as conics and quadrics precisely, as well as free-form shapes. This paper will focus on NURBS models because they can be used to integrate various geometric forms in geometric modeling (Fig. 1). Note that, there are several different types of NURBS representations, including NURBS curves, tensor-product NURBS surfaces, and triangular NURBS surfaces. Experienced practitioners can design a large variety of NURBS objects by adjusting the positions of control points, setting the values of associated weights, and modifying the distribution of knots [7, 22, 23, 24, 39]. Despite modern interaction techniques, however, this conventional geometric design process with NURBS can oftentimes be clumsy and laborious.

In this paper, we propose a physics-based geometric modeling framework with NURBS that addresses these problems through a new class of models known as *Dynamic* NURBS, or D-NURBS. D-NURBS are physics-based models that incorporate mass distributions, internal deformation energies, and other physical quantities into the NURBS geometric formulation (Fig. 2). D-NURBS models are governed by dynamic differential equations that, when integrated numerically through time, continuously evolve the control points and weights in response to applied forces. Within our physical dynamics framework, standard geometric NURBS objects inherit some of the universally familiar behaviors of physical, real-world objects. Thus, physics-based design augments (rather than supersedes) standard geometry and geometric design, offering attractive new advantages. In particular, the elastic energy functionals associated with D-NURBS allow the imposition of global qualitative "fairness" criteria through quantitative means. Linear or nonlinear shape constraints may be imposed either as hard constraints that must not be violated, or as soft

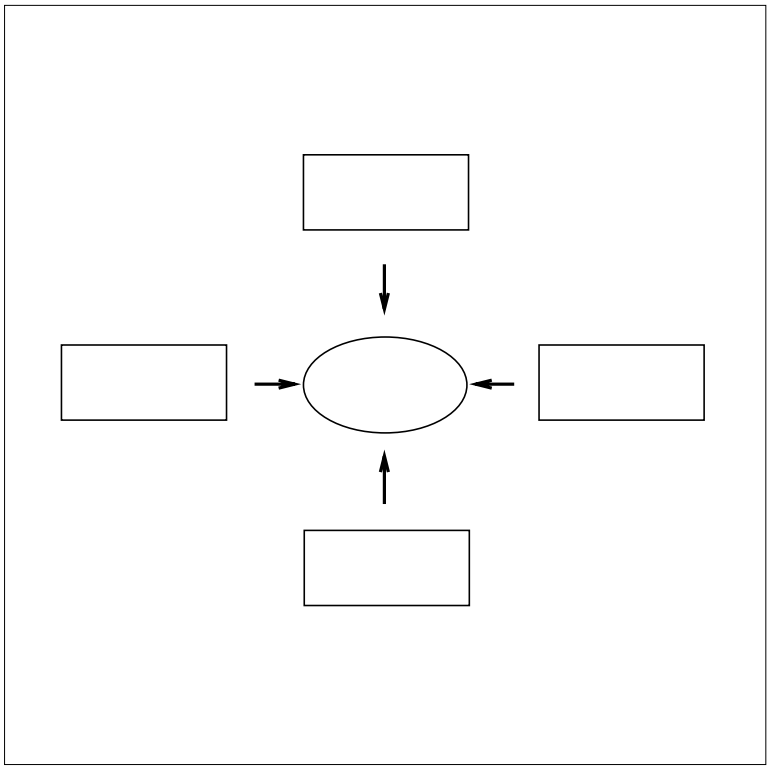

Figure 1: NURBS as an integrated representation for various geometric entities.

constraints to be satisfied approximately. Constraints may be interpreted intuitively as forces and optimal shape design results when D-NURBS are allowed to achieve static equilibrium subject to these forces. More importantly, the D-NURBS formulation supports interactive direct manipulation of NURBS objects, which results in physically meaningful, hence intuitively predictable, deformation and shape variation. Using D-NURBS, modelers can interactively sculpt complex shapes not merely by kinematic adjustment of control points and weights, but dynamically as well—by applying simulated forces. Additional control over dynamic sculpting stems from the modification of physical parameters such as mass, damping, and elastic properties.

2 Motivation

During the past three decades, a variety of geometric techniques have been developed to satisfy modeling, design, and manufacturing requirements. First, modelers can specify geometric entities either by interpolating or approximating a set of regular data points, scattered data points, or boundary curves. Second, users can indirectly manipulate the degrees of freedom (DOFs) of the underlying geometric formulation (e.g. control points and/or weights that define NURBS-based objects). 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 DOFs. In general, design requirements

