ELSEVIER

0010-4485(95)00029-1

Computer-Aided Design, Vol. 28, No. 4, pp. 251–262, 1996

0010-4485/96 $15.00 + 0.00

Deforming virtual objects interactively in accordance with an elastic model

HoSeok Kang and Avi Kak

We show how interactive deformations of a virtual 3D object can be carried out by using a hierarchical implementation of the finite element method (FEM). Basing deformations on the concepts of elasticity gives the human a measure of predictability when deciding where to apply forces to the object so that a desired shape would ensue. As is well known, one of the most powerful tools for analysing elasticity is the FEM, but the computational burden associated with a straightforward application of FEM to the problem at hand would make it too slow for any interactive process on even the fastest workstations. We have therefore developed a method in which the computational burden of FEM is alleviated by carrying out the FEM analysis at two different resolutions; a coarse resolution for a 3D calculation of the deformations and, subsequently, a finer resolution for just the surface layers of the object for a better (and smoother) delineation of the object shape. For the case of analysing the surface layers using the finer resolution, we show how a plate-theory version of FEM can be employed.

Keywords: virtual reality, interactive modelling, computer graphics, CAD, FEM, solid modelling, elastic deformation

INTRODUCTION

In this paper we show how a human can *interactively* deform an object that exists only in the memory of a computer. We are interested in deformations that do not grossly violate the basic principles of elastic theory, meaning that the deformations are not allowed to be arbitrary for given applied forces, although computational expediency may necessitate approximations that might cause small departures from the ideal. We use the familiar finite element method (FEM) for calculating the actual deformations but do so with a difference to alleviate the heavy computational burden associated with this approach. We use FEM in a two-resolution format. Initially, a coarse resolution is used for calculating gross deformations at a set of points in the object. Subsequently, the surface layers are analysed using a plate theory version of FEM for both a smoother determination of the surface shapes and for what is known as surface detailing.

In order to conform to a theory of elasticity, the deformations produced must be in response to forces applied to the object. In other words, the user must specify what forces to apply where. In our system, the applied forces may be specified either with the help of a mouse or with the help of a special force-input device we have built. The principal advantage of our force-input device over an ordinary mouse is that our device permits simultaneous and immediate specification of both the magnitude and the direction of an applied force. In this manner, via our force-input device, the applied forces can be made to be either longitudinal, or shear, or a combination of the two. The force-input device, shown with a power supply and an M68HC11EVB control board on the left in *Figure 1*, consists of four spring-loaded potentiometers, one at each corner of a pressure pad. The output of the potentiometers, proportional to the applied forces, is digitized and fed into a Silicon Graphics Personal Iris workstation for the virtual deformation experiments. As we shall explain later in this paper, the digitized signals are interpreted in different ways, depending on the mode chosen by the user.

To motivate the reader, we want to show at the very outset some results obtained with our interactive approach. The objects shown in *Colour Plate 1* were generated by repeated applications of forces from different sides to a cubic virtual stock. Another set of

Robot Vision Laboratory 1285 EE Building, Purdue University, West Lafayette, IN 47907-1285, USA
Paper received: 6 March 1994. Revised: 3 July 1995

