Dynamic Manipulation of Triangular B-Splines

Hong Qin and Demetri Terzopoulos
Department of Computer Science, University of Toronto[1]

Keywords: *CAGD, Solid Modeling, Triangular B-splines, Dynamics, Finite Elements, Constraint-Based Design, User Interaction Techniques, Parametric Design.*

Abstract: *Triangular B-splines provide a unified representation scheme for all piecewise polynomials. They are useful for modeling a broad class of complex objects defined over arbitrary, non-rectangular domains. To date, however, they have been viewed as purely geometric primitives requiring the manual adjustment of multiple control points to design shapes. This indirect design process can be laborious and often clumsy. As an alternative, we develop a new model based on the elegant triangular B-spline geometry and principles of physical dynamics. The dynamic behavior of our model, resulting from the numerical integration of differential equations of motion, produces physically meaningful and highly intuitive shape variation. The equations govern the evolution of control points in response to applied forces and constraints. We use Lagrangian mechanics to formulate the equations of motion and finite element analysis to reduce these equations to efficient numerical algorithms. Dynamic triangular B-splines provide a systematic and unified approach for a variety of solid modeling problems including shape blending, constraint-based design, and parametric design. They also support direct manipulation and interactive sculpting of shapes using force-based tools. We demonstrate several applications of dynamic triangular B-splines, including interactive sculpting using forces and physical parameters, the fitting of unstructured data, and solid rounding with geometric and physical constraints.*

1 Introduction

Univariate B-splines have been widely used in a large variety of applications due to their many nice properties. However, the main drawback of tensor-product B-splines in solid modeling is that the underlying shape (and the parametric domain) is "topologically" rectangular. As a result, the designer is forced to model multi-sided irregular shapes using degenerate patches with deteriorated continuity conditions along their boundaries. To compensate, explicit linear and/or non-linear smoothness constraints must be enforced on the patches, complicating the design task in general.

Triangular B-splines [7] are emerging as a powerful new tool in solid modeling. Using triangular B-splines, designers can represent complex shapes over arbitrary triangulated domains with low degree piecewise polynomials which maintain high-order continuity. For example, we can construct C^1 continuous surfaces with quadratic triangular B-splines, whereas biquadratic tensor-product B-splines are necessary to achieve the same continuity. In addition, triangular B-splines retain a considerable breadth of geometric coverage. They can express topologically complex objects without degeneracy. They model smoothness as well as discontinuities by varying the distribution of knots. In particular, they subsume Bernstein-Bézier triangles as a special case with n-fold knots. Moreover, any piecewise polynomial can be represented as a linear combination of triangular B-splines [23]. Thus, triangular B-splines can provide a common representation among various modeling systems for product data exchange and representation conversion.

To date, triangular B-splines have been viewed as purely geometric primitives requiring the manual adjustment of multiple control points to design shapes. This indirect design process can be laborious and often clumsy. As an alternative, we develop a new model, *dynamic triangular B-splines*, based on the elegant triangular B-spline geometry and principles of physical dynamics. The dynamic behavior of our model, resulting from the numerical integration of differential equations of motion, produces physically meaningful and highly intuitive shape variation. Like D-NURBS [29, 22], our new model not only provides a systematic and unified approach for a variety of solid modeling problems such as shape blending, constraint-based design, parametric design and user interaction techniques, but it also supports interactive sculpting and direct manipulation using a variety of force-based tools. An important advantage of this dynamic model is that shape design may proceed interactively or automatically at the physical level, while existing geometric toolkits are concurrently applicable at the underlying geometric level.

Section 2 compares features of geometric and dynamic models. Section 3 reviews triangular B-splines. In section 4, we formulate dynamic triangular B-splines and derive their equations of motion. Section 5 considers the numerical simulation of the dynamic model using finite element techniques. We discuss the physics-based design paradigm through the use of forces and constraints in Section 6. Section 7 presents applications of dynamic triangular B-splines to interactive sculpting, scattered data fitting and shape rounding. Section 8 concludes the paper.

[1] 10 King's College Road, Toronto, Ontario, Canada, M5S 1A4
E-mail: {qin|dt}@cs.toronto.edu

2 Geometric vs Physics-Based Modeling

Using triangular B-splines for geometric modeling, the designer can benefit from arbitrary parametric domains, non-degeneracy for multi-sided surfaces, and other important features. However, the purely geometric formulation leads to serious inconveniences. In conventional free-form geometric modeling, the designer specifies shape by adjusting control points and knots. This is known as indirect shape design. The indirect design process can be clumsy and time consuming in general. It is especially difficult and laborious for triangular schemes because of the irregularity of control points and knots. Moreover, design requirements are usually specified qualitatively or quantitatively in terms of shape itself, not in terms of the degrees of freedom of any specific shape representation. For example, attempting to design a qualitatively "fair" surface that approximates unstructured quantitative 3D data through indirect design is an *ad hoc* and ambiguous endeavor.

Physics-based modeling techniques provide a means of overcoming these difficulties. In the physics-based approach, 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 may be sculpted interactively using a variety of force-based "tools." Functional design requirements may be readily implemented as deformation (fairness) energies and geometric constraints. As a dynamic model reaches equilibrium, it acts as a shape optimizer subject to the imposed constraints. Furthermore, shape design is generally a time-varying process—a designer is often interested not only in the final equilibrium shape but also in the intermediate shape variation due to parameter changes. Since time is fundamental to the physics-based formulation, dynamic models can continuously evolve control points in response to applied forces to produce meaningful and predictable shape variation.

The new model developed in this paper is closely related to D-NURBS [29, 22], a fully dynamic physics-based generalization of non-triangular NURBS, which allow a designer to interactively sculpt and directly manipulate shapes in a natural and predictable way using a variety of force-based tools. The physics-based shape design paradigm has been explored by a growing number of researchers. Terzopoulos and Fleischer [27] demonstrated simple interactive sculpting using viscoelastic and plastic models. Bloor and Wilson [3] demonstrated free-form shape design using energy-minimizing tensor product B-splines subject to constraints. Celniker and Gossard [4] developed an interesting prototype system for interactive free-form design based on the triangle-finite-element optimization of energy functionals. They combined geometric constraints with sculpting operations based on forces and loads. Welch and Witkin [31] optimized similar energies based on trimmed hierarchical B-splines. Moreton and Sequin [19] interpolated a minimum energy curve network with quintic Bézier patches by minimizing the variation of curvature.

3 Geometry of Triangular B-splines

In this section, we first review triangular B-splines and present their formulation. We then describe their analytic and geometric properties (see [7, 24, 9, 12] for the details).