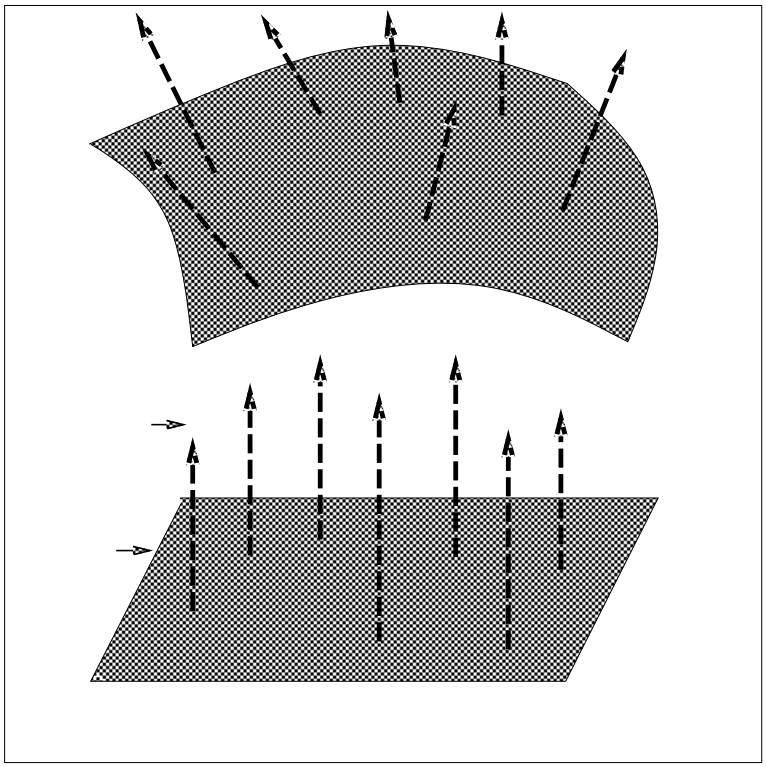

Figure 2: D-NURBS as new Physics-based geometric models.

can be posed as both quantitative or functional criteria and qualitative or aesthetic criteria. The latter criteria are usually satisfied through the use of optimization techniques. 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 generated 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.

Pure geometry is abstract and static. It does not have the intuitive behavior of traditional physical design media such as modeling clay. The modeling process, however, requires an interactive mechanism that can update geometric shapes efficiently and precisely. Designers prefer to have a real-time, interactive environment that offers them various tools for effectively handling complicated shapes and allows them to manipulate geometric models directly in an intuitive and predictable way. Unfortunately, traditional geometric modeling methodology falls short of offering designers such a dynamic and interactive framework. To bridge the gap between geometry and time-varying modeling requirements, we propose and develop a new physics-based modeling and design methodology. This methodology addresses existing problems through the integration of physical, especially dynamical, behavior with abstract, static geometry.

3 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 physics-based models (e.g. D-NURBS) 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 dynamic objects is characterized by a minimum of the potential energy of the model subject to imposed constraints [34]. 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

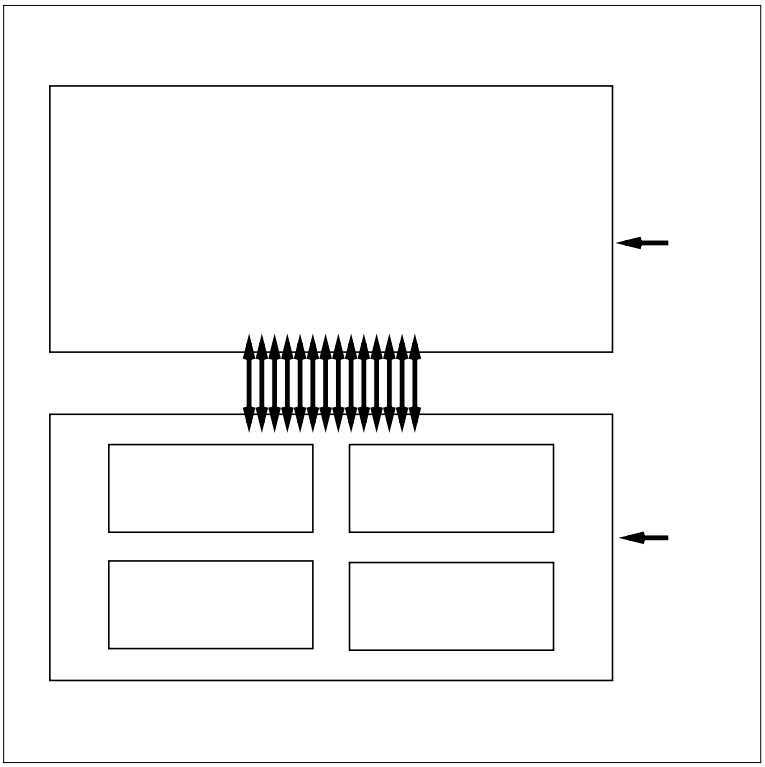

Figure 3: A Two-Level Physics-Based Design Paradigm.

or automatically at the physical level, existing geometric toolkits are concurrently applicable at the geometric level. (Fig. 3). More importantly, the two types of toolkits are compatible with each other. Designers are free to choose either one or both to achieve design requirements.

Physics-based shape design can free designers from having to make nonintuitive decisions, such as assigning weights to NURBS. In addition, with physics-based direct manipulation, non-expert users are able to concentrate on visual shape variation without necessarily comprehending the underlying mathematical formulation. Designers are allowed to interactively sculpt shapes in a natural and predictable way using a variety of force-based tools.

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 [4, 11], time is fundamental to physics-based design. Additional advantages accrue through the use of real-time dynamics:

- An "instantaneous" optimizer (if such a thing existed) can produce some kinematic deformation 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, non-smooth 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 augments the geometry with time, mass, force, and constraint. Dynamic

models produce smooth, natural motions that 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 that 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 relevant throughout the entire design and manufacturing process.