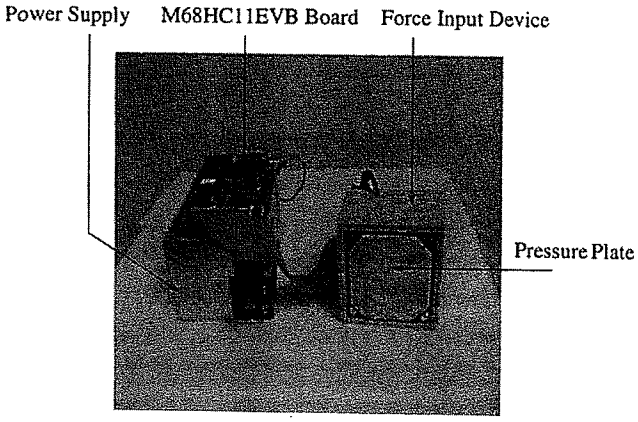
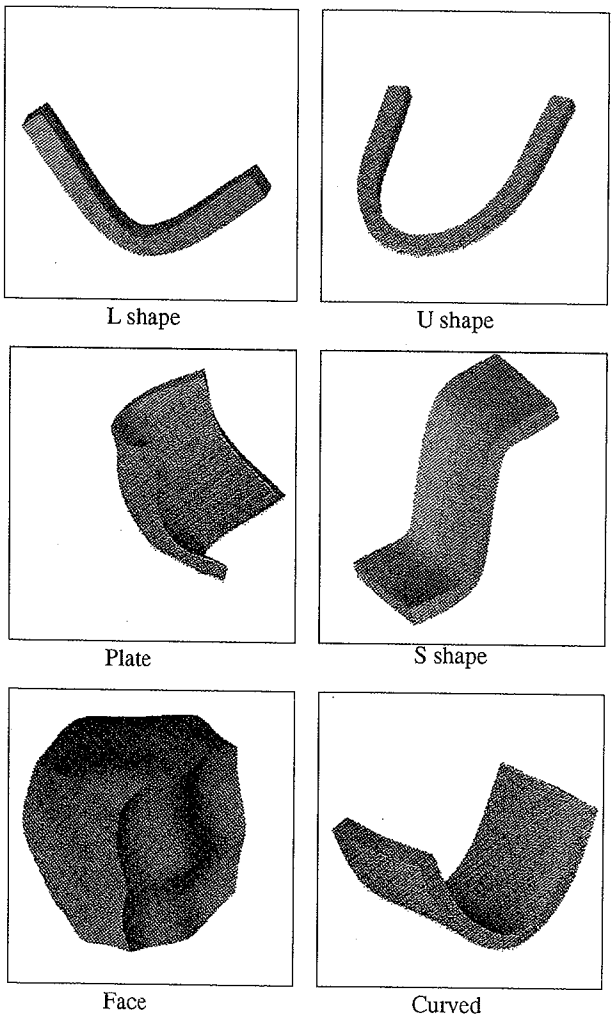
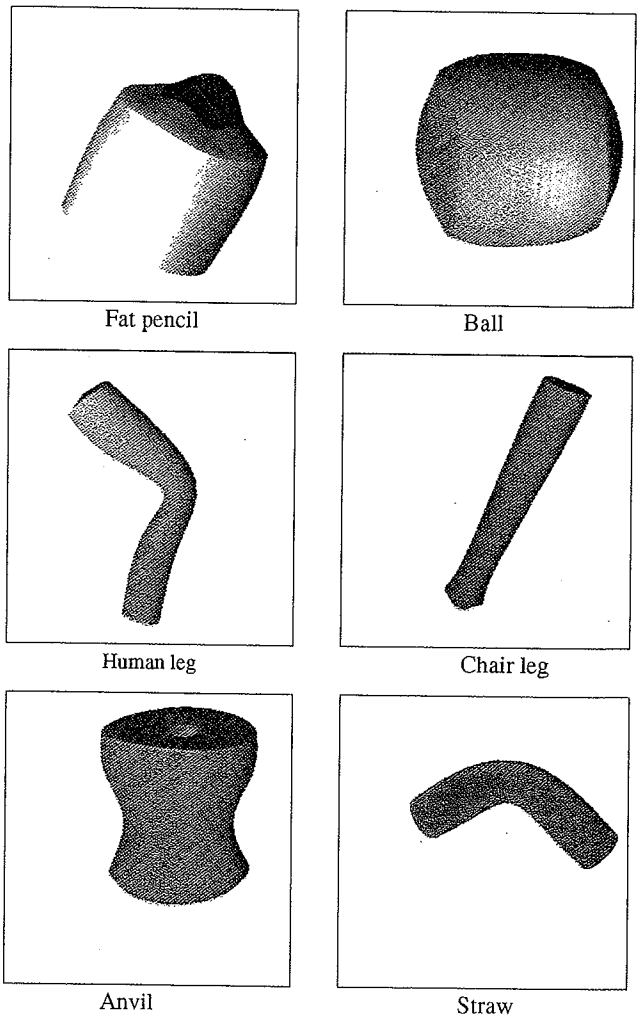