3.1 Background

The theoretical foundation of triangular B-splines was built upon the multivariate simplex spline in approximation theory. Motivated by an idea from Curry and Schoenberg of a geometric interpretation of univariate B-splines, De Boor [8] first presented a brief description of multivariate B-splines (simplex splines). Since then, the theory of multivariate B-splines has been extensively explored [17, 5, 6, 14]. The well-known recurrence relation of multivariate simplex splines was first proposed in [17]. Subsequently, Grandine [11] devised a stable evaluation algorithm. Dahmen and Micchelli [6] presented a thorough review of multivariate B-splines. From the point of view of blossoming, Dahmen, Micchelli and Seidel [7] proposed triangular B-splines which are essentially normalized simplicial splines.

In contrast to the theory, applications of multivariate B-splines are have not been extensively explored because of the complicated domain partitions involved and the time-consuming algorithms for evaluation and derivative computation, especially for high dimensional and high order problems. Nevertheless, 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 [30] discussed applications of bivariate quadratic simplicial spline finite elements and derived B-spline differentiation and inner product formulas. Auerbach et al. [1] use bivariate simplicial B-splines to fit geologic surfaces using scattered data by adjusting the triangulation of the parametric domain in accordance with the distribution of data points. Recently, the first experimental geometric modeling software based on the triangular B-splines was developed, demonstrating the practical feasibility of the basic algorithms [9].

3.2 Multivariate Simplex Splines

The basis functions of multivariate simplex splines can be defined either analytically or recursively. An s-variate simplex spline $M(\mathbf{x}|\{\mathbf{x}_0,\ldots,\mathbf{x}_m\})$ can be defined as a function of $\mathbf{x}\in R^s$ over the convex hull of a point set $\{\mathbf{x}_0,\ldots,\mathbf{x}_m\}$, depending on the $m+1$ knots $\mathbf{x}_i\in R^s$, $i=0,\ldots,m$, $(m\geq s)$. It is a piecewise polynomial with degree $d=m-s$ satisfying

$$\int_{R^s} f(\mathbf{x})\,M(\mathbf{x}|\{\mathbf{x}_0,\ldots,\mathbf{x}_m\})\,d\mathbf{x} = (m-s)!\int_{S^m} f(t_0\mathbf{x}_0+\ldots+t_m\mathbf{x}_m)\,dt_1\ldots dt_m, \qquad (1)$$

where f is an arbitrary integrable function defined over the region which covers the convex hull spanned by the knot sequences $\mathbf{x}_i$. The region S^m is the standard m-simplex:

$$S^m=\{(t_1,\ldots,t_m)|\sum_{i=0}^{m}t_i=1,\,t_i\geq 0\}.$$

$M(\mathbf{x}|\{\mathbf{x}_0,\ldots,\mathbf{x}_m\})$ has a very simple geometric interpretation. It is the projection of a higher dimensional simplex on a lower dimensional space. Let Δ be a m-simplex extended by $m+1$ vertices $[\mathbf{v}_0,\ldots,\mathbf{v}_m]$, $\mathbf{v}_i\in R^m$ such that the projection of $\mathbf{v}_i$ on subspace R^s is $\mathbf{x}_i$, and for arbitrary $\mathbf{x}$ define a point set

$$A_{\mathbf{x}}=\{\mathbf{v}|\mathbf{v}\in\Delta,\mathbf{v}|_{R^s}=\mathbf{x}\}\,.$$

Thus, we can explicitly formulate the basis function of s-variate simplex splines as

$$M(\mathbf{x}|\{\mathbf{x}_0,\ldots,\mathbf{x}_m\})=\frac{(m-s)!}{m!}\cdot\frac{\mathrm{vol}_{m-s}(A_{\mathbf{x}})}{\mathrm{vol}_m(\Delta)}, \qquad (2)$$

where vol_k denotes the k-dimensional volume of certain sets.

The basis function of multivariate simplex splines may also be formulated recursively, which facilitates evaluation and derivative computation: When $m=s$,

$$M(\mathbf{x}|\{\mathbf{x}_0,\ldots,\mathbf{x}_m\})=\begin{cases}\frac{1}{m!\,\mathrm{vol}_s([\mathbf{x}_0,\ldots,\mathbf{x}_m])}, & x\in[\mathbf{x}_0,\ldots,\mathbf{x}_m]\\ 0 & \text{otherwise}\end{cases}$$

and when $m>s$,

$$M(\mathbf{x}|\{\mathbf{x}_0,\ldots,\mathbf{x}_m\})=\sum_{i=0}^{m}\lambda_i M(\mathbf{x}|\{\mathbf{x}_0,\ldots,\mathbf{x}_{i-1},\mathbf{x}_{i+1},\ldots,\mathbf{x}_m\}), \qquad (3)$$

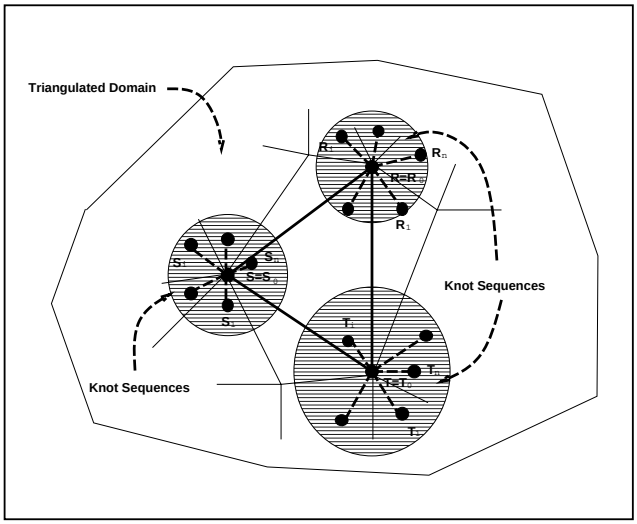


Figure 1: Knot vectors associated with each triangle in the domain.

where

$$\sum_{i=0}^{m} \lambda_i = 1; \qquad \sum_{i=0}^{m} \lambda_i \mathbf{x}_i = \mathbf{x}.$$

Note that when $m = s$ the basis function is discontinuous along the boundary of the convex hull $[\mathbf{x}_0, \ldots, \mathbf{x}_m]$; thus, the function value is not unique. Therefore, extra effort is needed to deal with the boundary evaluation (see [9] for the concept of semi-open convex hull in the context of bivariate B-splines). Second, when $m > s$ the barycentric coefficients are not unique. An efficient method frequently used in applications is to make at least $m - s$ of the λ_i vanish, while the remaining ones are taken as positive barycentric coordinates to obtain stable and fast evaluation. Similarly, the directional derivative $D_{\mathbf{w}}$ of multivariate simplex splines can be recursively formulated as

$$D_{\mathbf{w}} M(\mathbf{x}|\{\mathbf{x}_0, \ldots, \mathbf{x}_m\}) = \mathbf{w}^\top \nabla M = (m-s) \sum_{i=0}^{m} \mu_i M(x|\{\mathbf{x}_0, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_m\}), \qquad (4)$$

where μ_i are coefficients satisfying

$$\sum_{i=0}^{m} \mu_i = 0; \qquad \sum_{i=0}^{m} \mu_i \mathbf{x}_i = \mathbf{w}.$$

Again, these scalar coefficients are not unique. An efficient algorithm to evaluate derivatives is obtained by setting $\mu_i = D_{\mathbf{w}} \lambda_i$, $i = 0, \ldots, m$, where λ_i is defined in (3).