4 Background Review

The physics-based modeling paradigm is motivated by prior research aimed at applying the deformable modeling approach to shape design. Free-form deformable models, initially introduced to computer graphics in [36] and further developed in [35, 26, 32, 20, 33, 14] are particularly relevant. They are also useful for user interfaces, virtual reality, medical image processing, and computer animation. 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. Until recently, however, researchers in the fields of computer vision and graphics have been devoting most of their endeavors to the investigation of constraints, articulated rigid or non-rigid 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 [35]. Celniker and Gossard developed an interesting prototype system [5] for interactive free-form design based on the finite-element optimization of energy functionals proposed in [35]. 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 [3], and in [2], they proposed the use of B-splines for this purpose. Subsequently, Celniker and Welch investigated deformable B-splines with linear constraints [6].

Welch and Witkin extended the approach to trimmed hierarchical B-splines (see also [9]) for interactive modeling of free-form surface with constrained variational optimization [41]. The traditional control point approach is intuitive by allowing a conceptually simple change. However, to enforce a precise modification, many –even all– 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 [32]. Through the use of the energy minimization principles the constrained fractal can be applied to synthesize realistic terrain surface from sparse elevation data. Multi-resolution 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 [38]. 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 [20, 19, 18]. 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 [15]. 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 [33]. 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 prior work [2, 6, 41], deformable B-spline curves and surfaces are designed by imposing shape criteria via the minimization of energy functionals subject to hard or soft geometric constraints. These constraints

are imposed through Lagrange multipliers or penalty methods, respectively. The same techniques are applicable to D-NURBS [37]. Compared to deformable B-splines, however, D-NURBS are capable of representing a wider variety of free-form shapes, as well as standard analytic shapes. Previous models solve static equilibrium problems, or involve simple linear dynamics with diagonal (arbitrarily lumped) mass and damping matrices [6].

D-NURBS are a more sophisticated dynamic model derived through the systematic use of Lagrangian mechanics and finite element analysis without resorting to any of the *ad hoc* assumptions of prior schemes [37]. D-NURBS control points and associated weights are generalized coordinates in the Lagrangian equations of motion. From a physics-based modeling point of view, the existence of weights makes the NURBS geometry substantially more challenging than B-spline geometry. Since the NURBS rational basis functions are functionally dependent on the weights, D-NURBS dynamics are generally nonlinear, and the mass, damping, and stiffness matrices must be recomputed at each simulation time step. Fortunately, this does not preclude interactive performance on current graphics workstations, at least for the size of surface models that appear in our demonstrations. Because our dynamic models allow "non-discrete" mass and damping distributions, we obtain banded mass and damping matrices. We apply numerical quadrature to the underlying NURBS basis functions to compute efficiently the integral expressions for the matrix entries.

5 D-NURBS Physics and Implementation

By marrying advanced geometric modeling with computational physics, we have proposed and 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 [37], (ii) tensor product D-NURBS surfaces [37], (iii) swung D-NURBS surfaces [29], and (iv) triangular D-NURBS surfaces [28, 30]. The shape parameters of geometric NURBS play the role of generalized (physical) coordinates in dynamic NURBS. We introduce time, mass, and deformation energy into the standard NURBS formulation and employ Lagrangian dynamics to arrive at the system of nonlinear ordinary differential equations that govern the shape and motion of D-NURBS. In the following context, we shall only summarize the formulation of D-NURBS curves. The detailed mathematics and finite element implementation of other D-NURBS objects have been documented elsewhere in [37, 29, 28, 30].

A D-NURBS curve extends the geometric NURBS curve definition by explicitly incorporating time. It can be defined as a function of both the parametric variable u and time t:

$$\mathbf{c}(u,t) = \frac{\sum_{i=0}^{n} \mathbf{p}_i(t) w_i(t) B_{i,k}(u)}{\sum_{i=0}^{n} w_i(t) B_{i,k}(u)}, \tag{1}$$

where the $B_{i,k}(u)$ are the usual recursively defined piecewise basis functions [8], $\mathbf{p}_i(t)$ are the $n+1$ control points, and $w_i(t)$ are associated non-negative weights. Assuming basis functions of degree $k-1$, the curve has $n+k+1$ knots t_i in non-decreasing sequence: $t_0 \leq t_1 \leq \ldots \leq t_{n+k}$. 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 $\mathbf{p}_n$.

To simplify notation, we define the vector of generalized coordinates $\mathbf{p}_i(t)$ and weights $w_i(t)$ as

$$\mathbf{p}(t) = \begin{bmatrix} \mathbf{p}_0^\top & w_0 & \cdots & \mathbf{p}_n^\top & w_n \end{bmatrix}^\top,$$

where $^\top$ denotes transposition. We then express the curve (1) as $\mathbf{c}(u, \mathbf{p})$ in order to emphasize its dependence on $\mathbf{p}$ whose components are functions of time.