Figure 1 On the left is shown the microprocessor interface to the SGI workstation and on the right the force input device. The side of the force input device that is facing the reader consists of a pressure plate that is connected to four spring-loaded potentiometers

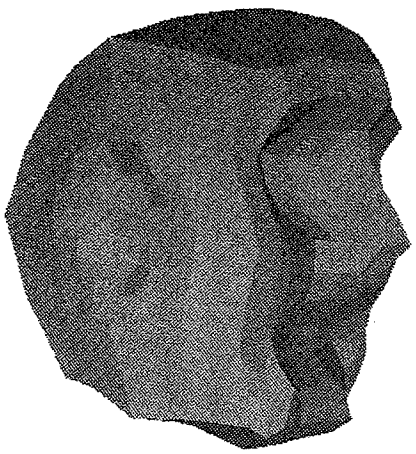
objects generated in a similar fashion from a cylindrical virtual stock is shown in *Colour Plate 2*. A more complex shape, which may be thought of as a ghost-like face, generated from a cube-shaped stock, is displayed in *Colour Plate 3*. The *CPU* times and the session times



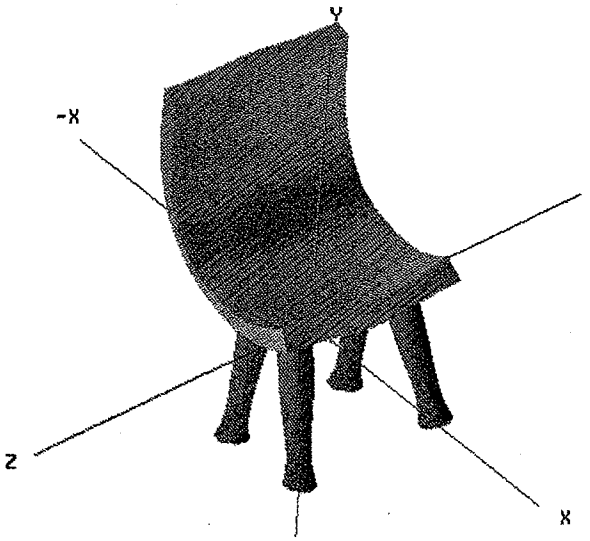
Colour Plate 1 Objects created by deforming a cube-shaped stock using coarse resolution. The total computation time (CPU time) needed for any of these objects ranged from 1 to 8 s on a SGI Personal Iris 4D/35 workstation. This CPU time corresponds to 2–5 interactive cycles that are needed for these objects. The total session time for any of the objects was under 5 min. The objects were displayed after bi-cubic spline interpolation of the coordinates of the deformations generated by our system



Colour Plate 2 Same as for *Colour Plate 1* except that these shapes were created from a cylinder-shaped stock using coarse resolution. The CPU times and the session times are again the same as for the objects in *Colour Plate 2*



Colour Plate 3 This ghost-like face was created from a cube-shaped stock using the two-resolution implementation of EEM. Total CPU time here was under 2 min on SGI Personal Iris 4D/35. This shape required 16 interactive cycles and the total session time was roughly 15 min



Colour Plate 4 Shapes synthesized by constructing Boolean combinations of some of the simple objects shown in *Colour Plates 1* and *2*

needed for these objects are indicated in the figure captions. Shown in *Colour Plate 4* is a Boolean combination constructed from some of the more primitive objects shown in *Colour Plates 1* and *2*.

In response to the applied forces, the deformations themselves are computed by using the finite element method (FEM). However, two major issues arise when applying FEM for virtual and interactive deformations. The first has to do with the computational burden of FEM. As will be stated in greater detail later, we have resolved this by using a two-resolution approach to the representation of objects and for the implementation of FEM. The second major issue has to do with the demands placed by FEM on a human user. Using FEM requires that the shapes of the elements into which the object is divided be specified, and then the user must somehow identify the nodal points, these being the junctions of the elemental shapes. Finally, in the traditional applications of FEM, the user is provided with some input/output facilities for specifying the forces applied to the object. As the reader can imagine, for a user to specify applied forces and boundary conditions textually requires that the user maintain in his/her mind the associations between the different nodal indices and the physical locations of those nodes on the object. All this makes using FEM a daunting task, expecially so for the non-specialist.

