Virtual Clay: A Real-time Sculpting System with Haptic Toolkits

Kevin T. McDonnell, Hong Qin and Robert A. Wlodarczyk
Department of Computer Science
State University of New York at Stony Brook
Stony Brook, NY 11794–4400
{ktm|qin|rwlodarc}@cs.sunysb.edu

Abstract

In this paper we systematically develop a novel, interactive sculpting framework founded upon *subdivision solids* and physics-based modeling. In contrast with popular subdivision surfaces, subdivision solids have the unique advantage offering both the boundary representation and the interior material of a solid object. We unify the geometry of subdivision solids with the principle of physics-based models and formulate *dynamic subdivision solids*. Dynamic subdivision solids respond to applied forces in a natural and predictive manner and give the user the illusion of manipulating semi-elastic *virtual clay*. We have developed a real-time sculpting system that provides the user with a wide array of intuitive sculpting toolkits. The flexibility of the subdivision solid approach allows users to easily modify the topology of sculpted objects, while the inherent physical properties are exploited to provide a natural interface for direct, force-based deformation. More importantly, our sculpting system is equipped with natural, haptic-based interaction to provide the user with a realistic sculpting experience.

CR Categories: I.3.5 [Computer Graphics]: Physics-based modeling; I.3.3 [Computer Graphics]: Modeling packages; I.3.6 [Computer Graphics]: Interaction techniques; H.5.2 [User Interfaces]: Haptic I/O; I.3.7 [Computer Graphics]: Virtual reality; I.3.7 [Computer Graphics]: Animation.

1 Introduction and Motivation

To date, most solid modeling techniques have been based on one or more of the following geometric foundations: implicit functions, including constructive solid geometry (CSG) and blobby models; boundary representations (b-reps), such as free-form surfaces, subdivision surfaces, and polygonal meshes; spline-based solids, including Bézier and B-spline solids; and cell decompositions, such as voxel-based models. Although these approaches are ideal for certain applications, each tends to fall short in offering designers the flexible and unified ability to represent and interactively manipulate deformable solid objects of inhomogeneous material distributions. In this work we attempt to overcome some of the difficulties associated with existing modeling approaches by developing a physics-based solid model founded upon *subdivision solids*. Our novel dynamic model serves as the basis for a powerful sculpting system with an intuitive haptic interface as well as a large variety of virtual sculpting tools.

Solid modeling approaches are inherently complex due to the tremendous number of geometric degrees of freedom that are required to represent real-world objects. In principle, such freedom is necessary in order to design complex mechanical parts and to model natural objects such as sculptures. When interacting with spline-based solids, certain geometric quantities, such as the knot vectors in the formulation of Non-Uniform Rational B-splines (NURBS), have little intuitive or physical meaning. Users are typically forced to manipulate a large number of geometric parameters in order to make small changes to solid objects. This problem is especially severe when the geometric data of the modeled object consists of both interior structure and boundary information. The inclusion of interior information necessarily increases the number of requisite degrees of freedom by at least one order of magnitude. In addition, most geometry-based interfaces permit only *indirect* interaction through control point manipulation, which is usually tedious and very time-consuming.

Physics-based modeling techniques can alleviate many of these problems by augmenting geometric objects with physical attributes such as mass, damping and stiffness distributions. Geometric parameters can be hidden from the user by providing natural, force-based interfaces that facilitate *direct* manipulation of solid objects through virtual sculpting tools. The system synchronizes the geometric and physical representations of objects in order to maintain the underlying geometric structures of sculpted solids. Such dynamic and interactive approaches can provide inexperienced users with a natural, force-based interface, while at the same time provide expert users with the familiar geometric interface if desired. Physics-based modeling does not attempt to replace existing geometric-based interfaces but rather to augment them.

In this paper we use physics-based modeling to hide from users the inherent topological and geometric complexity of subdivision solids. Physical attributes are assigned both on the boundary and in the interior of dynamic subdivision solid objects. In addition, we develop a number of haptics-based tools that assist users during 3D interaction with force-feedback and enable them to gain a better understanding of the physical attributes of a given object through haptic exploration. Haptics can enhance the functionality of deformable models in a natural and intuitive way. There is a natural connection between the two because both haptics and dynamic models depend on real-world physical laws to drive the realistic simulation and interaction of dynamic objects. The significant impact of our haptic interfaces extends from virtual sculpting to education, surgical simulation, virtual reality, and entertainment. Such interfaces can fundamentally improve understanding by permitting direct interaction with virtual objects and by enhancing the sense of realism through computer communication.

The remainder of this paper is organized as follows. In Section 2 we briefly outline the contributions of this research. We present some relevant background material in Section 3 and follow in Section 4 with a detailed discussion of our novel dynamic solid model. We give an overview of the structure of our sculpting system in Section 5 and then detail its key functionality in Sections 6 and 7. We then document the performance data of our sculpting system in Section 8 and conclude the paper with example sculptures.

2 Research Contribution

Our novel dynamic modeling approach is based on a relatively new type of subdivision model, recently pioneered by MacCracken and Joy [11]. However, the original work in [11] was developed only as

a new mechanism for free-form deformation (FFD). We employ the subdivision rules in [11] to systematically formulate free-form subdivision solids for physics-based volumetric sculpting. Moreover, we have demonstrated that the new subdivision solids naturally include free-form volumetric splines such as B-spline solids as their special cases.

In comparison with mature modeling techniques associated with subdivision surfaces, subdivision solid formulations transcend surface-based approaches by defining geometry and topology both in the interior and on the boundary of solid objects. We augment subdivision solids with physical properties that permit users to directly manipulate solids through a haptic interface and physical forces. The inherent geometric parameters of the solid maintain the subdivision structure, while the physical attributes support direct manipulation. The geometric and topological complexities of subdivision solids are concealed by both our physics-based formulation and by a carefully designed user interface. Our sculpting system provides users with many virtual tools that let them directly manipulate solid objects. The user can quickly and easily make topological, geometric and physical changes to sculpted objects without the need to understand the complicated mathematics of subdivision solids.

In this paper we have significantly improved upon the results reported by McDonnell and Qin in [13]. While their sculpting application provides only an explicit numerical solver, our system offers both an explicit and implicit solver, resulting in more robust and stable simulations. In addition, our physical model incorporates diverse types of elastic behavior that can produce more structurally stable objects with more shape variation than those in [13]. Furthermore, we have devised and implemented a local, adaptive subdivision algorithm that can be employed to create smaller and finer features in any region(s) of interest. More importantly, our novel sculpting system provides the user with a suite of more powerful sculpting tools with various types of natural haptic-based interaction and supports real-time manipulation of realistic, dynamic, virtual solid objects.

3 Related Work

3.1 Subdivision Solids

Recently, subdivision solids [11] emerged as a generalization of tricubic B-spline solids to free-form solid models of arbitrary topologies. In essence, B-spline solids and other spline-based solid representations can be cumbersome to use in sculpting applications since many separate solids must be carefully patched together to create simple features like holes. Subdivision solids, however, can represent a flexible object containing many holes with a single *control lattice*. In contrast to the *control mesh* that defines a typical subdivision surface, which consists of points, edges and polygons, the control lattice of a subdivision solid consists of points, edges, polygons and closed polyhedra (which we call "cells"). Note that the use of cells results in the complete subdivision of the 3D space occupied by the control lattice, and hence allows users to represent a truly "solid" object without any geometric ambiguity and topological inconsistency. The geometric construction of our dynamic solid model is based on [11], described in Appendix A. (Note that since the original subdivision rules do not define the surface structure, we use the Catmull-Clark subdivision surfaces rules [4] to obtain the smooth boundary.)

Unlike any subdivision surface, our subdivision solid unambiguously defines the 3D space occupied by its control lattice. Subdivision surfaces, in contrast, can characterize only the boundary of a 3D object. This critical distinction is exploited frequently in our formulations and algorithms. The more sophisticated topological structure allows the user to modify material properties in the interior of solids with ease and can thus significantly enhance the functionality of virtual sculpting. Such a solid representation also lends itself very well to quickly removing and adding portions of virtual material by modifying the polyhedron-based control structure.

Hence, subdivision solids can provide a very general framework for representing objects. While subdivision surfaces are very convenient and general for creating models in which only the boundary is important, subdivision solids can offer users additional flexibility whenever necessary or appropriate. Like subdivision surfaces, subdivision solids are natural candidates for multiresolution analysis and level-of-detail (LOD) control since one can subdivide a given model until the desired amount of detail is achieved. Subdivision solids depend solely on the use of simple subdivision algorithms that are similar to those found in most subdivision surface approaches. Hence, the solid subdivision algorithm is relatively easy to implement and does not require very sophisticated data structures.