The velocity of the D-NURBS curve is

$$\dot{\mathbf{c}}(u, \mathbf{p}) = \mathbf{J}\dot{\mathbf{p}}, \tag{2}$$

where the overstruck dot denotes a time derivative and $\mathbf{J}(u, \mathbf{p})$ is the Jacobian matrix. Because $\mathbf{c}$ is a 3-component vector-valued function and $\mathbf{p}$ is a $4(n+1)$ dimensional vector, $\mathbf{J}$ is the $3 \times 4(n+1)$ matrix

$$\mathbf{J} = \begin{bmatrix} \cdots & \begin{bmatrix} \frac{\partial \mathbf{c}_x}{\partial \mathbf{p}_{i,x}} & 0 & 0 \\ 0 & \frac{\partial \mathbf{c}_y}{\partial \mathbf{p}_{i,y}} & 0 \\ 0 & 0 & \frac{\partial \mathbf{c}_z}{\partial \mathbf{p}_{i,z}} \end{bmatrix} & \frac{\partial \mathbf{c}}{\partial w_i} & \cdots \end{bmatrix}, \tag{3}$$

where

$$\frac{\partial \mathbf{c}_x}{\partial \mathbf{p}_{i,x}} = \frac{\partial \mathbf{c}_y}{\partial \mathbf{p}_{i,y}} = \frac{\partial \mathbf{c}_z}{\partial \mathbf{p}_{i,z}} = \frac{w_i B_{i,k}}{\sum_{j=0}^{n} w_j B_{j,k}};$$

$$\frac{\partial \mathbf{c}}{\partial w_i} = \frac{\sum_{j=0}^{n} (\mathbf{p}_i - \mathbf{p}_j) w_j B_{i,k} B_{j,k}}{(\sum_{j=0}^{n} w_j B_{j,k})^2}.$$

The subscripts x, y, and z denote the components of a 3-vector. Furthermore, we can express the curve as the product of the Jacobian matrix and the generalized coordinate vector:

$$\mathbf{c}(u, \mathbf{p}) = \mathbf{J}\mathbf{p}. \tag{4}$$

The proof of (4) can be found elsewhere [37].

The equations of motion of our dynamic NURBS curves are derived from the work-energy version of Lagrangian dynamics [10]. Applying the Lagrangian formulation to D-NURBS curves, we obtain the

second-order nonlinear equations of motion

$$\mathbf{M}\ddot{\mathbf{p}} + \mathbf{D}\dot{\mathbf{p}} + \mathbf{K}\mathbf{p} = \mathbf{f}_p + \mathbf{g}_p, \tag{5}$$

where the mass matrix $\mathbf{M}(\mathbf{p})$, the damping matrix $\mathbf{D}(\mathbf{p})$, and the stiffness matrix $\mathbf{K}(\mathbf{p})$ can all be formulated explicitly (refer to [37, 29, 30] for the details). The $N \times N$ mass and damping matrices are are

$$\mathbf{M}(\mathbf{p}) = \int \mu \mathbf{J}^\top \mathbf{J}\, du; \quad \mathbf{D}(\mathbf{p}) = \int \gamma \mathbf{J}^\top \mathbf{J}\, du; \tag{6}$$

where $\mu(u)$ is the prescribed mass density function over the parametric domain of the curve and $\gamma(u)$ is the prescribed damping density function. To define an elastic potential energy for the curve, we adopt the *thin-plate under tension* energy model [34, 5, 41, 12, 37]. This yields the $N \times N$ stiffness matrix

$$\mathbf{K}(\mathbf{p}) = \int (\alpha_1 \mathbf{J}_u^\top \mathbf{J}_u + \beta_1 \mathbf{J}_{uu}^\top \mathbf{J}_{uu})\, du, \tag{7}$$

where the subscripts on $\mathbf{J}$ denote parametric partial derivatives. The $\alpha_1(u)$ and $\beta_1(u)$ are elasticity functions that control tension and rigidity, respectively, along the parametric coordinate direction. Other energies are applicable, including the non-quadratic, curvature-based energies [36, 17]. The generalized force $\mathbf{f}_p(\mathbf{p}) = \int \mathbf{J}^\top \mathbf{f}(u,t)\, du$ is obtained through the principle of virtual work [10] done by the applied force distribution $\mathbf{f}(u,t)$. Because of the geometric nonlinearity, generalized inertial forces $\mathbf{g}_p(\mathbf{p})$ are also associated with the models (see [37, 29] for the details).

The equations of motion (5) that determine the evolution of $\mathbf{p}$ cannot be solved analytically in general. Instead, we pursue an efficient numerical implementation using finite-element techniques [13]. Standard finite element codes explicitly assemble the global matrices that appear in the discrete equations of motion [13]. We use an iterative matrix solver to avoid the cost of assembling the global $\mathbf{M}$, $\mathbf{D}$, and $\mathbf{K}$. In this way, we work with the individual element matrices and construct finite element data structures that permit the parallel computation of element matrices.

We consider a D-NURBS curve arc (or surface patch) defined by consecutive knots in the parametric domain to be a type of finite element. We define an element data structure which contains the geometric specification of the curve arc (or surface patch) element along with its physical properties, as is illustrated in Fig. 4. A complete D-NURBS curve (or surface) is then implemented as a data structure that consists of an ordered array of D-NURBS elements with additional information. The element structure includes pointers to appropriate components of the global vector $\mathbf{p}$ (control points and weights). Note that neighboring elements will share some generalized coordinates (see Fig. 4). We also allocate in each element an elemental mass, damping, and stiffness matrix, and include in the element data structure the quantities needed to