3.3 Triangular B-splines

The triangular B-spline is essentially a normalized bivariate simplex spline. Let $T = \{\Delta(\mathbf{i}) = [\mathbf{r}, \mathbf{s}, \mathbf{t}] | \mathbf{i} = (i_0, i_1, i_2) \in Z_+^3\}$ be an arbitrary triangulation of the planar parametric domain, where i_0, i_1, and i_2 denote indices of $\mathbf{r}$, $\mathbf{s}$, and $\mathbf{t}$ in the vertex array of the triangulation, respectively. For each vertex $\mathbf{v}$ in the triangulated domain, we then assign a knot sequence (also called a cloud of knots) $[\mathbf{v} = \mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n]$ (which are inside the shaded circles in Fig. 1). Next, we define a convex hull

$$V_{\mathbf{i},\beta} = \{\mathbf{r}_0, \ldots, \mathbf{r}_{\beta_0}, \mathbf{s}_0, \ldots, \mathbf{s}_{\beta_1}, \mathbf{t}_0, \ldots, \mathbf{t}_{\beta_2}\}$$

where subscript $\mathbf{i}$ is a triangle index and β is a triplet $(\beta_0, \beta_1, \beta_2)$ such that $\beta_0 + \beta_1 + \beta_2 = n$. Thus, the bivariate simplex spline $M(\mathbf{u}|V_{\mathbf{i},\beta})$ with degree n over $V_{\mathbf{i},\beta}$ can be defined recursively according to (3), where $\mathbf{u} = (u, v)$ is a 2-vector on the triangulated domain. We then define a bivariate B-spline basis function as

$$N_{\mathbf{i},\beta}(\mathbf{u}) = d(\mathbf{r}_{\beta_0}, \mathbf{s}_{\beta_1}, \mathbf{t}_{\beta_2}) M(\mathbf{u}|V_{\mathbf{i},\beta}), \qquad (5)$$

where $d(\mathbf{r}_{\beta_0}, \mathbf{s}_{\beta_1}, \mathbf{t}_{\beta_2})$ is twice the area of $\Delta(\mathbf{r}_{\beta_0}, \mathbf{s}_{\beta_1}, \mathbf{t}_{\beta_2})$, which can be explicitly expressed as

$$d(\mathbf{r}_{\beta_0}, \mathbf{s}_{\beta_1}, \mathbf{t}_{\beta_2}) = \left| \det \begin{pmatrix} 1 & 1 & 1 \\ \mathbf{r}_{\beta_0} & \mathbf{s}_{\beta_1} & \mathbf{t}_{\beta_2} \end{pmatrix} \right|.$$

Like the ordinary tensor-product B-spline, the triangular B-spline surface of degree n defined over an arbitrary triangulated domain is the combination of a set of basis functions with control points $\mathbf{p}_{\mathbf{i},\beta}$:

$$\mathbf{s}(\mathbf{u}) = \sum_{\mathbf{i}} \sum_{|\beta|=n} \mathbf{p}_{\mathbf{i},\beta} N_{\mathbf{i},\beta}(\mathbf{u}). \qquad (6)$$

3.4 Properties

Triangular B-splines are ideal for solid modeling applications because of their many nice properties, among them, 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 position. 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 [12], they provide a unified representation scheme for polynomial models with arbitrary topology.

As was stated in the introductory sections, design techniques based on triangular B-splines include the specification of a control point vector and knot sequences, or interpolation/approximation of data points to generate the initial shape. The initial shape is then refined into the final desired shape through interactive adjustment of the control point and knot array. Because of the irregularity of triangulation vertices and knot sequences, the refinement process is *ad hoc* and can become extremely tedious and laborious. Hence, the geometric power of triangular B-splines can be offset by the inconveniences of the conventional geometric design methodology. To ameliorate matters, we consider a physics-based generalization of triangular B-splines.

4 Dynamic Triangular B-Splines

In this section, we formulate our dynamic model based on triangular B-splines. We augment standard triangular B-splines with mass, dissipation, and deformation energy, and employ Lagrangian dynamics to derive their equations of motion. The control points of the geometric model in Section 3 become generalized (physical) coordinates in the dynamic model.

4.1 Geometry and Kinematics

The dynamic triangular B-splines extend the geometric triangular B-splines in (6) first by explicitly incorporating time. They are a function of both the parametric variable $\mathbf{u}$ and time t:

$$\mathbf{s}(\mathbf{u}, t) = \sum_{\mathbf{i}} \sum_{|\beta|=n} \mathbf{p}_{\mathbf{i},\beta}(t) N_{\mathbf{i},\beta}(\mathbf{u}). \qquad (7)$$

To simplify the notation, we define the vector of generalized coordinates (control points) $\mathbf{p}_{\mathbf{i},\beta}$ as

$$\mathbf{p} = [\ldots, \mathbf{p}_{\mathbf{i},\beta}^{\top}, \ldots]^{\top}.$$

We then express (7) as $\mathbf{s}(\mathbf{u}, \mathbf{p})$ in order to emphasize its dependence on $\mathbf{p}$ whose components are functions of time. Thus, the velocity of the dynamic triangular B-splines is

$$\dot{\mathbf{s}}(\mathbf{u}, \mathbf{p}) = \mathbf{J}\dot{\mathbf{p}}, \tag{8}$$

where the overstruck dot denotes a time derivative and Jacobian matrix $\mathbf{J}(\mathbf{u})$ is the concatenation of the vectors $\partial \mathbf{s}/\partial \mathbf{p}_{\mathbf{i},\beta}$. Assuming m triangles in the parametric domain, β traverses $k = (n+2)!/(n!2!)$ possible triplets whose components sum to n. Because $\mathbf{s}$ is a 3-vector and $\mathbf{p}$ is an $M = 3mk$ dimensional vector, $\mathbf{J}$ is a $3 \times M$ matrix, which is expressed as

$$\mathbf{J} = \left[\ldots, \begin{bmatrix} R_{\mathbf{i},\beta} & 0 & 0 \\ 0 & R_{\mathbf{i},\beta} & 0 \\ 0 & 0 & R_{\mathbf{i},\beta} \end{bmatrix}, \ldots \right], \tag{9}$$

where

$$R_{\mathbf{i},\beta}(\mathbf{u}) = \frac{\partial \mathbf{s}_x}{\partial \mathbf{p}_{\mathbf{i},\beta,x}} = \frac{\partial \mathbf{s}_y}{\partial \mathbf{p}_{\mathbf{i},\beta,y}} = \frac{\partial \mathbf{s}_z}{\partial \mathbf{p}_{\mathbf{i},\beta,z}} = N_{\mathbf{i},\beta}(\mathbf{u}).$$