In this paper, we will describe a graphical system that hides all of the user-unfriendly details associated with the use of FEM. Our system displays on a screen all the nodes on the visible surfaces of an object and the user can select with mouse clicks the nodes for the application of the forces. During the first phase of interaction, the forces are used for the calculation of volumetric deformations using a coarse-resolution implementation of a volumetric FEM. Subsequently, in the second phase, the surfaces of the object are remeshed at a higher resolution. Forces applied subsequently yield displacements at the old nodes and, via a plate-theory based FEM, at the newly created surface nodes. Finally, in the third phase, surface detailing for fine deformations can be affected by imagining the surface nodes to exist on their own, without any connections to the interior, and by applying just the plate-theory based FEM to these surface nodes.

In what follows, we present on overview of the related literature. Next we very quickly go through some relevant concepts from finite element analysis. We then discuss the force-input device. Subsequently we discuss how volumetric FEM is coupled with a plate-theory based FEM, the former employing coarse spatial resolution for the delineation of overall shape and the latter finer resolution for creating more detailed surface effects. Finally, the overall graphical system is presented, followed by a summary and conclusions.

RELATED LITERATURE

Despite all the progress that has been made in the graphics modelling of the more geometrical industrial objects, the modelling of objects with free-form shapes remains a daunting task. This is not for lack of any published literature on the subject, since much has been published in recent years on how objects may be deformed interactively by displacing pre-designated control points or points on the surfaces[1–8]. Although evidently pioneering, these approaches are somewhat limited in their utility because, for complex free-form objects, the number of control point displacements needed for a desired shape can be too large and therefore inconvenient for a user. We have therefore declared our allegiance to physics-based methods. Here also there have been many pioneering publications[9–19], but more in the area of the dynamics of elastic objects. In the rest of this section, we will discuss in greater detail those previous contributions that have some intersection with our own work.

Barr's approach to modelling free-form deformations[1,2] is based on global and local transformations of superquadric surfaces. The concept of deformations by global transformations cannot be used directly for what we have in mind since the effects produced tend to be highly stylized and mostly symmetrical and therefore not suitable for our purposes. Sederberg and Parry[3] immerse the undeformed object in a coordinate frame in which a lattice of points, called the control points, is defined. The coordinates of the lattice points, especially after they are moved by a user to new locations, serve as coefficients of Bernstein polynomials that are used to determine the deformed positions of the object points. An object can be given an arbitrary deformation by displacing the control points to desired locations, the smoothness of the deformation of the object, *vis-à-vis* these displaced control points, is then ensured by the Bernstein polynomials. The work of Sederberg and Parry has been extended by Coquillart[4] by the use of non-rectangular lattices for control points, and by Lamousin and Waggenspack[5] by the use of NURBS. These approaches evidently have much merit when virtual sculpting is used as a metaphor for free-form geometrical modelling of shape, although manipulation of the control points can be daunting for complex shapes. Forsey and Bartels[6] have presented a B-spline based approach to the deformation of objects. We find this work interesting inasmuch as it is also hierarchical, in which a low-resolution mesh is used for generating an initial surface that for local refinements is subsequently

augmented with higher-density meshing wherever desired. A different and more direct approach to free-form deformation has been presented by Hsu, Hughes and Kaufmann[7] and Yamashita and Fukui[8]; in these two works the user is allowed to manipulate the surface points directly, as opposed to the control points. All of these methods, using either the manipulation of control points or of the surface points directly, can give the user better control of a deformation but at the expense of requiring a large number of inputs from the user.

Yet another approach to virtual sculpting captures in a computer the notion of a sculptor using a tool to move around and shape chunks of clay. This approach, pioneered by Galyean and Hughes[20], represents objects by voxels that can be squeezed out of a tube, cut away, melted away, sanded, etc., by an assortment of tools made available to the user in a manner similar to the use of 'paintbrush' for drawing and editing 2D imagery on personal computers.

All the methods mentioned above are devoid of any considerations pertaining to elasticity. What that means is that it would be possible to deform an object in a manner that would be physically impossible. These approaches lack connection between the deformations produced and the underlying physical reality.