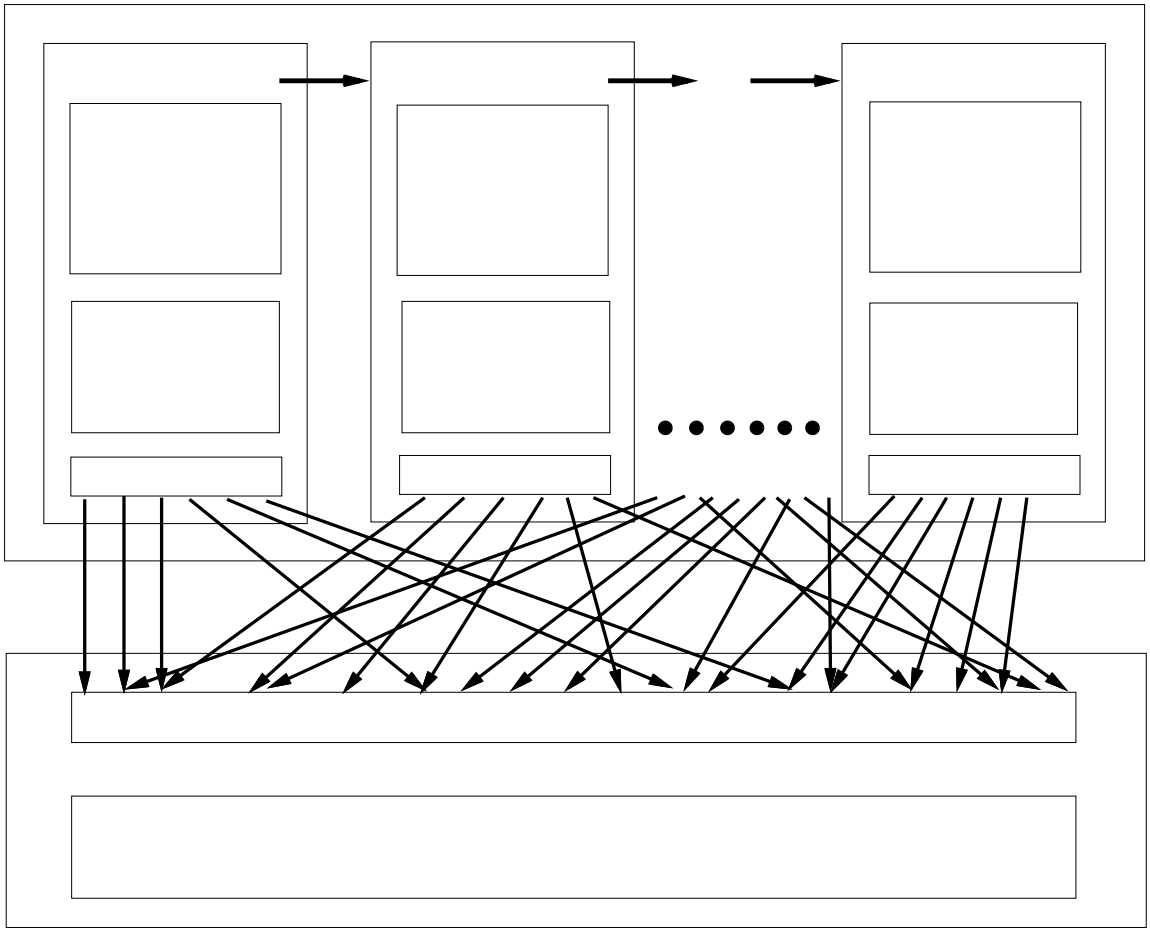

Figure 4: Element data structure of D-NURBS.

compute these matrices. These quantities include the mass $\mu(u)$, damping $\gamma(u)$, and elasticity $\alpha_1(u)$, $\beta_1(u)$ density functions, which may be represented as analytic functions or as parametric arrays of sample values. Note that, the integral expressions for the mass, damping, and stiffness matrices associated with each element are evaluated numerically using Gaussian quadrature [27]. The detailed algorithm for element matrix assembly and calculations can be found in [30].

To integrate (5) in an interactive modeling environment, it is important to provide the modeler with visual feedback about the evolving state of the dynamic model. Rather than using costly time integration methods that take the largest possible time steps, it is more crucial to provide a smooth animation by maintaining the continuity of the dynamics from one step to the next. Hence, less costly yet stable time integration methods that take modest time steps are desirable.

The state of the dynamic NURBS at time $t + \Delta t$ is integrated using prior states at time t and $t - \Delta t$. To maintain the stability of the integration scheme, we use an implicit time integration method, which employs the time integration formula

$$\left(2\mathbf{M} + \Delta t\mathbf{D} + 2\Delta t^2\mathbf{K}\right)\mathbf{p}^{(t+\Delta t)} = 2\Delta t^2(\mathbf{f}_p + \mathbf{g}_p) + 4\mathbf{M}\mathbf{p}^{(t)} - (2\mathbf{M} - \Delta t\mathbf{D})\mathbf{p}^{(t-\Delta t)} \tag{8}$$

where the superscripts denote evaluation of the quantities at the indicated times. The matrices and forces are evaluated at time t.

We employ the conjugate gradient method to obtain an iterative solution [27]. To achieve interactive simulation rates, we limit the number of conjugate gradient iterations per time step to 10. We have observed that 2 iterations typically suffice to converge to a residual of less than 10^{-3}. More than 2 iterations tend to be necessary when the physical parameters (mass, damping, tension, stiffness, applied forces) are changed significantly during dynamic simulation. Hence, our implementation permits the real-time simulation of dynamic NURBS objects on common graphics workstations.