3.2 Physics-based Modeling

Our dynamic subdivision solid model is based on well-established techniques and algorithms from physics-based modeling. Dynamic solid models were introduced to the modeling and computer graphics communities by Terzopoulos and colleagues [24, 23, 22, 14]. In a nutshell, the geometry of their models is discretized from continuous surfaces and solids and is attached to mass-spring meshes. Others have used modal dynamics [17] and the finite element method [5] in order to improve the stability and accuracy of dynamic models. Baraff, Witkin, Kass and others [27, 2, 3] have developed techniques for animating and constraining non-penetrating dynamic surfaces. Qin and colleagues [20, 19, 12] derived dynamic models for direct manipulation of spline- and subdivision-based surfaces. Most recently, James and Pai [9] developed a dynamic surface model based on the Boundary Element Method (BEM).

3.3 Haptic Interfaces

Our sculpting system offers a number of haptics-based sculpting tools that attempt to enhance the sense of realism experienced by the user. A good review of haptics literature can be found in [21]. Thompson *et al.* [25] derived efficient intersection techniques that permit direct haptic rendering of NURBS surfaces. Miller and Zeleznik [15] defined a set of robust haptic principles and techniques that can be applied to nearly any type of haptic interface. Dachille *et al.* [6] developed a haptic interface that permits direct manipulation of dynamic surfaces. Balakrishnan *et al.* [1] developed *ShapeTape*, a curve and surface manipulation technique that can sense user-steering bending and twisting motions of the rubber tape. In sum, haptics is an active area of research that has demonstrated great potential for the development of natural and intuitive human-computer interfaces.

4 Dynamic Subdivision Solids

Our dynamic subdivision solid model marries the geometric information and topological structure of subdivision solids with physical attributes and other relevant material quantities. A control lattice may be initialized by the user at any time to define the overall shape of certain object. Figure 1 shows a cubical control lattice after one level of subdivision. The complex cell structure in the interior of the solid is highlighted in Figure 1(b). The difference in complexity between the subdivision solid wireframe (Figure 1(c)) and subdivision surface wireframe (Figure 1(d)) can also be easily observed. The second wireframe is that of a Catmull-Clark surface which matches exactly the boundary of the solid. The first

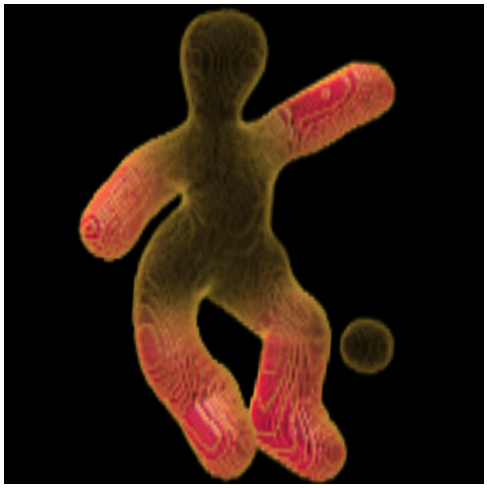


Figure 2: A volume rendering of the stiffness distribution of a soccer player model. High stiffness in the extremities is indicated by red, while low-stiffness regions are colored yellow. See also the color plate.

wireframe clearly demonstrates the topological and geometric complexity introduced by the cells of the solid. In order to manage the complex non-manifold topological nature of subdivision solids, we used the radial-edge data structure invented by Weiler [26]. This storage scheme supports efficient queries about the non-manifold topology by explicitly storing adjacency and orientation information. This approach, in spirit, is very similar to the winged-edge and quad-edge data structures for polygonal surfaces. Note that, in the interest of clarity and convenience, we render solid sculptures as surfaces for most of the examples in this paper. Nevertheless, a large amount of topological, geometric, physical and material information is, in fact, employed for the interior description and is therefore hidden. As an example, the volume rendering in Figure 2 shows the stiffness distribution for a particular soccer player model. Red regions indicate high stiffness, while regions colored in darker yellow indicate low stiffness (See also the color plate).

After a user-specified number of subdivisions of the control lattice, the resulting subdivision solid is endowed with physical properties such as mass, damping and stiffness distributions. To avoid ambiguity, we shall use the term "subdivided lattice" to describe the control lattice after one or more applications of the subdivision algorithm. Material properties are assigned both on the inside and on the outside of the subdivided solid. Note that a subdivision surface could not represent heterogeneous interior attributes because the interior of a subdivision surface is empty. The control lattice is retained but is not assigned any physical parameters because material properties are associated only with the limit shape of subdivision solids and not with the initial control lattice. The control structure is required to maintain the geometric validity of the subdivided solid. More details on this issue are provided later.

Using our approach, each point in the subdivided solid is assigned an initial mass by the application. To avoid confusion with points in the control lattice, these points will henceforth be called "mass points." Vertices in the control lattice shall be called "control points." Each edge between mass points is assigned an initial spring stiffness. Such edges are termed "normal springs," while edges in the control lattice shall be called "control edges." Each face in the subdivided lattice is also assigned a set of "angular springs," which we describe in detail later. These faces in the subdivided solid are called "subdivided faces," while faces in the control lattice will be termed "control faces." Similarly, cells in the subdivided lattice are

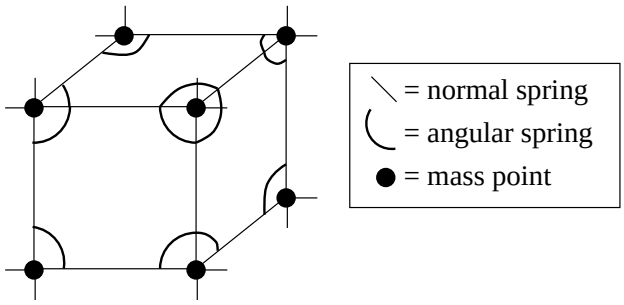


Figure 3: A typical cell in the subdivided lattice. Normal springs (straight lines) resist stretching forces, while angular springs (arcs) resist shearing forces. Each subdivided edge is assigned a normal spring, and each vertex in each face is assigned an angular spring.

"subdivided cells," and cells in the control lattice are "control cells." In summary, the initial control lattice is subdivided and is then used to define a mass-spring lattice containing mass points and two types of springs.

Like many subdivision surface algorithms, the subdivision solid representation can be expressed as a global matrix multiplication:

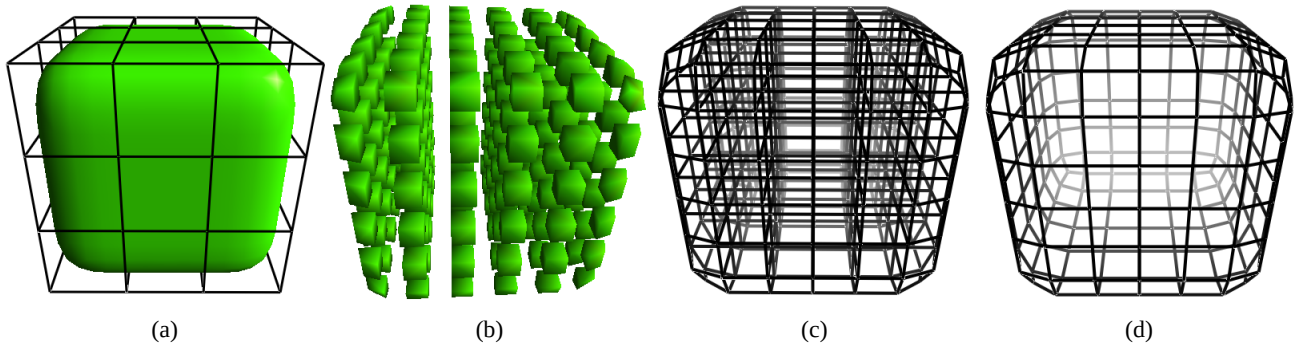
$$\mathbf{d} = \mathbf{A}\mathbf{p}. \tag{1}$$

$\mathbf{p}$ is a column vector of the positions of the control points; the matrix $\mathbf{A}$ is a sparse matrix that contains weights given by the subdivision rules; and the column vector $\mathbf{d}$ gives the positions of the mass points after multiplication of the control point positions by the subdivision matrix. Note that matrix $\mathbf{A}$ is a global subdivision matrix and that local changes to the subdivision rules can be expressed by changing a few rows of the matrix. Due to the sparsity of $\mathbf{A}$, we employ a storage scheme for sparse matrices to both reduce memory overhead and to improve time performance.