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

$$\mathbf{s}(\mathbf{u}, \mathbf{p}) = \mathbf{J}\mathbf{p}. \tag{10}$$

4.2 Lagrange Equations of Motion

We derive the equations of motion of dynamic triangular B-splines by applying Lagrangian dynamics [10]. We express the kinetic energy due to a prescribed mass distribution function $\mu(u, v)$ over the parametric domain of the surface and a Raleigh dissipation energy due to a damping density function $\gamma(u, v)$. We adopt the *thin-plate under tension* energy model [26, 4, 31, 13] in order to define an elastic potential energy

$$U = \frac{1}{2} \int\int (\alpha_{1,1}\mathbf{s}_u^2 + \alpha_{2,2}\mathbf{s}_v^2 + \beta_{1,1}\mathbf{s}_{uu}^2 + \beta_{1,2}\mathbf{s}_{uv}^2 + \beta_{2,2}\mathbf{s}_{vv}^2)\, du\, dv.$$

The subscripts on $\mathbf{s}$ denote parametric partial derivatives. The $\alpha_{i,j}(u, v)$ and $\beta_{i,j}(u, v)$ are elasticity functions which control tension and rigidity, respectively. Other energies are applicable, including, at greater computational cost, the non-quadratic, curvature-based energies in [28, 19]). Applying the Lagrangian formulation, we obtain the second-order equations of motion

$$\mathbf{M}\ddot{\mathbf{p}} + \mathbf{D}\dot{\mathbf{p}} + \mathbf{K}\mathbf{p} = \mathbf{f}_p, \tag{11}$$

where the mass matrix is

$$\mathbf{M} = \int\int \mu \mathbf{J}^{\top}\mathbf{J}\, du\, dv,$$

the damping matrix is

$$\mathbf{D} = \int\int \gamma \mathbf{J}^{\top}\mathbf{J}\, du\, dv,$$

and the stiffness matrix is

$$\mathbf{K} = \int\int (\alpha_{1,1}\mathbf{J}_u^{\top}\mathbf{J}_u + \alpha_{2,2}\mathbf{J}_v^{\top}\mathbf{J}_v + \beta_{1,1}\mathbf{J}_{uu}^{\top}\mathbf{J}_{uu} + \beta_{1,2}\mathbf{J}_{uv}^{\top}\mathbf{J}_{uv} + \beta_{2,2}\mathbf{J}_{vv}^{\top}\mathbf{J}_{vv})\, du\, dv.$$

All are $M \times M$ matrices. The generalized force obtained through the principle of virtual work [10] done by the applied force distribution $\mathbf{f}(u, v, t)$ is

$$\mathbf{f}_p = \int\int \mathbf{J}^{\top}\mathbf{f}(u, v, t)\, du\, dv.$$

The derivation of (11) proceeds as for D-NURBS (see [29] for the details).

5 Finite Element Implementation

The evolution of the vector of generalized coordinates $\mathbf{p}(t)$, determined by (11) with physical parameter dependent matrices, cannot be solved analytically in general. We pursue an efficient numerical implementation using finite-element techniques [15].

Standard finite element codes explicitly assemble the global matrices that appear in the discrete equations of motion [15]. We use an iterative matrix solver to avoid the cost of assembling the global $\mathbf{M}$, $\mathbf{D}$, and $\mathbf{K}$, working instead with the individual element matrices. We construct finite element data structures that permit the parallel computation of element matrices.

5.1 Element Data Structure

We define an element data structure which contains the geometric specification of the triangular patch element along with its physical properties. In each element, we allocate an elemental mass, damping, and stiffness matrix, and include the quantities such as the mass $\mu(u, v)$, damping $\gamma(u, v)$, and elasticity $\alpha_{i,j}(u, v)$, $\beta_{i,j}(u, v)$ density functions. A complete dynamic triangular B-spline then consists of an ordered array of elements with additional information. The element structure includes pointers to appropriate components of the global vector $\mathbf{p}$. Neighboring elements will share some generalized coordinates.

5.2 Calculation of Element Matrices

The integral expressions for the mass, damping, and stiffness matrices associated with each element are evaluated numerically using Gaussian quadrature [21]. We explain the computation of the element mass matrix; the computation of the damping and stiffness matrices follow suit. Assuming the parametric domain of the element is $\Delta(\mathbf{r}, \mathbf{s}, \mathbf{t})$, the expression for entry m_{ij} of the mass matrix assumes the integral form

$$m_{ij} = \int\int_{\Delta(\mathbf{r},\mathbf{s},\mathbf{t})} \mu(u, v) f_{ij}(u, v)\, du\, dv.$$

Given integers N_g, we can find Gauss weights a_g, and parametric abscissas u_g, v_g such that m_{ij} can be approximated by

$$m_{ij} \approx \sum_{g=1}^{N_g} a_g \mu(u_g, v_g) f_{ij}(u_g, v_g).$$

We apply the recursive expression (3) (see also [17] for details) to evaluate $f_{ij}(u_g, v_g)$. In our system we choose N_g to be 7 for quadratic and cubic triangular B-splines. Note that because of the irregularity of the knot distribution, many of the f_{ij} vanish over the sub-region of $\Delta(\mathbf{r}, \mathbf{s}, \mathbf{t})$. We can further subdivide the $\Delta(\mathbf{r}, \mathbf{s}, \mathbf{t})$ to minimize the numerical error. We subdivide a triangle into 9 sub-triangles as shown in Fig. 2. We observe that matrices computed according to this subdivision lead to stable, convergent solutions. Alternatively, a more precise and expensive approach is to convert a triangular B-spline into a set of piecewise Bézier surfaces defined on a *finer* triangulation with extra knot lines (Fig. 3 illustrates the quadratic case). In general, about $10 - 100$ finer triangles are obtained for one triangle in the parametric domain for cubics and quadratics.

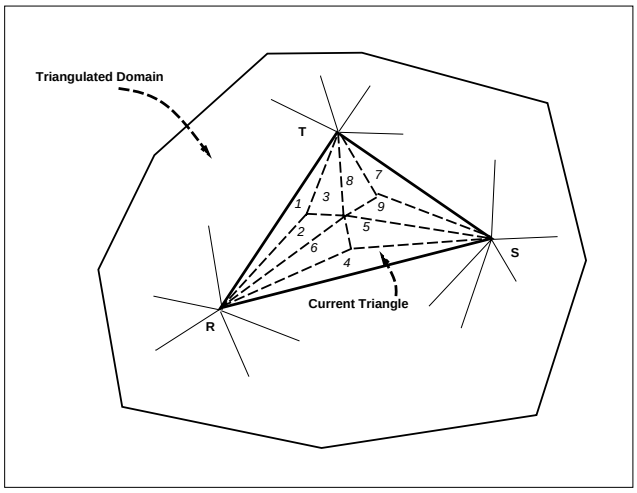


Figure 2: Nine smaller triangles for numerical quadrature.