In the second category are papers where the authors have tried to conform to the laws of physics, in the sense that the deformations do not violate elastic and plastic constraints. This type of work was pioneered by Terzopoulos and co-workers[9–13]. They define elastic and inelastic energy functions and solve them using finite difference methods. Using this approach they are able to simulate elastic, viscoelastic, plastic and fracture effects. A variation on this theme consists of using superquadrics that undergo parameterized global deformations in response to applied forces and boundary constraints[14, 15]. Most recently, Terzopoulos and Qin[16] have proposed the use of dynamic NURBS for interactive sculpting. While NURBS is a purely analytical description of a surface[21], in dynamic NURBS one associates mechanical properties with these surfaces so that their deformations can be studied by using dynamical equations.

Pentland and his co-workers[17,18] take a different tack at physically-based modelling of free-form deformations and show how precomputed low-order vibrational modes for certain pre-designated primitives can be used for this purpose. In the ThingWorld system that has ensued from this approach, elastic equations governing the deformations are diagonalized to result in 2nd-order time-dependent equations describing the various natural modes of vibration of an object. This approach therefore expesses an object deformation, produced in response to applied forces or forces of collision, as a linear sum of the natural modes of vibration. While the ThingWorld system appears ideal for deformations that have a certain global flow to them, we do not believe the system would be very efficient for deformations that are very local to a particular region of the object. For example, while ThingWorld can straightforwardly simulate pinching that is in the middle of a cylinder — this will generate shape perturbations primarily in the low-frequency modes — it would have a much harder time if one were to pinch asymmetrically one end of the cylinder. In that case, the user will have to decompose the asymmetric force into a sum of modal forces, which is non-trivial. On a slightly different but related note, Sclaroff and Pentland[19] use implicit functions and displacement maps for deformations. They represent an object with an implicit function and each point on the surface generated by that function can be displaced additionally by the value in the displacement maps.

All of these physically-based methods for deforming surfaces appear to be ideal for the generation of animation sequences. But if the goal is merely to create a static (but free-form) shape, there is less of a compelling reason for solving dynamic equations and the alternative approach presented by us in this paper might suffice. This is probably a good place to mention that all of the above physically-based approaches use only surface points; in other words the objects are considered to be hollow. On the contrary, our approach uses both volumetric and surface points; this can give a larger elastic coupling between distant points on and in an object, an immediate advantage being that a force application can result in a more pervasive global shape change.

FINITE ELEMENT METHOD: A BRIEF REVIEW

The principal goal of this paper is to deform virtual objects by the application of external forces. Evidently, this requires that the relevant equations, relating the displacements inside and on the surface of an object to the applied forces, be solved subject to whatever boundary conditions are appropriate. During the last couple of decades, a powerful tool called the 'finite element method' (FEM) has gained much currency for solving such problems[22,23]. Basically, FEM divides the object in question into a large number of discrete elements. Then, speaking figuratively, the applied forces are propagated through all the elements, via the nodes that connect the elements, in a manner that is consistent with the material properties of the object and the boundary conditions on the elements. The deformation of an object depends on its geometry, on the mechanical properties of the material, and on the appled force. Stress represents the intensity of force while strain represents the intensity of deformation. Usually, stress, strain, and displacement are related[24, 25]. In the rest of this section, we will first provide the reader with a brief introduction to FEM; this we do because FEM plays a central role in our system and we believe it is incumbent upon us to tell the reader how objects are meshed, how boundary conditions are satisfied, and what sort of equations are solved — issues that taken together form the art and science of FEM. Next we will discuss, again only briefly, the plate-theory based version of FEM. As we mentioned earlier, the volumetric and the plate-theory based FEMs constitute our two-pronged approach to creating object deformations.

Volumetric FEM

If we use only the linear terms in the displacement components while neglecting the higher order terms, the simplest relations between the components of the strain and the displacement components **u**, **v**, and **w** at

a point are

$$\begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{Bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{bmatrix} \begin{Bmatrix} \mathbf{u} \\ \mathbf{v} \\ \mathbf{w} \end{Bmatrix}$$

or

$$\{\epsilon\} = [B]\{\mathbf{u}\} \tag{1}$$

where $\{\epsilon\}$ and $\{\mathbf{u}\}$ are the strain and displacement vectors, respectively.