Figure 3 shows a typical subdivided cell with its mass points, normal springs, and angular springs. Normal springs are traditional linear springs found in many other dynamic models and in introductory physics textbooks. Their end-points are mass points in the subdivided lattice. The force exerted on their end-points is given by Hooke's law: $f = -kx$, where k is the spring's stiffness and x is the difference between the spring's current and rest lengths. In our dynamic solid model, each normal spring can have a different stiffness and rest length, both of which can be changed by the user through the interface (see below for the details).

Traditional mass-spring lattices suffer from structural instability since they do not attempt to maintain internal angles (*i.e.*, to avoid shearing). Some other dynamic models rely on diagonal springs to avoid these problems, but these types of springs can exert forces even when shearing is not occurring (see Figure 4). In order to better characterize an object's resistance to shearing forces, we employ angular springs instead. The system assigns one angular spring for each mass point in each subdivided face. An angular spring exerts forces on points only when its associated angle deviates from its initial value. The angular spring pushes end-points away from (or towards) each other in order to regain the original angle. In this way the model attempts to enforce a soft curvature constraint on faces.

Figure 5 shows the arrangement of angular springs in a subdivided face. For each mass point in each subdivided face, we compute the initial rest angle of the two normal springs that coincide at that mass point. When shearing occurs in the face, the angular springs exert forces on the two end-points to bring the angle back to its initial value (see Figure 6). The initial angle (θ^0) and current angle (θ) can be computed directly using the dot product of the two incident edges ($\mathbf{s}_1$ and $\mathbf{s}_2$): $\theta = \arccos \frac{\mathbf{s}_1 \cdot \mathbf{s}_2}{|\mathbf{s}_1||\mathbf{s}_2|}$. To com-



(a) (b) (c) (d)

Figure 1: (a) A subdivision solid after one level of subdivision along with its control lattice. (b) After scaling the cells, the complex interior structure becomes visible. (c) A wireframe rendering of the subdivided solid. (d) A wireframe of a Catmull-Clark subdivision surface that defines the same boundary. The difference in geometric and topological complexity of the solid and surface is significant.

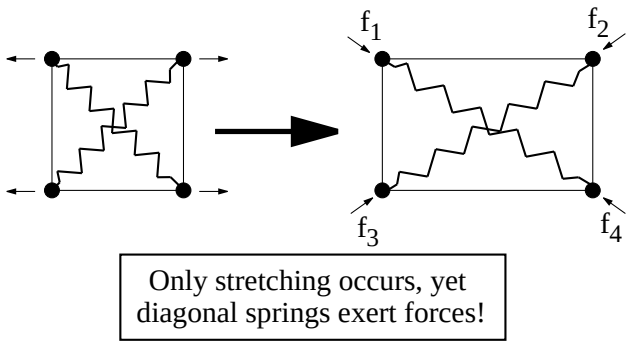


Figure 4: Diagonal (anti-shearing) springs exhibit incorrect behavior when stretching occurs but not shearing, as this figure demonstrates. In this special case, diagonal springs erroneously exert forces to resist stretching. Angular springs (Figures 5 and 6) should be used instead in order to correctly characterize an object's resistance to shearing forces.

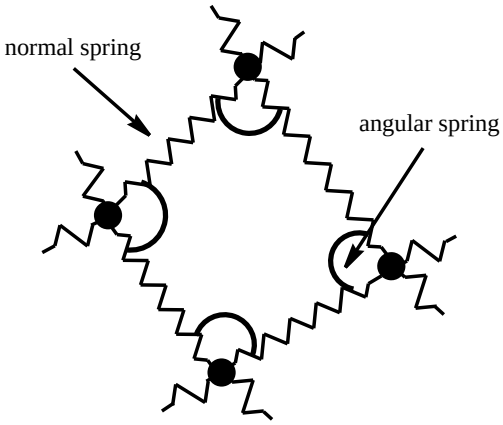


Figure 5: To help maintain the structural stability of a dynamic subdivision solid, special angular springs are used to help maintain internal angles and to counteract shearing forces. Traditional springs, indicated by jagged lines, are employed to resist stretching forces.

pute the forces that will be applied to each of the end-points of $\mathbf{s}_1$ and $\mathbf{s}_2$ (indicated by $\mathbf{e}_1$ and $\mathbf{e}_2$ in Figure 6), we calculate where the end-points would be if the current angle equaled the rest angle (*i.e.*, $\theta = \theta^0$). This requires computing two rotations around the mass point. The axis of rotation is easily computed using the cross product of $\mathbf{s}_1$ and $\mathbf{s}_2$: $\mathbf{s}_1 \times \mathbf{s}_2$. The rotations provide two new *virtual* points, $\mathbf{e'}_1$ and $\mathbf{e'}_2$, from which we compute displacements $\vec{\mathbf{d}}_1 = \mathbf{e'}_1 - \mathbf{e}_1$ and $\vec{\mathbf{d}}_2 = \mathbf{e'}_2 - \mathbf{e}_2$. Using Hooke's law, linear spring forces are applied to $\mathbf{e}_1$ and $\mathbf{e}_2$ in the directions of $\mathbf{d}_1$ and $\mathbf{d}_2$ to help bring the angle back to its rest value.

It may be noted that the formulation of our dynamic subdivision solids and its aforementioned stiffness distribution (especially the structure of its angular springs) are founded upon concepts and their rigorous analysis from differential geometry [22]. Consider a 3×3 metric tensor function $\mathbf{G}(\mathbf{s})$:

$$\mathbf{G} = \begin{bmatrix} \frac{\partial \mathbf{s}}{\partial u} \cdot \frac{\partial \mathbf{s}}{\partial u} & \frac{\partial \mathbf{s}}{\partial u} \cdot \frac{\partial \mathbf{s}}{\partial v} & \frac{\partial \mathbf{s}}{\partial u} \cdot \frac{\partial \mathbf{s}}{\partial w} \\ \frac{\partial \mathbf{s}}{\partial v} \cdot \frac{\partial \mathbf{s}}{\partial u} & \frac{\partial \mathbf{s}}{\partial v} \cdot \frac{\partial \mathbf{s}}{\partial v} & \frac{\partial \mathbf{s}}{\partial v} \cdot \frac{\partial \mathbf{s}}{\partial w} \\ \frac{\partial \mathbf{s}}{\partial w} \cdot \frac{\partial \mathbf{s}}{\partial u} & \frac{\partial \mathbf{s}}{\partial w} \cdot \frac{\partial \mathbf{s}}{\partial v} & \frac{\partial \mathbf{s}}{\partial w} \cdot \frac{\partial \mathbf{s}}{\partial w} \end{bmatrix} \quad (2)$$

where (u, v, w) is a local parameterization for $\mathbf{s}(\mathbf{x})$. One important result is that the above matrix tensor remains unchanged if there is no solid deformation for any solid object $\mathbf{s}(\mathbf{x})$ (*i.e.*, rigid-body motion does not modify the tensor function $\mathbf{G}(\mathbf{s})$). The amount of deformation of a given object is therefore defined by a function of $G^0_{ij} - G^n_{ij}$, which is the difference between the metric tensors of the initial and current states of the object, where the superscript denotes time, and the subscripts stand for matrix entries. Therefore, in the context of dynamic subdivision solids, normal springs attempt to minimize the change for the diagonal terms of $\mathbf{G}(\mathbf{s})$, while the angular springs attempt to minimize the variation of the off-diagonal terms.

5 Overview of Sculpting System

The user interface (see Figure 7) of our sculpting system consists of a Sensable Technologies[1] PHANToM 1.0 3D haptic input/output device, a standard 2D mouse, and on-screen GUI controls. The PHANToM features a thimble for the user's index finger and can exert a real-world force in any 3D direction. The mouse is used to activate or enable over a dozen sculpting tools as well as control various simulation parameters through GUI sliders and check boxes. Forces are exerted on the user's finger only when a haptics-based sculpting tool is active. For the geometric/topological sculpting tools, the PHANToM is used only to obtain a 3D cursor position

[1] www.sensable.com

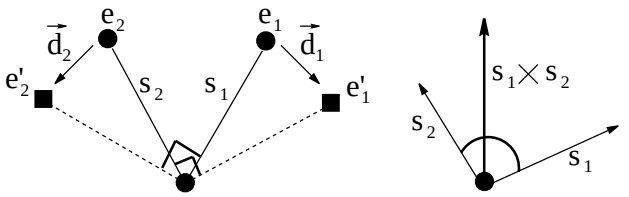


Figure 6: To compute the forces exerted by an angular spring, we begin by computing where end-points $\mathbf{e}_1$ and $\mathbf{e}_2$ would be if there were no shearing (given by $\mathbf{e'}_1$ and $\mathbf{e'}_2$). The points are rotated about the mass point using the axis of rotation (given by the cross product $\mathbf{s}_1 \times \mathbf{s}_2$). Using these virtual positions we compute a pair of displacements ($\vec{\mathbf{d}}_1$ and $\vec{\mathbf{d}}_2$). The displacements are used to provide the spring forces we apply to the end-points to help bring the angle back to its rest configuration. In this example we assume the rest angle is 90 degrees.