The equations of motion allow realistic dynamics such as would be desirable for physics-based computer graphics animation. It is possible, however, to make simplifications that further reduce the computational cost of (8) to interactively sculpt larger surfaces. For example, in geometric modeling applications such as data fitting where the modeler is interested only in the final equilibrium configuration of the model, it makes sense to simplify (5) by setting the mass density function $\mu(u)$ to zero, so that the inertial terms vanish.

6 Physics-Based Geometric Design

The physics-based geometric design approach allows modeling requirements to be expressed and satisfied through the use of energies, forces, and constraints. Users may apply time-varying forces to sculpt shapes interactively or to optimally approximate data. Certain aesthetic constraints such as "fairness" are expressible in terms of elastic energies that give rise to specific stiffness matrices $\mathbf{K}$. Other constraints include position or normal specification at surface points, and continuity requirements between adjacent elements. By building D-NURBS upon the standard NURBS geometry, we allow the modeler to continue to use the whole spectrum of advanced geometric design tools that have become prevalent, among them, the imposition of geometric constraints that the final shape must satisfy. Our physics-based shape design approach [37, 29, 30] which utilizes energies, forces, and constraints has proven to be simpler and more intuitive than conventional geometric design methods (e.g., the manipulation and adjustment of control points and weights). This approach is especially attractive for triangular NURBS, because of the complexity and irregularity of their control point and knot vectors.

In the D-NURBS design scenario, sculpting tools may be implemented as applied forces. The force $\mathbf{f}(u,t)$ represents the net effect of all applied forces. Typical force functions are spring forces, repulsion forces, gravitational forces, inflation forces, etc. [36]. In practical applications, design requirements may also be posed as a set of physical parameters or as geometric constraints. Nonlinear constraints can be enforced through Lagrange multiplier techniques [16]. This approach increases the number of degrees of

freedom, hence the computational cost, by adding unknowns λ_i, known as Lagrange multipliers, which determine the magnitudes of the constraint forces. The augmented Lagrangian method [16] combines the Lagrange multipliers with the simpler penalty method [25]. The Baumgarte stabilization method [1] solves constrained equations of motion through linear feedback control [14, 37]. These techniques are appropriate for D-NURBS with nonlinear constraints. Linear geometric constraints such as point, curve, and surface normal constraints can be easily incorporated into D-NURBS by reducing the matrices and vectors in (5) to a minimal unconstrained set of generalized coordinates. They can then be implemented by applying the same numerical solver on an unconstrained subset of $\mathbf{p}$ [37].

D-NURBS have an interesting idiosyncrasy due to the weights. While the control point components of $\mathbf{p}$ may take arbitrary finite values in $\Re$, negative weights may cause the denominator to vanish at some evaluation points, causing the matrices to diverge. Although not forbidden, negative weights are not useful. We enforce positivity of weights at each simulation time step by simply projecting any weight value that has drifted below a small positive threshold back to this lower bound. Alternatively, we can give the designer the option of constraining the weights near certain desired target values w_i^0 by including in the surface energy the penalty term $c\sum(w_i - w_i^0)$, where c controls the tightness of the constraint.

7 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 are applicable to the optimal fitting of regular or scattered data [31]. The most general and often most useful case occurs with scattered data, when there are fewer or more data points than unknowns—i.e., when the solution is underdetermined or overdetermined by the data. In this case, D-NURBS can yield "optimal" solutions by minimizing the thin-plate under tension deformation energy [34]. The surfaces are optimal in the sense that they provide the smoothest curve or surface (as measured by the deformation energy) that interpolates or approximates the data [4, 21].

The data point interpolation problem amounts to a linear constraint problem when the weights are fixed, and it is amenable to the constraint techniques presented in previous sections. The optimal approximation problem can be approached in physical terms, by coupling the D-NURBS 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. 5(a-d) shows the final reconstructed shapes.