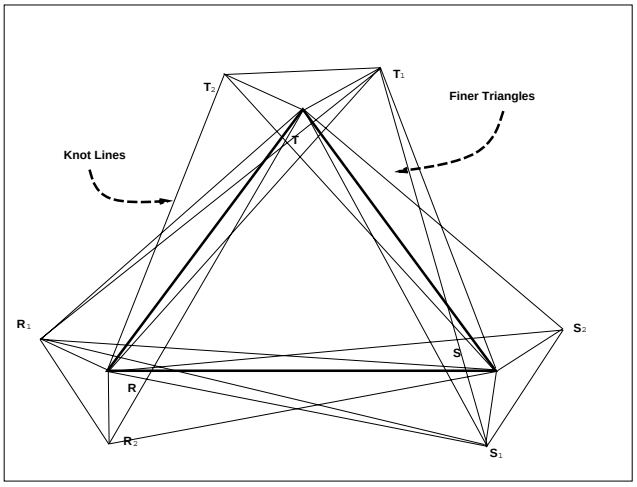


Figure 3: Finer triangulation due to intersections of knot lines.

5.3 Discrete Dynamics Equations

To integrate (11) in an interactive modeling system, 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 smoothly animated display 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 triangular B-splines 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 (especially for high stiffness configurations with large elasticity functions) we use an implicit time integration method, which employs discrete derivatives of $\mathbf{p}$ using backward differences

$$\ddot{\mathbf{p}}^{(t+\Delta t)} \approx (\mathbf{p}^{(t+\Delta t)} - 2\mathbf{p}^{(t)} + \mathbf{p}^{(t-\Delta t)})/\Delta t^2,$$

and

$$\dot{\mathbf{p}}^{(t+\Delta t)} \approx (\mathbf{p}^{(t+\Delta t)} - \mathbf{p}^{(t-\Delta t)})/2\Delta t.$$

We obtain the time integration formula

$$\begin{aligned}&\left(2\mathbf{M} + \Delta t\mathbf{D} + 2\Delta t^2\mathbf{K}\right)\mathbf{p}^{(t+\Delta t)} = \\ &\quad 2\Delta t^2\mathbf{f}_p + 4\mathbf{M}\mathbf{p}^{(t)} - (2\mathbf{M} - \Delta t\mathbf{D})\mathbf{p}^{(t-\Delta t)},\end{aligned} \tag{12}$$

where the superscripts denote evaluation of the quantities at the indicated times. The matrices and forces are evaluated at time t. Our experiments have shown that this discretization scheme produces satisfactory results. Instability due to large transient applied forces may be mitigated by adaptively reducing the size of the integration time step.

We employ the conjugate gradient method to obtain an iterative solution for $\mathbf{p}^{(t+\Delta t)}$ [21]. 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 triangular B-splines on common graphics workstations. Quadratic and cubic surfaces with about 100 control points can be simulated at real-time, interactive rates.

The equations of motion allow realistic dynamics for physics-based computer graphics animation. It is possible, however, to make simplifications that further reduce the computational cost of solving (12) to interactively sculpt larger triangular B-spline surfaces. In certain solid modeling applications where the designer is interested only in the final equilibrium configuration of the model, we can simplify (11) by setting the mass density function $\mu(u, v)$ to zero, so that the inertial terms vanish. This economizes on storage and makes the algorithm more efficient. With zero mass density, (11) reduces to

$$\mathbf{D}\dot{\mathbf{p}} + \mathbf{K}\mathbf{p} = \mathbf{f}_p. \tag{13}$$

Discretizing the derivatives of $\mathbf{p}$ in (13) with backward differences, we obtain the integration formula

$$(\mathbf{D} + \Delta t\mathbf{K})\,\mathbf{p}^{(t+\Delta t)} = \Delta t\mathbf{f}_p + \mathbf{D}\mathbf{p}^{(t)} \tag{14}$$

6 Dynamic Interaction

In the dynamic interaction paradigm, design requirements may be satisfied through the use of energies, forces, and constraints. The modeler may apply time-varying forces to sculpt shapes interactively or to optimally approximate data. Certain aesthetic constraints 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 patches. By building the dynamic model upon the standard geometry of triangular B-splines, we allow the modeler to continue to use the whole spectrum of advanced geometric tools that have become prevalent, among them, the imposition of geometric constraints that the final shape must satisfy.

6.1 Force Tools

Sculpting tools may be implemented as applied forces. The force distribution $\mathbf{f}(u, v, t)$ represents the net effect of all applied forces. Typical force functions are spring forces, repulsion forces, gravitational forces, inflation forces, etc. [28]. Consider connecting a material point (u_0, v_0) of a dynamic triangular B-spline to a point $\mathbf{d}_0$ in space with an ideal Hookean spring of stiffness k. The net applied spring force is

$$\mathbf{f}(u, v, t) = \iint k(\mathbf{d}_0 - \mathbf{s}(u, v, t))\delta(u - u_0, v - v_0)\,du\,dv, \tag{15}$$

where δ is the unit delta function. Equation (15) implies that $\mathbf{f}(u_0, v_0, t) = k(\mathbf{d}_0 - \mathbf{s}(u_0, v_0, t))$ and that it vanishes elsewhere on the surface, but we can generalize it by replacing the δ function with a smooth kernel (e.g., a unit Gaussian) to spread the applied force over a greater portion of the surface. In general, the points

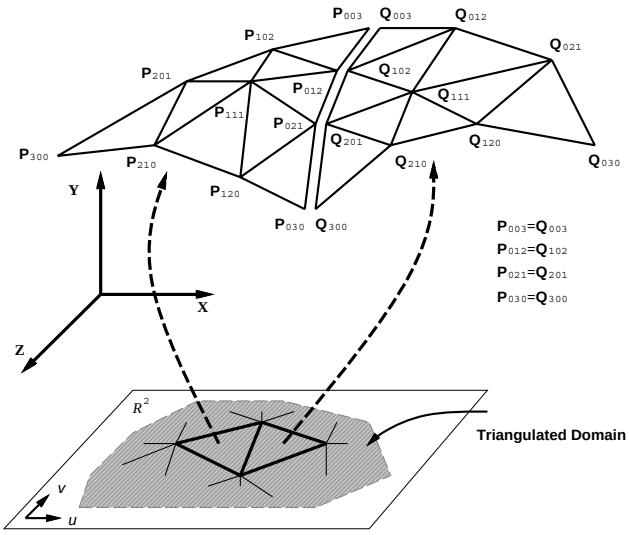


Figure 4: A constrained triangular B-Spline configuration: the continuous net.