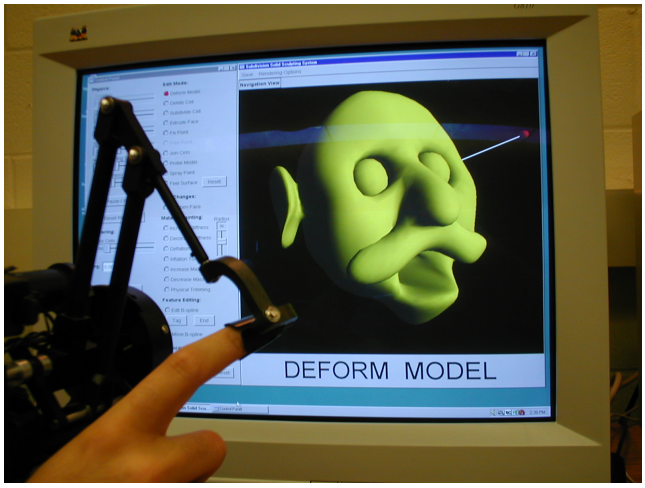


Figure 7: The user interface consists of a PHANToM haptic device, a standard 2D mouse, and on-screen GUI controls.

– no forces are felt. The sculpted object is rendered using OpenGL on an inexpensive Nvidia TNT 2 Ultra chipset-based graphics accelerator. The GUI was built using the Fast Light Tool Kit[2] (FLTK), which offers a full suite of widgets and very good OpenGL support. The entire system runs on a generic Microsoft Windows NT PC with an Intel Pentium III 550 Mhz CPU and 512 MB RAM. Complete descriptions of all of the sculpting tools are provided in Section 7, and system performance statistics are given in Section 8.

An overview of the physical simulation and design process is shown in Figure 8. After an initialization step in which certain data structures are assembled, the system runs in a loop and continuously updates the physical state of a sculpted object. The system traverses all springs and computes the total internal forces acting on the mass points. External forces are queried from the input devices and are added to the system. The aggregate forces on the mass points are then mapped to the control vertices (see Section 6). The acceleration and velocity of the control lattice are then computed in order to move the control lattice to its new position. The subdivision matrix is applied to the control vertices to obtain the new positions of the mass points. This step is required to maintain the geometric and subdivision structure of the subdivided lattice. At any time during the simulation, the user may pause the execution and invoke a tool that causes a change in topology (for instance, to create a protrusion). This type of action requires re-subdivision of the geometry and the reconstruction of certain data structures.

A second thread is required to control the haptic interaction and to accept 3D input from the user. If a haptic-based tool is in use, this thread computes the haptic force to exert on the user's hand. Without a second thread of execution the haptic device would not be able to send real-time forces to the user's finger, which would result in unrealistic buzzing or jerky motions.

As far as data structures are concerned, most geometric quantities and matrices are stored in one- and two-dimensional arrays. To represent the sophisticated topological structure of subdivision solids, we use a simplified version of Weiler's radial-edge data structure [26, 16]. By storing adjacency information, this data structure allows our system to efficiently query and update topological information. Fast accesses are critical in order to subdivide a given solid in a reasonable amount of time and to make topological changes as efficiently as possible. In addition, the radial-edge data structure facilitates both local and global subdivisions.

6 Numerical Solvers

We have implemented both an explicit numerical solver for time integration and an implicit one. When extensive user interaction is required, the explicit solver is used to provide real-time update rates. The implicit solver can be invoked when numerical stability is more of a concern or when one is performing offline simulations.

6.1 Explicit Time Integration

Our dynamic model is founded upon the energy-based Lagrangian equation of motion, which has the following continuous and discrete forms, respectively:

$$\mu\ddot{\mathbf{s}} + \gamma\dot{\mathbf{s}} + \frac{\partial E(\mathbf{s})}{\partial \mathbf{s}} = \mathbf{f} \tag{3}$$

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{D}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} = \mathbf{f_x} \tag{4}$$

The $\mathbf{M}$, $\mathbf{D}$ and $\mathbf{K}$ matrices represent the mass, damping and internal energy distributions of an object; $\mathbf{x}, \dot{\mathbf{x}}$ and $\ddot{\mathbf{x}}$ represent the discrete position, velocity and acceleration of an object; and $\mathbf{f_x}$ contains the total external forces acting an object.

We augment the discrete Lagrangian equation of motion with geometric and topological quantities related to the subdivision solid algorithm. $\mathbf{p}$ is a vector containing the positions of the control points, and $\mathbf{d}$ stores the positions of the mass points. $\mathbf{f_d}$ stores the external forces acting on each mass point. The matrix $\mathbf{A}$ stores the subdivision weights that define the positions of the mass points based on the positions of the control points. Subject to the constraints defined by Equation 1 we augment the Lagrangian equation of motion as follows. First we substitute $\mathbf{d}$ for the generic $\mathbf{x}$ and multiply each side by $\mathbf{A}^\top$:

$$\mathbf{A}^\top\mathbf{M}\ddot{\mathbf{d}} + \mathbf{A}^\top\mathbf{D}\dot{\mathbf{d}} + \mathbf{A}^\top\mathbf{K}\mathbf{d} = \mathbf{A}^\top\mathbf{f_d}$$

By applying Equation 1 and rearranging terms we obtain:

$$\mathbf{A}^\top\mathbf{M}\mathbf{A}\ddot{\mathbf{p}} + \mathbf{A}^\top\mathbf{D}\mathbf{A}\dot{\mathbf{p}} + \mathbf{A}^\top\mathbf{K}\mathbf{A}\mathbf{p} = \mathbf{A}^\top\mathbf{f_d}$$

$$\mathbf{A}^\top\mathbf{M}\mathbf{A}\ddot{\mathbf{p}} = \mathbf{A}^\top\mathbf{f_d} - \mathbf{A}^\top\mathbf{D}\dot{\mathbf{d}} - \mathbf{A}^\top\mathbf{K}\mathbf{d}$$

Then we solve for the acceleration of the control points using a least-squares approach:

$$\ddot{\mathbf{p}} = (\mathbf{A}^\top\mathbf{M}\mathbf{A})^{-1}(\mathbf{A}^\top\mathbf{f_d} - \mathbf{A}^\top\mathbf{D}\dot{\mathbf{d}} - \mathbf{A}^\top\mathbf{K}\mathbf{d}). \tag{5}$$

[2] www.fltk.org

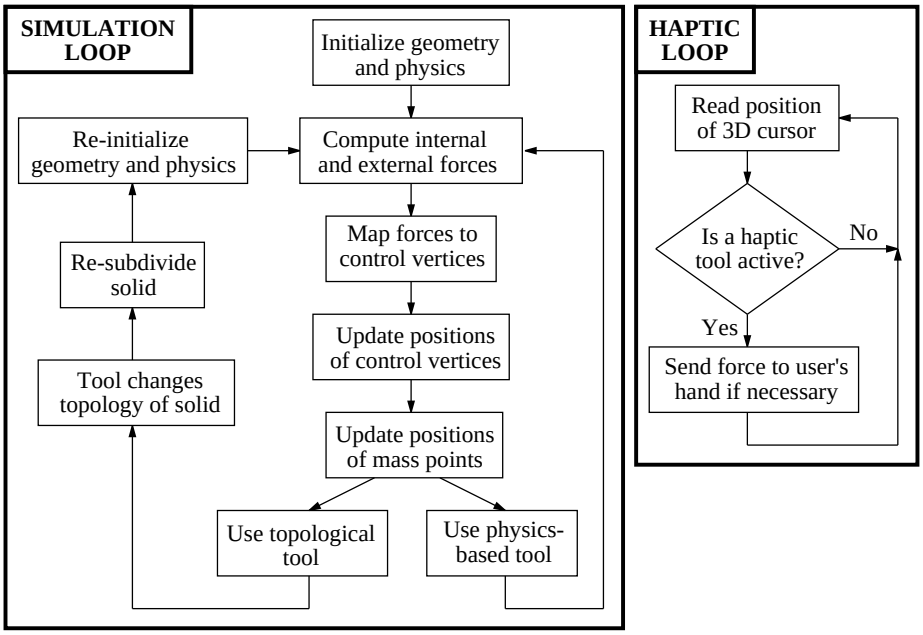


Figure 8: An overview of the physical simulation and design process featured in our sculpting system. Since our sculpting interface features haptics, the application requires two separate execution threads in order to generate real-time haptic feedback.