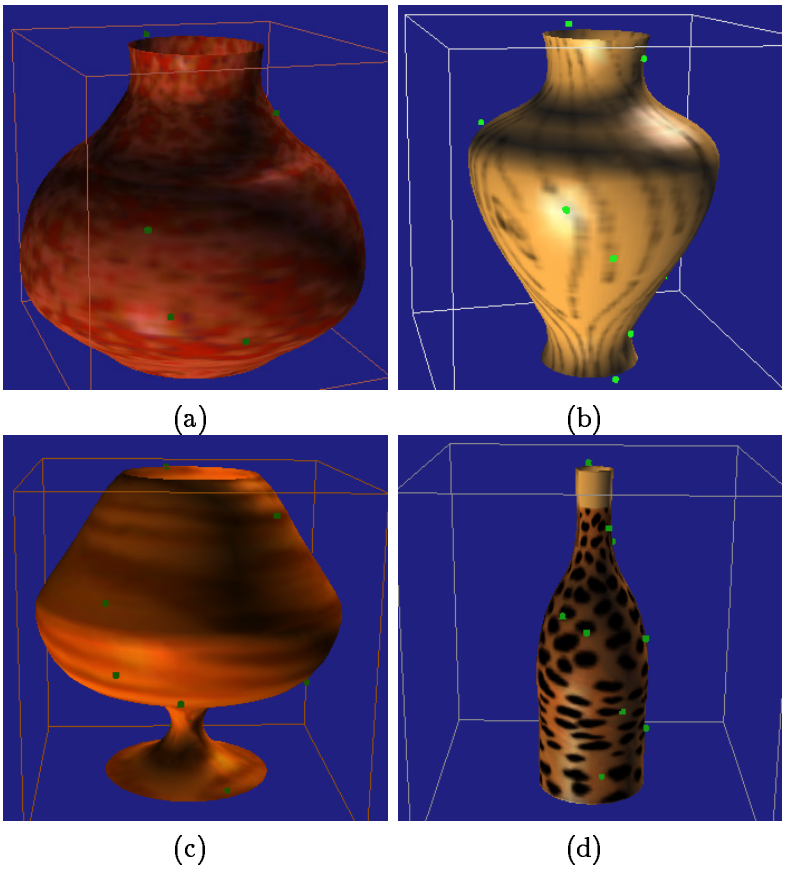

Figure 5: Four fitted shapes. (a) Pot. (b) Vase. (c) Glass. (d) Bottle.

The rounding operation is usually attempted geometrically by enforcing continuity requirements on the fillet that interpolates between two or more surfaces. By contrast, the D-NURBS can produce a smooth fillet by minimizing its internal deformation energy subject to position and normal constraints. The dynamic simulation automatically produces the desired final shape as it achieves static equilibrium. Fig. 6 demonstrates edge rounding using D-NURBS surfaces. In Fig. 6(a1), we round an edge at the intersection of two planar faces. The faces are formed using quadratic D-NURBS patches with 8×5 control points. The D-NURBS rounds the corner as it achieves the minimal energy equilibrium state shown in Fig. 6(a2). Fig. 6(b1) illustrates the rounding of a trihedral corner of a cube. The corner is represented using a quadratic D-NURBS surface with 6×5 control points. The corner is rounded with position and normal constraints along the far boundaries of the faces (Fig. 6(b2)).

In the physics-based modeling approach, not only can designers manipulate the individual degrees of freedom with conventional geometric methods, but they can also move the object or refine its shape with interactive sculpting forces. The physics-based modeling approach is ideal for interactive sculpting of surfaces. It provides direct manipulation of the dynamic surface to refine the shape of the surface through the application of interactive sculpting tools in the form of forces. Fig. 7 illustrates four shapes sculpted using spring forces. The initial open surface is generated using a quadratic triangular B-splines with 24

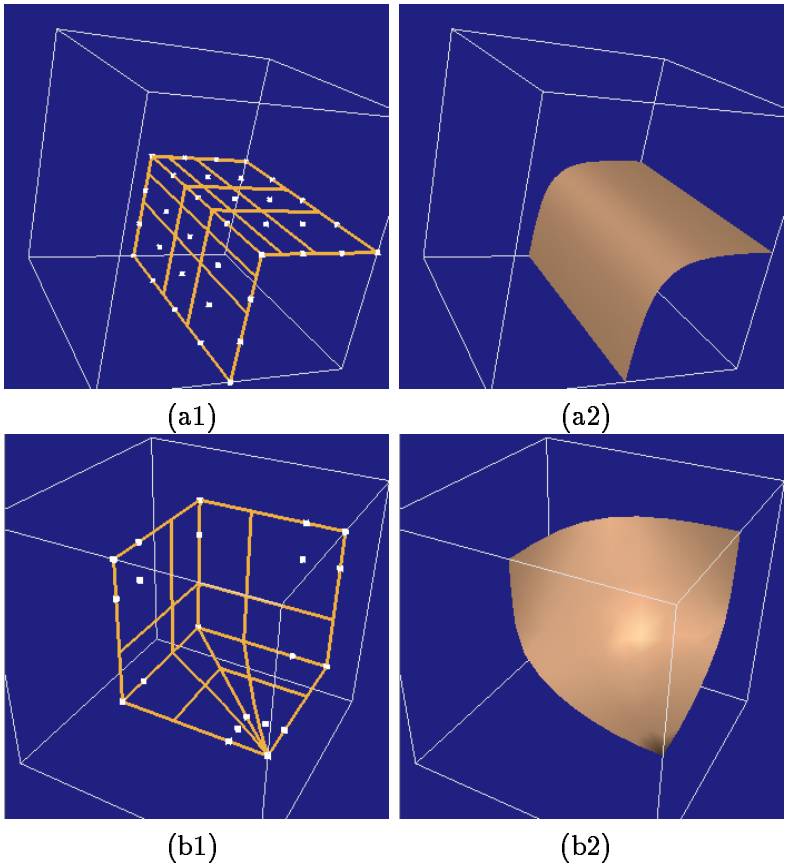

Figure 6: Solid rounding: (a) rounding an edge between polyhedral faces; (b) rounding a trihedral vertex. (a1) Initial configuration of control points and patches. (a2) Rounded D-NURBS surface in static equilibrium. (b1) Initial configuration of control points and patches. (b2) Rounded D-NURBS surface. In both examples, the control points along edges have multiplicity 2.

control points.