(u_0, v_0) and $\mathbf{d}_0$ need not be constant. We can control either or both using a mouse to obtain an interactive spring force. More advanced force tools are readily implemented to intuitively manipulate geometrically intrinsic quantities such as normal and curvature anywhere on the surface.

6.2 Constraints

In practical applications, design requirements may be posed as a set of physical parameters or as geometric constraints. Nonlinear constraints can be enforced through Lagrange multiplier techniques [18, 25, 20]. 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 [18] combines the Lagrange multipliers with the simpler penalty method [20]. The Baumgarte stabilization method [2] solves constrained equations of motion through linear feedback control (see also [16, 29]). These techniques are appropriate to enforce constraints on dynamic triangular B-splines.

Linear geometric constraints such as point, curve, and surface normal constraints can be easily incorporated into dynamic triangular B-splines by reducing the matrices and vectors in (11) to a minimal unconstrained set of generalized coordinates. For example, we arrive at the *continuous net* [9] (which is a special case of general triangular B-splines) by constraining respective control points along common boundaries of two adjacent triangles in the parametric triangulation (Fig. 4). Linear constraints can be implemented by applying the same numerical solver to an unconstrained subset of $\mathbf{p}$. See [29] for a detailed discussion on linear constraints.

7 Solid Modeling Applications

We have developed prototype modeling software based on dynamic triangular B-splines. We have adopted the data structure, file, and rendering formats of existing geometric triangular B-spline software [9]. To implement the Lagrangian dynamics model on top of this software, we have had to implement a new algorithm for simultaneously evaluating non-zero basis functions and their derivatives up to second order at arbitrary domain points for finite element assembly and dynamic simulation. Our parallelized iterative numerical algorithm takes advantage of an SGI Iris 4D/380VGX multiprocessor.

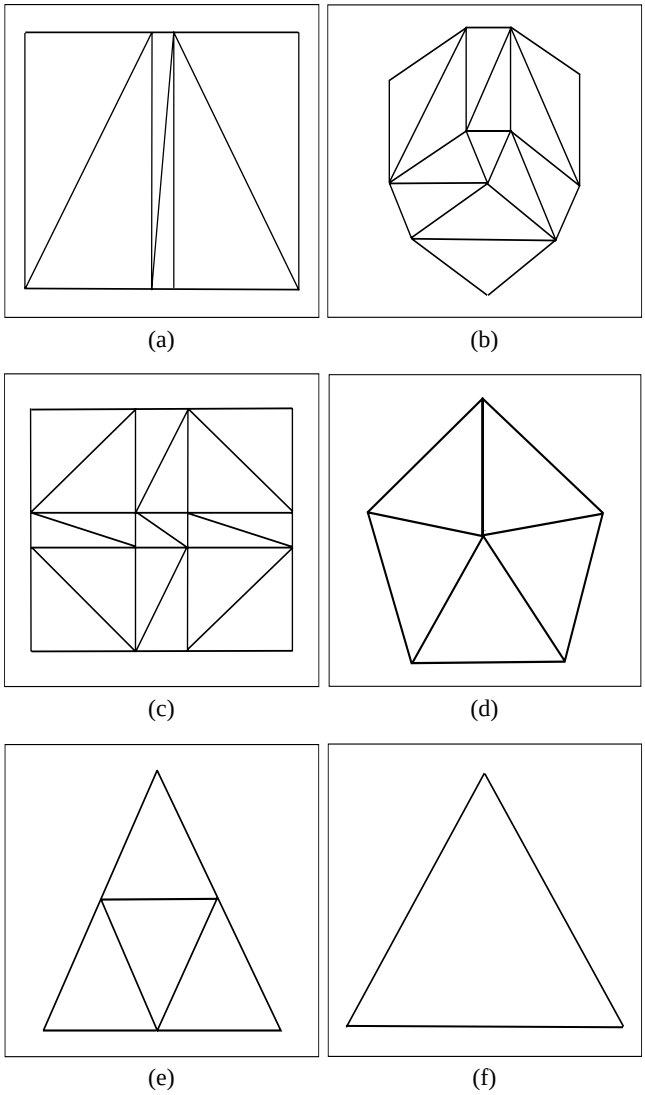


Figure 5: Domain triangulation of surfaces used in the examples (see text). (a) An edge. (b) A trihedral corner. (c) A bevel joint. (d) A pentagonal surface. (e) An open quadratic surface. (f) A cubic patch.

In accordance with the physics-based paradigm, users can sculpt surface shapes through the use of forces. They can satisfy design requirements by adjusting the internal physical parameters such as the mass, damping, and stiffness densities, along with force gain factors. Linear constraints such as the freezing of control points have been associated with physics-based toolkits in our prototype system. Local geometric constraints can be used to achieve real-time local manipulation for interactive sculpting of extremely complex objects.

In the following sections we demonstrate applications of dynamic triangular B-splines to interactive sculpting, solid rounding, and scattered data fitting. Table 1 specifies the physical parameters used in the subsequent experiments. Fig. 5 illustrates the parametric domain triangulation of the various surfaces used in these experiments.

7.1 Rounding

The rounding operation is usually attempted geometrically by enforcing continuity requirements on the fillet which interpolates be-

Applications	*Physical Parameters*								
	μ	γ	$\alpha_{1,1}$	$\alpha_{2,2}$	$\beta_{1,1}$	$\beta_{1,2}$	$\beta_{2,2}$	Δt	k
Edge rounding	0.0	500.0	1000.0	0.0	1.0	0.0	0.0	0.04	0.0
Corner rounding	0.0	50.0	1000.0	1000.0	10.0	10.0	10.0	0.04	0.0
Bevel rounding	0.0	25.0	100.0	100.0	0.0	0.0	0.0	0.04	0.0
Hill fitting	0.0	10.0	0.0	0.0	10.0	10.0	10.0	0.04	1000.0
Convex/Concave fitting	0.0	5.0	0.0	0.0	5.0	5.0	5.0	0.04	2000.0
Mountain/Valley fitting	0.0	25.0	0.0	0.0	1.0	1.0	1.0	0.04	2000.0
Quadratic object sculpting	5.0	25.0	10.0	10.0	1000.0	1000.0	1000.0	0.04	2000.0
Cubic patch sculpting	5.0	10.0	100.0	100.0	10.0	10.0	10.0	0.04	1000.0

Table 1: Physical parameters used in the examples. Parameter k denotes the stiffness of the spring force.

tween two or more surfaces. By contrast, the dynamic triangular B-spline surface 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 equilibrium.

Fig. 6 demonstrates the rounding of a sharp edge represented by a quadratic triangular B-spline surface with 36 control points. The sharp edge can be represented exactly with multiple control points. By restricting the control polygon to be a continuous net, we reduced the number of control points to 21. The initial wireframe and shaded shapes are shown in Fig. 6(a–b). After initiating the physical simulation, the sharp edges are rounded as the final shape equilibrates into the minimal energy state shown in Fig. 6(c).