Equation 5 states that the external forces acting on the mass points are mapped to the control points (which are purely geometric objects) using the subdivision matrix. Once the accelerations of the control points are computed, the model's position and velocity are computed using a forward Euler method to drive the simulation:

$$\dot{\mathbf{p}}_{i+1} = \dot{\mathbf{p}}_i + \ddot{\mathbf{p}}_i \Delta t \qquad \mathbf{p}_{i+1} = \mathbf{p}_i + \dot{\mathbf{p}}_i \Delta t$$

6.2 Implicit Time Integration

While the explicit numerical solver described above is simple to implement and runs very quickly, it suffers from numerical instability when large time-steps are taken or when spring stiffnesses are very high. To tackle this problem we derived and implemented an implicit solver based on backward Euler integration. Discrete derivatives are computed using backward differences:

$$\ddot{\mathbf{p}}_{i+1} = \frac{(\mathbf{p}_{i+1} - 2\mathbf{p}_i + \mathbf{p}_{i-1})}{\Delta t^2}$$

and

$$\dot{\mathbf{p}}_{i+1} = \frac{(\mathbf{p}_{i+1} - \mathbf{p}_{i-1})}{2\Delta t}.$$

We obtain the time integration formula

$$\left(2\mathbf{M_p} + \Delta t \mathbf{D_p} + 2\Delta t^2 \mathbf{K_p}\right) \mathbf{p}_{i+1} = 2\Delta t^2 \mathbf{f_p} + 4\mathbf{M_p}\mathbf{p}_i - (2\mathbf{M_p} - \Delta t \mathbf{D_p})\mathbf{p}_{i-1}, \quad (6)$$

where

$$\mathbf{M_p} = \mathbf{A}^\top \mathbf{M} \mathbf{A},$$

$$\mathbf{D_p} = \mathbf{A}^\top \mathbf{D} \mathbf{A},$$

$$\mathbf{K_p} = \mathbf{A}^\top \mathbf{K} \mathbf{A}$$

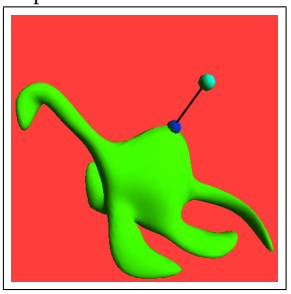
and the subscripts denote evaluation of the quantities at the indicated time-steps. It is straightforward to employ the conjugate gradient method [18] to obtain an iterative solution for $\mathbf{p}_{i+1}$. To achieve interactive simulation rates, we limit the number of conjugate gradient iterations per time-step to 10. We have observed that two iterations typically suffice to converge the system to a residual error of less than 10^{-4}. More than two iterations tend to be necessary when the physical parameters are changed dramatically during interactive sculpting.

7 User Interface

Our sculpting system features three classes of tools: haptic tools, which exert real forces on the user's hand; geometric/topological tools, which cause geometric and/or topological changes in the subdivision structure; and physics-based tools, which evoke local or global changes in the physical properties of an object. The material behaves like semi-elastic virtual clay, although this behavior can be made more plastic through the GUI.

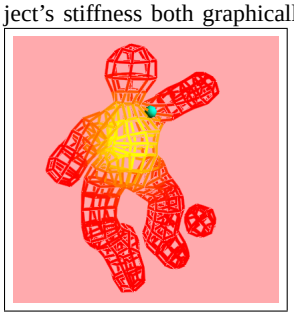
7.1 Haptic Tools

Haptic-based Deformation: Through our non-compressive "rope" tool the user can select any of the mass points, both on the boundary and in the interior, and exert a linear spring force on the point. This tool is employed to deform a shape much in the same way that an artist might mold a piece of clay with his fingers. A button in the GUI activates the tool and creates a temporary spring between the cursor and the mass point. The PHANToM exerts a real force on the user's hand that is linearly proportional to the mass of the attached point, the distance of the cursor to the point, and the strength of the rope tool (whose stiffness can be changed in the GUI). The tactile sensation is akin to pulling on a taut rubber band. The intensity of the force is



visually manifested by shading the background a red color – high color saturation corresponds to strong haptic forces.

Haptic-based Probing: The user has the ability to study an object's stiffness both graphically and haptically. When active, the

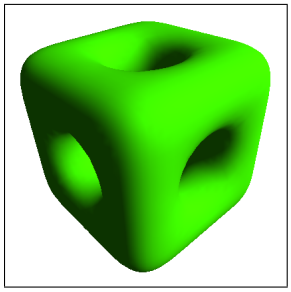


probing tool exerts a force on the user's hand proportional to the local stiffnesses of those springs within a given radius of the tool. A spring's influence on the aggregate force decreases linearly with increasing distance from the tool. A wireframe rendering indicates the stiffness distribution (red ↔ orange ↔ yellow = high ↔ medium ↔ low stiffness) to aid in the probing process. The user feels a viscous-like force that smoothly increases and decreases as the 3D cursor moves into and out of regions of high and low stiffness. Like the rope tool, the probing tool modulates the background color depending on the strength of the force (high force = deep red).

Haptic Surface Rendering: The user can also run the cursor over the surface of a solid to examine the boundary tactilely. In this way a designer can feel how smooth the surface is at the current subdivision level and also look for any surface anomalies that might be hard to detect visually.

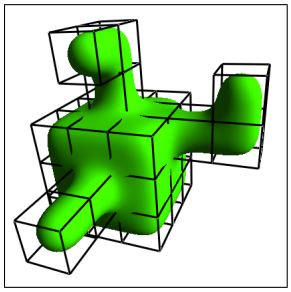
7.2 Geometric and Topological Tools

Cell Deletion: Using the 3D cursor, the user can select any cell in the control lattice and delete it. This action lets the



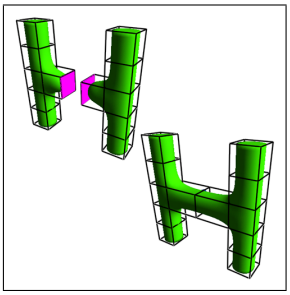
user remove material from a particular sculpture. A simple linear search is performed in order to determine which cell is currently closest to the cursor. As with all of the tools that modify the topology of the sculpted object, the radial-edge data structure and certain matrices need to be re-computed to effect the change. The cell-deletion tool, as well as all of the other geometric- and topological-based sculpting tools, does not employ haptic feedback.

Extrusion: When the extrusion tool is activated, new material is extruded from the face on the surface nearest the 3D cursor.



Since an extrusion from an interior face would not have a proper physical meaning, it is prohibited in our current implementation. A linear search is conducted to determine the nearest surface face. When the user activates the tool, a new control cell is created and attached to the face. It "grows" in the direction of the surface normal, and its size is computed based on the size of the control cell to which it is attached.

Joining: Two cells near the surface of a sculpture can be joined together by highlighting one face in each cell and one point in each of those faces. The point highlighting is needed in order to make a proper point-to-point correspondence for the radial-edge data structure. The 3D cursor is used to select the two faces and two points. The cell-joining tool is very useful for making loops, twisted cylindrical structures, ring-like features, and handles. In reality this kind of action is somewhat difficult to achieve even with actual clay.



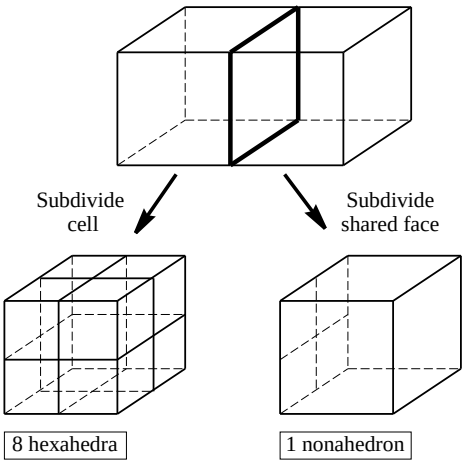
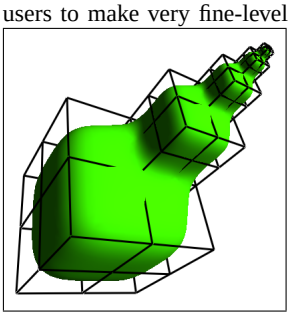


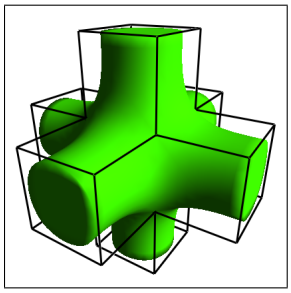
Figure 9: The cell on the left is subdivided locally. The face indicated by heavy lines is shared between the two cells. This face must be subdivided to maintain topological consistency.