8 Conclusion

In this paper, we have presented D-NURBS, a physics-based generalization of geometric NURBS. D-NURBS have been derived systematically through the application of Lagrangian mechanics and implemented using concepts from finite element analysis and efficient numerical method. Based on D-NURBS theory and applications, we also proposed and presented a new physics-based geometric modeling paradigm, which generalizes well established geometric design approaches. This paradigm was the basis of a D-NURBS interactive modeling environment. The 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.

Since D-NURBS 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

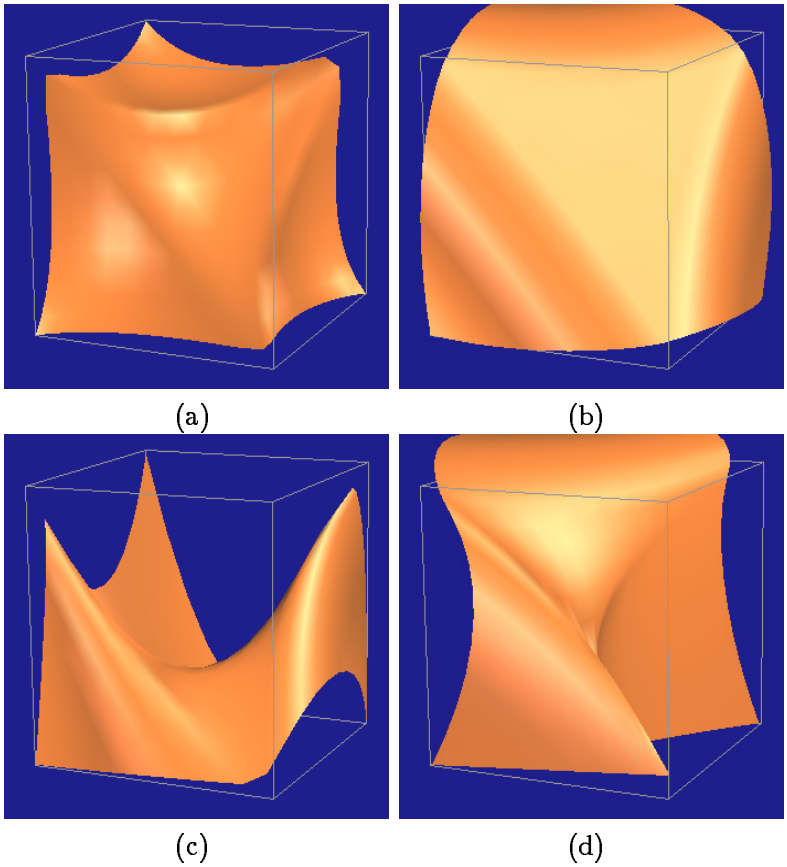

Figure 7: Interactive sculpting of an open quadratic surface into four different shapes (a–d).

high-performance graphics systems, however, the physics-based framework is poised for incorporation into commercial design systems to interactively model and sculpt complex shapes in real-time. Thus, D-NURBS can unify the features of the industry-standard geometry with the many demonstrated conveniences of interaction through physical dynamics.

References

[1] J. Baumgarte. Stabilization of constraints and integrals of motion in dynamical systems. *Comp. Meth. in Appl. Mech. and Eng.*, 1:1–16, 1972.

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

[3] 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.

[4] B. Brunnett, H. Hagen, and P. Santarelli. Variational design of curves and surfaces. *Surveys on Mathematics for Industry*, 3:1–27, 1993.

[5] 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).

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

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

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

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

[10] B.R. Gossick. *Hamilton's Principle and Physical Systems.* Academic Press, New York and London, 1967.

[11] G. Greiner. Variational design and fairing of spline surfaces. In *Proceedings of EUROGRAPHICS'94*, pages 143–154, Blackwell, 1994. Eurographics Association.

[12] M. Halstead, M. Kass, and T. DeRose. Efficient, fair interpolation using Catmull-Clark surfaces. In *Computer Graphics* Proceedings, Annual Conference Series, Proc. ACM Siggraph'93 (Anaheim, CA, Aug., 1993), pages 35–44, 1993.

[13] H. Kardestuncer. *Finite Element Handbook.* McGraw–Hill, New York, 1987.

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

[15] 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.

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

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

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

[19] 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.

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

[21] R. Pfeifle and H.-P. Seidel. Fitting triangular B-Splines to functional scattered data. In *Proceedings of Graphics Interface'95*, pages 26–33, Morgan Kaufmann, 1995. Canadian Human-Computer Communications Society.

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

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

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

[25] J. Platt. A generalization of dynamic constraints. *CVGIP: Graphical Models and Image Processing*, 54(6):516–525, 1992.

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

[27] W. Press, B. Flanney, S. Teukolsky, and W. Verttering. *Numerical Recipes: The Art of Scientific Computing.* Cambridge University Press, Cambridge, 1986.

[28] 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.

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

[30] H. Qin and D. Terzopoulos. Triangular NURBS and their dynamic generalizations. *Computer Aided Geometric Design*, 14(4):325–347, 1997.

[31] L.L. Schumaker. Fitting surfaces to scattered data. In G.G. Lorentz, C.K. Chui, and L.L. Schumaker, editors, *Approximation Theory II*, pages 203–267. Academic Press, New York, 1976.

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

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

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

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

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

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

[38] 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.

[39] W. Tiller. Rational B-splines for curve and surface representation. *IEEE Computer Graphics and Applications*, 3(6):61–69, Sept. 1983.

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

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