Fig. 7 illustrates the rounding of a trihedral corner of a cube. The corner is represented using a quadratic triangular B-spline with 78 control points. The initial wireframe and shaded shapes are demonstrated in Fig. 7(a–b). The rounding operation is applied in the vicinity of three sharp edges. The sharp edges and corner are rounded with position and normal constraints along the far boundaries of the faces shown in Fig. 7(c).

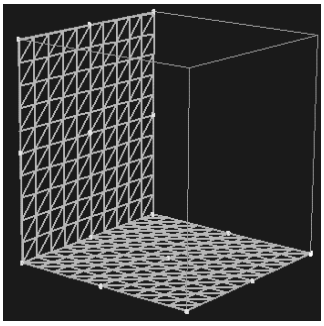
Fig. 8 shows a rounding example involving a bevel joint. The bevel joint is a quadratic triangular B-spline with 108 control points. The initial right-angle joint and the final rounded shapes are shown in Fig. 8(a–c).

7.2 Scattered Data Fitting

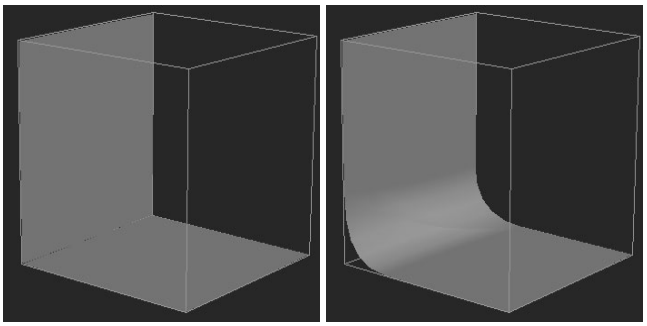
A useful modeling technique, generally known as scattered data fitting, is based on fitting surfaces to unstructured constraints. Interesting situations arise when there are fewer or more data points than there are degrees of freedom in the model, leading to underconstrained and overconstrained fitting problems, respectively. The inclusion of an elastic energy in our dynamic surfaces makes them applicable to such problems.

The data interpolation problem is amenable to common constraint techniques [18]. Approximation can be achieved by physically coupling the dynamic triangular B-splines to the data through Hookean spring forces (15). We interpret $\mathbf{d}_0$ in (15) as the data point (generally in $\Re^3$) and (u_0, v_0) as the parametric coordinates associated with the data point (typically the nearest material point of the surface). The spring constant c determines the closeness of fit to the data point.

We present three examples of surface fitting using dynamic triangular B-spline surfaces coupled to data points through spring forces. The initial surface is a quadratic pentagon with 30 control points defined over the pentagonal domain. Three different sets of 6 data points are shown in Fig. 9(a1–c1) along with the initial surfaces. The spring forces associated with the data points are applied to the nearest points on the surfaces. Note that the spring attachments shown in Fig. 9(a1–c1) show the initial correspondence and are not fixed during the dynamic surface fitting process. Fig. 9(a2–c2) show the final fitted surfaces.



(a)



(b) (c)

Figure 6: Rounding of an edge. (a) Initial wireframe object. (b) Initial shaded object. (c) Rounded object.

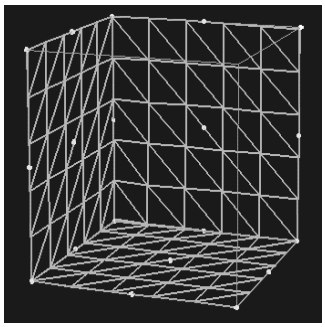
7.3 Interactive Sculpting

In the physics-based modeling approach, not only can the designer manipulate the individual degrees of freedom with conventional geometric methods, but he can also move the object or refine its shape with interactive sculpting forces.

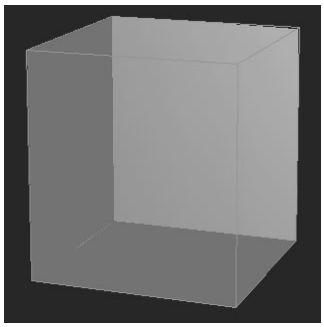
The physics-based modeling approach is ideal for interactive sculpting of surfaces. It provides direct manipulation of the dynamic surface to refine its shape through the application of interactive sculpting tools in the form of forces. Fig. 10 illustrates four shapes sculpted using spring forces. The initial open surface is generated using a quadratic B-splines with 24 control points. Second, a cubic triangular planar patch with 10 control points shown in Fig. 11(a) was dynamically manipulated into the shape shown in Fig. 11(b).

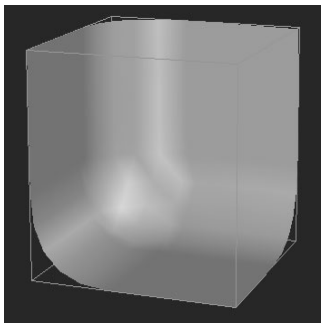
8 Conclusion

We have proposed dynamic triangular B-splines, a new free-form shape model that marries the elegant geometry of multivariate simplex splines with physical dynamics. We have applied Lagrangian dynamics to systematically derive the equations of motion of dy-



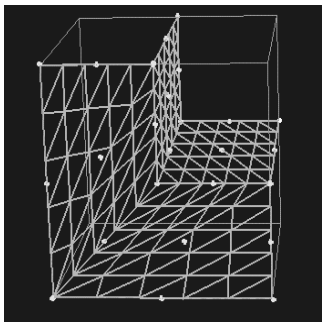
(a)



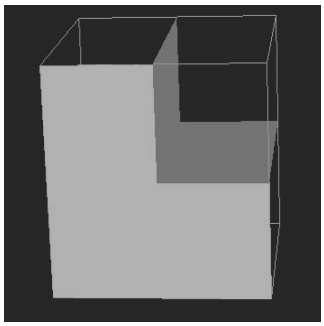


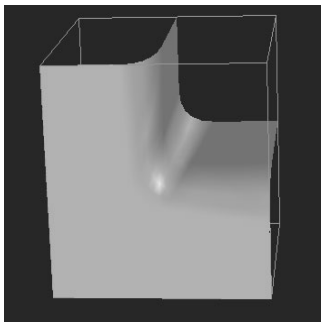
(b) (c)

Figure 7: Rounding of a trihedral corner. (a) Initial wireframe object. (b) Initial shaded object. (c) Rounded object.