Local Subdivision: Our local subdivision algorithm allows users to make very fine-level details on their models. Much like



the cell-deletion tool, the user selects a cell to subdivide with the 3D cursor. The system replaces the highlighted cell with several smaller cells whose structure is based on the normal subdivision solid rules. Faces and edges in adjacent cells are also subdivided in order to avoid T-shaped and other topological anomalies (such as a vertex in the middle of a face). This process is illustrated in Figure 9. Although this procedure complicates the local topological structure, the resulting control lattice is still valid and can be treated and manipulated like any other subdivision control lattice. That is, this particular local subdivision algorithm does not generate any special cases for the sculpting tools or for the global subdivision algorithm. The avoidance of special cases simplifies the sculpting tools and subdivision algorithm considerably.

Sharp Features: Sharp features such as corners and creases can be created by tagging faces, edges, or points as "sharp". To achieve

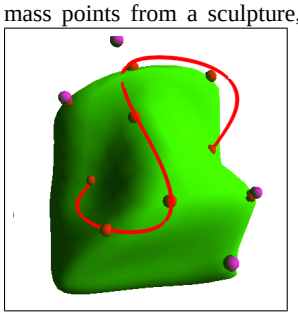


the same effect in real life, sculptors who work with real clay need to either cut straight slices of material from a model or to carefully smooth an area until the crease or corner appears. In our implementation, sharp features are created by using different subdivision rules for tagged geometric elements. For instance, when a face is tagged as sharp, different face-point, edge-point, and vertex-point subdivision rules are applied to the control lattice. Specifically, the face-point is the face's centroid, while the vertex-end edge-point positions are given by the subdivision rules for uniform cubic B-spline curve construction [10]. Note that since we use a Catmull-Clark surface to represent the boundary, one can use other special rules, such as those found in [7, 11, 8]. Although interior faces, edges and points can be tagged as sharp, we have found that doing so usually does not noticeably affect the boundary.

7.3 Physics-based Tools

In addition to the deformation/rope tool, we have several other sculpting tools that use physics to interact with a model. The use of physical tools frees the user from the need to deal with the multitude of control vertices inherent in subdivision solid objects. In addition, the user can interact with dynamic subdivision solids directly, rather than through indirect and often non-intuitive control point manipulation.

B-spline Curve Manipulation: This tool lets the user select mass points from a sculpture, build a cubic B-spline curve that



interpolates the points, and then manipulate the spline using its control points. The selected mass points are used to sample the parametric domain at uniform intervals. These parametric values and the positions of mass points give the initial positions of the B-spline control points. Temporary high-stiffness springs are used to attach the spline to the mass points. When the spline's control points are moved by the user, the attached springs exert forces on the model that cause the object's mass points to move towards the spline. In this way the user can use a B-spline to quickly design curved regions as well as make large, well-controlled deformations. It is straightforward to generalize this tool to B-spline surfaces and other types of curve and surface representations for more sophisticated deformation patterns and more meaningful sculpting. This generality creates the possibility of the development of advanced feature-based design toolkits.

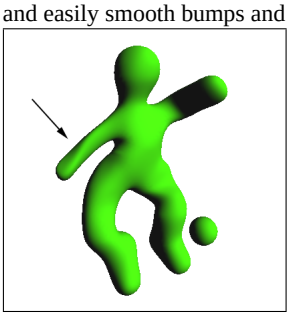
Stiffness Painting: Since our model defines geometric and physical properties on both the boundary and interior, the user has the

option of modifying the stiffness distribution both on the inside and outside. Unlike real clay, our virtual material can actually have different physical properties throughout the sculpture, offering extra flexibility and advantages. This functionality is useful when one wishes to localize the effects of certain tools, especially the rope tool. For instance, to sculpt the plesiosaur neck in Figure 10, we first made the head very stiff and then dragged it upwards. The neck stretched readily while the head maintained its desired shape.

In our sculpting system we use a painting metaphor to describe the process of changing material properties. Using the 3D cursor, springs within a user-supplied radius of the tool are slowly "painted" with increasing or decreasing stiffness, based on which tool is active. The rate at which a spring's stiffness changes decreases linearly with increasing distance from the tool. This functionality allows the user to create smooth transitions from regions of low stiffness to regions of high stiffness. The stiffness distribution can be viewed as a color-mapped wireframe or can be physically felt by using the haptic probing tool (see above).

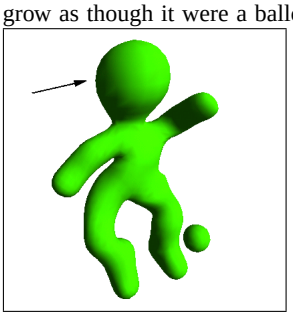
Mass Painting: Similarly, the user can locally modify the masses of mass points through painting. This tool can allow a user to control how quickly a part of a sculpture will respond to deformation. Regions of high mass density tend to move slowly while less massive parts respond quickly to user-applied forces.

Deflation Tool: The deflation tool allows the user to quickly and easily smooth bumps and taper protrusions. While this process



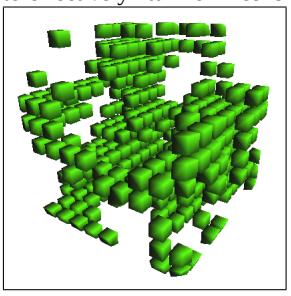
can be extremely time-consuming and tedious when working with real clay, our system can achieve this goal very effectively. A painting metaphor also describes this process in a very natural fashion. As the tool passes near springs, their rest lengths are slowly decreased. This has the effect of shrinking the solid towards the tool's center. The rate of change in stiffness decreases linearly with increasing distance from the tool. This prevents sudden changes in the solid's shape and subsequently reduces the likelihood of accidentally creating unwanted pits and depressions.

Inflation Tool: The inflation tool causes a localized region to grow as though it were a balloon being inflated. This tool allows



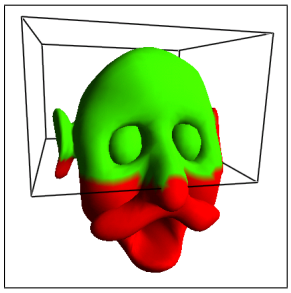
the user to make rounded features in a very short period of time. The plesiosaur's main body was created in this way, for instance. The inflation tool increases rest lengths as the cursor passes near springs. The simulation recognizes that the current lengths of the springs are too short and in response exerts forces on the mass points to increase the current spring lengths. This kind of action is very difficult to achieve when working with real clay and requires many incremental changes to obtain the same effect.

Physical Trimming: The physical trimming tool allows users to effectively "turn off" cells in the subdivided lattice. Unlike the



cell-deletion tool, which modifies the topology of a sculpture, this tool directly operates on the physical representation. It can be used to "delete" any of the subdivided cells, although the sculpture must be of high enough resolution to avoid aliasing. Such rapid changes would be difficult to achieve with most surface models. In order to prevent disabled cells from affecting the physical simulation, spring stiffnesses are set to zero in such cells.

Physical Window: Sometimes the user may wish to deform only a local region of the model. This action can be facilitated by us-



ing our "physical window" feature that constrains the physical simulation to occur in a local area of interest. Such functionality is useful when one wishes to ensure that previous changes to the model will not be affected by new deformations. With the 3D cursor, the user marks a 3D rectangular region in which he wishes the deformation to be active. Control points and mass points outside the marked region are not processed by the system and remain fixed in space while the physical window is active. Their color is changed to red to remind the user that they are un-movable.

In addition to localizing the effect of deformation, the physical window speeds up the simulation and improves frame rates by reducing the amount of data that needs to be processed by the application. Large matrices can be significantly reduced in size to improve running time. For instance, the subdivision matrix $\mathbf{A}$ as well as the vectors $\mathbf{d}$ and $\mathbf{p}$ can be partitioned into "free" $(\mathbf{A}_1, \mathbf{p}_1, \mathbf{d}_1)$ and "fixed" $(\mathbf{A}_2, \mathbf{p}_2)$ matrices:

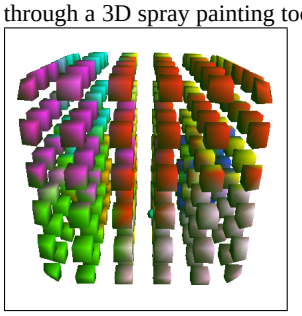
$$\mathbf{d} = \mathbf{A}\mathbf{p} \qquad \rightarrow \qquad \mathbf{d}_1 = \mathbf{A}_1\mathbf{p}_1 + \mathbf{A}_2\mathbf{p}_2$$

The fixed portions can be computed once and stored for look-up later. Other matrices such as $\mathbf{M}$, $\mathbf{D}$ and $\mathbf{K}$ can also be modified in a similar manner in order to decrease the number of variables that must be solved. Hence, for very large sculpted objects, the physical window is indispensable for maintaining interactive frame rates.

Fixed/free Points: The user can also fix and free control points individually to lock their positions in space. Used in conjunction with the physical window to mark regions, the point fix/free tool allows one to freeze the positions of any control points quickly.

7.4 Artistic Tool

Spray Painting: The user can add color to his sculptures through a 3D spray painting tool. Color is painted on both interior and boundary vertices. In our current implementation, both the interior and the exterior are rendered with polygons. In order to prevent occlusion by outer polygons, only those vertices with non-black and non-zero alpha values are rendered. Polygons are alpha-blended in order to indicate the color in the interior. Vertex color is smoothly blended between its existing color and the spray paint color while the tool is active. The radius of the spray paint tool can be set by the user in the GUI, while the paint color can be selected using a very convenient "color chooser" widget that is part of FLTK.



7.5 GUI Controls

In addition to the sculpting tools, our GUI contains several widgets that allow the user to change the global state of the model and various simulation parameters. The time-step can be modified to increase and decrease the speed of the simulation. When the implicit solver is in use, any time-step can be chosen and the system always remains numerically stable. The global damping factor is also manipulated by a slider and counteracts the potential of the mass-spring lattice to oscillate. The stiffness parameters of all springs can be reset to a new value by using the spring stiffness slider. A similar slider controls the stiffness parameters of the angular springs.

Model Name	Control Cells	Subdivided Cells	Update Time (ms)
Man's head	123	1069	114.7
Plesiosaur	29	232	30.5
Plesiosaur w/ open mouth	38	310	40.8
Soccer player	24	1536	85.0
Table	133	1288	146.4
Chair	76	744	82.5
Cave	50	400	40.5
Large cactus	19	152	20.3
I3D 2001 logo	30	240	30.44

Table 1: Dataset sizes and simulation timing information for some of the models described in this paper. All statistics are for one level of subdivision, except for the soccer player, which was subdivided twice.

By modifying both stiffness distributions, the user can sculpt virtual clay that behaves completely elastically to semi-elastically to completely plastically. The user can also change the stiffness (or strength) of the rope tool. High values of this stiffness allow the user to made large, sweeping deformations in a short amount of time.

The "Reset Rest Shape" button resets all normal and angular spring lengths to their current values. This has the effect of maintaining the current shape of the solid and is important for the creation of permanently deformed regions. Essentially, the tool redefines the rest shape of a sculpture to its current shape. Without this tool a sculpted object would always revert to its initial physical state because of the elasticity of the springs.

7.6 Example Sculpture: Human Head

We shall use the sculpted cartoon head in Figure 10 to describe how our sculpting tools can be used in practice. Since the sculpting of the head required the use of almost every design tool, it serves as a good example of how to use our sculpting system. We began with a $5 \times 5 \times 5$ block containing 125 control cells. Cells on the underside and back were deleted to give the head its general shape. The eye sockets were created by deleting one cell each and the eyes inserted by extruding a single cell from each hole. The nose was extruded from the front of the face and then shaped with the deformation tool. The mustache was extruded from the nose, bent, and slightly twisted. The mouth was created first by deleting a cell, by locally subdividing the lower jaw, and then by stretching it to create the thin lower lip and protruding chin. The ears were sculpted by extruding a few cells on either side of the head, locally subdividing them, and then stretching and denting them to make them thin and oval. The deformation and inflation tools were employed to give the top of the head a more rounded appearance. Then the deflation tool was used to smooth out bumps and pits. For certain features the physical window was activated to constrain the influence of the rope tool on the model. Finally, the surface of the solid was extracted and ray-traced offline.

8 Results and Time Performance

Table 1 details data sizes and run-time statistics for several dynamic models sculpted in our system. This table contains the running times only for the simulation loop. Note that to generate final images the models were subdivided several additional times to produce smooth boundary surfaces.

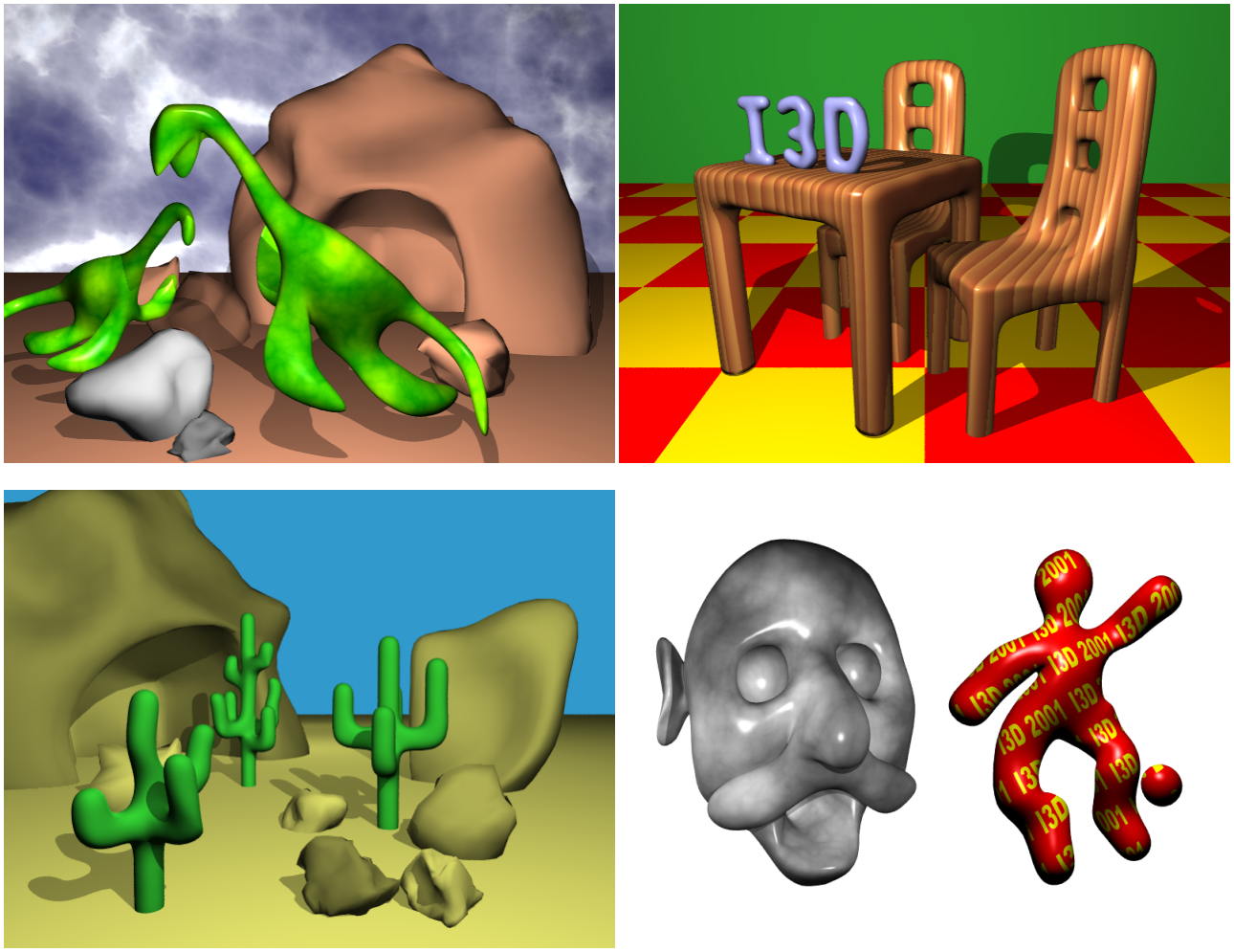


Figure 10: Several sculptures and scenes created entirely with our system and then composed and rendered with POV-Ray (www.povray.org). See also the color plate.