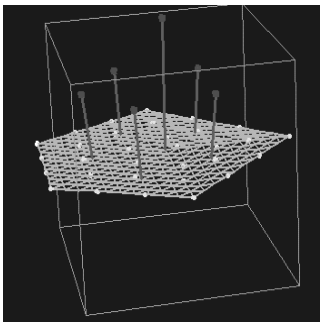
(a)

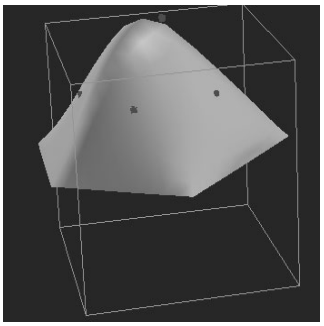




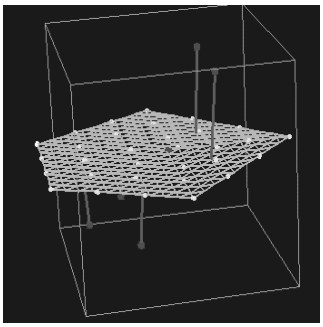
(b) (c)

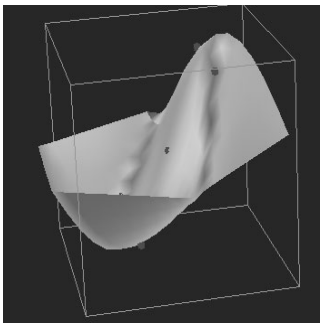
Figure 8: Rounding of a bevel joint. (a) Initial wireframe object. (b) Initial shaded object. (c) Rounded object.



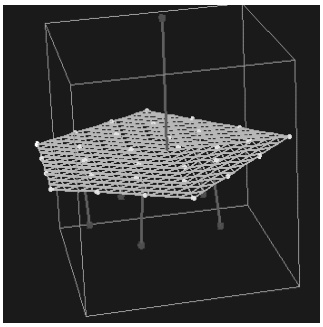


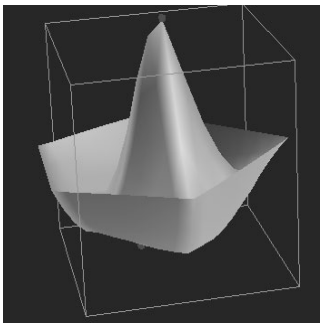
(a1) (a2)





(b1) (b2)





(c1) (c2)

Figure 9: Fitting a pentagonal surface to three different sets of scattered data. Data points and initial surfaces (a1,b1,c1). Final fitted surfaces (a2,b2,c2).

namic triangular B-splines and techniques from finite element analysis to reduce the equations to efficient numerical algorithms.

The physics-based model responds to applied simulated forces with natural and predictable dynamics, while the underlying geometric parameters are determined automatically. Designers can employ force-based "tools" to perform direct manipulation and interactive sculpting. Additional control over the shape is available through the modification of physical parameters. Elastic energy functionals allow the qualitative imposition of fairness criteria through quantitative means. Linear or nonlinear constraints may be imposed either as hard constraints that must not be violated, or as soft constraints to be satisfied approximately in the form of simple forces. Constraint-based energy optimization for shape fairing is an automatic consequence of the dynamic model achieving static equilibrium.

Our experimental software demonstrates the ease of use of dynamic triangular B-splines in a variety of applications, including constraint-based optimization, automatic parametric design, shape blending, and interactive sculpting. Since our dynamic model is built upon existing geometric primitives, designers working with it can continue to use existing geometric design toolkits. Thus, dy-

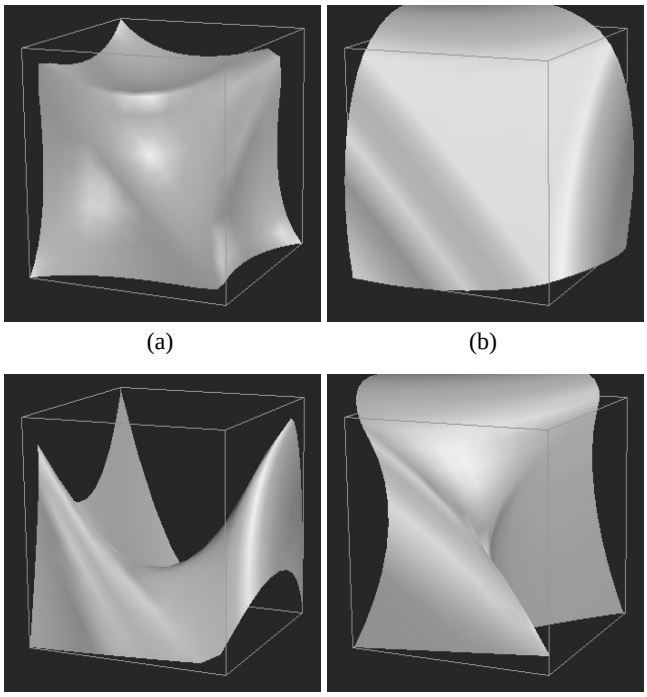


Figure 10: Interactive sculpting of an open quadratic surface into four different shapes (a–d).

namic triangular B-splines appear to be a promising new tool for solid modeling.

Acknowledgments

We are grateful to Professor Hans-Peter Seidel for kindly making available the software for triangular B-spline surfaces that he developed with Philip Fong. This research is supported by grants from the Natural Sciences and Engineering Research Council of Canada and the Information Technology Research Center of Ontario. DT is a fellow of the Canadian Institute for Advanced Research.

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

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

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

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

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

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

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

[13] 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.

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

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

[16] D. Metaxas and D. Terzopoulos. Dynamic deformation of solid primitives with constraints. *Computer Graphics*, 26(2):309–312, 1992.

[17] 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.

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

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

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

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

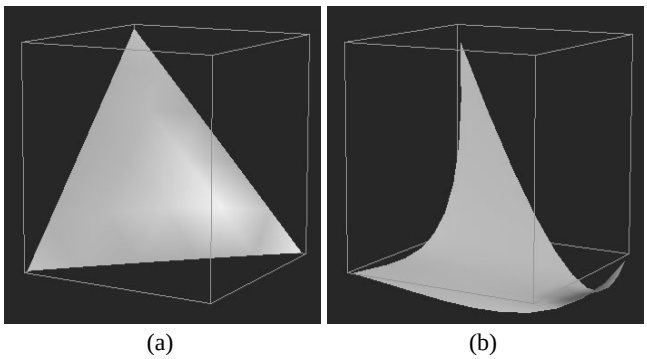


Figure 11: Interactive sculpting of a cubic patch (a) into another shape (b).

[22] H. Qin and D. Terzopoulos. Dynamic NURBS swung surfaces for physics-based shape design. *Computer Aided Design*, 27(2), 1995. in press.

[23] H.-P. Seidel. Representing piecewise polynomials as linear combinations of multivariate B-Splines. In T. Lyche and L.L. Schumaker, editors, *Curves and Surfaces*, pages 559–566. Academic Press, New York, 1992.

[24] H.-P. Seidel. An introduction to polar forms. *IEEE Computer Graphics and Applications*, 13(1):38–46, 1993.

[25] G. Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, MA, 1986.

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

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

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

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

[30] 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.

[31] W. Welch and A. Witkin. Variational surface modeling. *Computer Graphics*, 26(2):157–166, 1992.