9 Conclusions

We have presented a new dynamic solid modeling framework and its intuitive, natural haptic interface based on the novel integration of subdivision solids and physics-based techniques. Our sculpting system permits the user to create real-world, complicated models in real-time using an extensive suite of geometry-, topology-, physics- and haptic-based virtual sculpting tools. Within our sculpting environment, the virtual clay responds to direct, user-applied forces in a predictable and intuitive manner while the haptic feedback can significantly enhance the sense of realism. Our physical formulation of topologically complex subdivision solids frees users from the need to deal with abstract geometric quantities and esoteric topological structures that often hinder the extensive use of sophisticated solid models on a large scale. To verify the applicability of our sculpting system on virtual clay and its powerful toolkits, we have conducted many sculpting sessions which have resulted in numerous interesting sculptures. The gallery of our virtual sculptures in Figure 10 should appeal to a wide spectrum of users including researchers, designers, animators, artists and even non-professionals. Future research involving dynamic subdivision solids includes data fitting applications, volumetric morphing, scientific visualization, physical simulation for flow dynamics and heat transfer, and others.

Acknowledgments

This research was supported in part by the NSF CAREER award CCR-9896123, the NSF grant DMI-9896170, the NSF ITR grant IIS-0082035, the GAANN grant P200A9703199, and a research grant from Ford Motor Company.

References

[1] R. Balakrishnan, G. Fitzmaurice, G. Kurtenbach, and K. Singh. Exploring interactive curve and surface manipulation using a bend and twist sensitive input strip. In *Proceedings of the 1999 ACM Symposium on Interactive 3D Graphics*, pages 111–118, 1999.

[2] D. Baraff and A. Witkin. Dynamic simulation of non-penetrating flexible bodies. In *Computer Graphics (SIGGRAPH 92 Proceedings)*, volume 26, pages 303–308, July 1992.

[3] D. Baraff and A. Witkin. Large steps in cloth simulation. In *Computer Graphics (SIGGRAPH 98 Proceedings)*, pages 43–54, July 1998.

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

[5] G. Celniker and D. Gossard. Deformable curve and surface finite elements for free-form shape design. In *Computer Graphics (SIGGRAPH 91 Proceedings)*, pages 257–266, July 1991.

[6] F. Dachille IX, H. Qin, A. Kaufman, and J. El-Sana. Haptic sculpting of dynamic surfaces. In *Proceedings of the 1999 Symposium on Interactive 3D Graphics*, pages 103–110, 1999.

[7] T. DeRose, M. Kass, and T. Truong. Subdivision surfaces in character animation. In *Computer Graphics (SIGGRAPH 98 Proceedings)*, pages 85–94, July 1998.

[8] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise smooth surface reconstruction. In *Computer Graphics (SIGGRAPH 94 Proceedings)*, pages 295–302, July 1994.

[9] D. L. James and D. K. Pai. ARTDEFO: Accurate Real Time Deformable Objects. In *Computer Graphics (SIGGRAPH 99 Proceedings)*, pages 65–72, Aug. 1999.

[10] K. I. Joy and R. MacCracken. The refinement rules for Catmull-Clark solids. Technical Report CSE-96-1, Department of Computer Science, University of California, Davis, Feb. 1999.

[11] R. MacCracken and K. I. Joy. Free-form deformations with lattices of arbitrary topology. In *Computer Graphics (SIGGRAPH 96 Proceedings)*, pages 181–188, August 1996.

[12] C. Mandal, H. Qin, and B. C. Vemuri. A novel FEM-based dynamic framework for subdivision surfaces. In *Proceedings of Solid Modeling '99*, pages 191–202, 1999.

[13] K. T. McDonnell and H. Qin. Dynamic sculpting and animation of free-form subdivision solids. In *Proceedings of IEEE Computer Animation 2000*, pages 126–133, May 2000.

[14] D. Metaxas and D. Terzopoulos. Dynamic deformation of solid primitives with constraints. In *Computer Graphics (SIGGRAPH 92 Proceedings)*, pages 309–312, July 1992.

[15] T. Miller and R. C. Zeleznik. The design of 3D haptic widgets. In *Proceedings of the 1999 Symposium on Interactive 3D Graphics*, pages 97–102, 1999.

[16] M. J. Muuss and L. A. Butler. Combinatorial solid geometry, boundary representations, and n-manifold geometry. In D. F. Rogers and R. A. Earnshaw, editors, *State of the Art in Computer Graphics: Visualization and Modeling*, pages 185–223. Springer-Verlag, 1991.

[17] A. Pentland and J. Williams. Good vibrations: Modal dynamics for graphics and animation. In *Computer Graphics (SIGGRAPH 89 Proceedings)*, pages 215–222, July 1989.

[18] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, second edition, 1992.

[19] H. Qin, C. Mandal, and B. C. Vemuri. Dynamic Catmull-Clark subdivision surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 4(3):215–229, July 1998.

[20] H. Qin and D. Terzopoulos. D-NURBS: a physics-based framework for geometric design. *IEEE Transactions on Visualization and Computer Graphics*, 2(1):85–96, Mar. 1996.

[21] M. A. Srinivasan and C. Basdogan. Haptics in virtual environments: taxonomy, research status, and challenges. *Computers & Graphics*, 21(4):393–404, July 1997.

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

[23] D. Terzopoulos and K. Fleischer. Modeling inelastic deformation: viscoelasticity, plasticity, fracture. In *Computer Graphics (SIGGRAPH 88 Proceedings)*, pages 269–278, August 1988.

[24] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *Computer Graphics (SIGGRAPH 87 Proceedings)*, July 1987.

[25] T. V. Thompson, D. E. Johnson, and E. Cohen. Direct haptic rendering of sculptured models. In *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, pages 167–176, 1997.

[26] K. J. Weiler. *Topological Structures for Geometric Modeling*. PhD thesis, Rensselaer Polytechnic Institute, August 1986.

[27] A. Witkin and M. Kass. Spacetime constraints. In *Computer Graphics (SIGGRAPH 88 Proceedings)*, volume 22, pages 159–168, Aug. 1988.

Appendix A: Geometry Rules of Subdivision Solids

Given an initial *control lattice* embedded in $\Re^3$, the solid subdivision rules in [11] recursively subdivide the lattice and refine the three-dimensional space occupied by the lattice. The lattice consists of a set of closed cells that are defined by a collection of their constituent faces. The faces in the lattice are comprised of an ordered list of edges that are defined by their end-vertices. Hence, there are four types of geometric entities in the lattice: cells, faces, edges and points.

The subdivision scheme recursively applies a set of four rules, one for each type of element, in order to achieve successively finer representations of the original lattice. Each element in the lattice produces a new vertex that must be subsequently incorporated into the next finer level of the subdivided lattice. The solid subdivision rules include:

- Cell points: for each cell, the cell point is its centroid.
- Face points: for each face, the face point is the weighted average: $\mathbf{f} = \frac{\mathbf{c}_0+2\mathbf{a}+\mathbf{c}_1}{4}$, where $\mathbf{a}$ is the face's centroid and $\mathbf{c}_0$ and $\mathbf{c}_1$ are the centroids of the cells on either side of the face.
- Edge points: for each edge, the edge point is the weighted average: $\mathbf{e} = \frac{\mathbf{c}_{avg}+2\mathbf{a}_{avg}+(n-3)\mathbf{m}}{n}$, where $\mathbf{c}_{avg}$ is the average of the centroids of the cells that contain the edge, $\mathbf{a}_{avg}$ is the average of the centroids of the faces that contain the edge, $\mathbf{m}$ is the midpoint of the edge, and n is the number of faces that contain the edge.
- Vertex points: for each vertex $\mathbf{p}$, the vertex point is the weighted average: $\mathbf{v} = \frac{\mathbf{c}_{avg}+3\mathbf{a}_{avg}+3\mathbf{m}_{avg}+\mathbf{p}}{8}$, where $\mathbf{c}_{avg}$ is the average of the centroids of the cells that contain the point, $\mathbf{a}_{avg}$ is the average of the centroids of the faces that contain the point, and $\mathbf{m}_{avg}$ is the average of the midpoints of the edges that contain the point.

The new lattice is then assembled as follows. Cell points are connected to the new face points of the faces that defined the cell; face points are connected to the new edge points of the edges that defined the face; and edge points are connected to the new vertex points of the vertices that defined the edge. Since the subdivision rules are not defined for faces, edges and points on the *boundary* of the solid, typically we employ the Catmull-Clark surface rules [4] for these geometric entities.