For linear, isotropic, and elastic materials, the relations between the components of stress and strain can be represented by using only two independent elastic constants: Young's modulus E and Poisson's ratio υ. They constitute the following specialization of the generalized Hook's law:

$$\begin{Bmatrix} \sigma_x \\ \sigma_y \\ \sigma_x \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{zx} \end{Bmatrix} = [D] \begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_x \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{Bmatrix} \tag{2}$$

where $[D]$ is given by:

$$\frac{E}{(1+\upsilon)(1-2\upsilon)} \times \begin{bmatrix} 1-\upsilon & \upsilon & \upsilon & & & & \\ \upsilon & 1-\upsilon & \upsilon & & & & \\ \upsilon & \upsilon & 1-\upsilon & & & & \\ & & & \frac{1-2\upsilon}{2} & & & \\ & & & & \frac{1-2\upsilon}{2} & & \\ & & & & & \frac{1-2\upsilon}{2} & \end{bmatrix}$$

Equation 2 may be written more compactly as

$$\{\sigma\} = [D]\{\epsilon\} \tag{3}$$

where $\{\sigma\}$ and $\{\epsilon\}$ are the stress and strain vectors, respectively.

Based on the above equations, we can set up a finite element system to compute displacements from force inputs. Consider, for example, a meshed object that is subject to some external force and we wish to compute the displacement at each of its nodes. A typical FEM first 'assembles' the material properties that exist in each of the elements into a large matrix called the global stiffness matrix. Next, the unknown displacements at all the nodes are 'assembled' into a large vector. Subsequently, the forces applied to the nodes of the object, either externally or internally, are 'assembled' into a single global force vector. And, finally, a vector–matrix equation is set up relating all of these composite vectors and matrices:

$$[\mathbf{K}]\{\mathbf{Q}\} = \{\mathbf{F}\} \tag{4}$$

where $\{\mathbf{K}\}$ is the global stiffness matrix, this matrix in general being symmetrical and banded, $\{\mathbf{Q}\}$ the global displacement vector, and $\{\mathbf{F}\}$ the global force vector. The resulting Equation 4 is then solved subject to whatever boundary conditions exist with regard to how the object is anchored, etc.

Before the FEM method can be invoked, it is necessary to select a mesh for the initial stock. Once a mesh is chosen for the stock, it remains with the object through all its deformations. In our work, the stock is either a cube or a cylinder. Consider first the case of a cube-shaped stock. It is usual to divide the stock into cubic cells as shown in *Figure 2a*. The finite element equations for cubic meshing may be derived either directly from a discretization of the elastic fields on the mesh or by taking into account the fact that a cubic mesh lends itself to the representation of each cubic cell by a union of five tetrahedral elements. If we assume that each tetrahedral element is small enough so that linear interpolation may be used to relate the displacement at any interior point to the displacements at the nodes of the tetrahedron, it is possible to derive a stiffness matrix that relates the nodal displacements to the forces applied to the tetrahedral element. Assembling all the nodal displacements into a single vector and doing the same for the nodal forces then yields the overall FEM equation, such as Equation 4 above. For details, the reader is referred to References 23 and 26.

It should be obvious to the reader that how the nodes are indexed in a given mesh should have an important bearing on the structure of the matrix [**K**] in Equation 4. As an extreme case, suppose we assigned indices randomly to the nodes, in that case the matrix [**K**] may not possess any regularity or symmetry that could be exploited computationally. On the other hand, if the nodes are assigned indices in the fashion shown in *Figure 2a*, we end up with a banded form for [**K**], as shown in *Figure 3*. An alternative way of indexing is shown in *Figure 2b*. The indexing of *Figure 2a* yields the tightest banding of the stiffness matrix and was therefore used for the work reported here for cube-shaped stocks.

In *Figure 4* we have shown what we believe is a novel meshing of a cylinder in terms of hexahedral elements. Because of its similarity to the meshing for a cube-shaped stock, both the cubic meshing and the cylindrical meshing can be described using only three parameters, c, r, p. In cubic meshing, c, r and p are the cell divisions of the stock along the three axes that define the stock. For cylindrical meshing, as shown in *Figure 4*, c is the number of layers in what we will refer to as

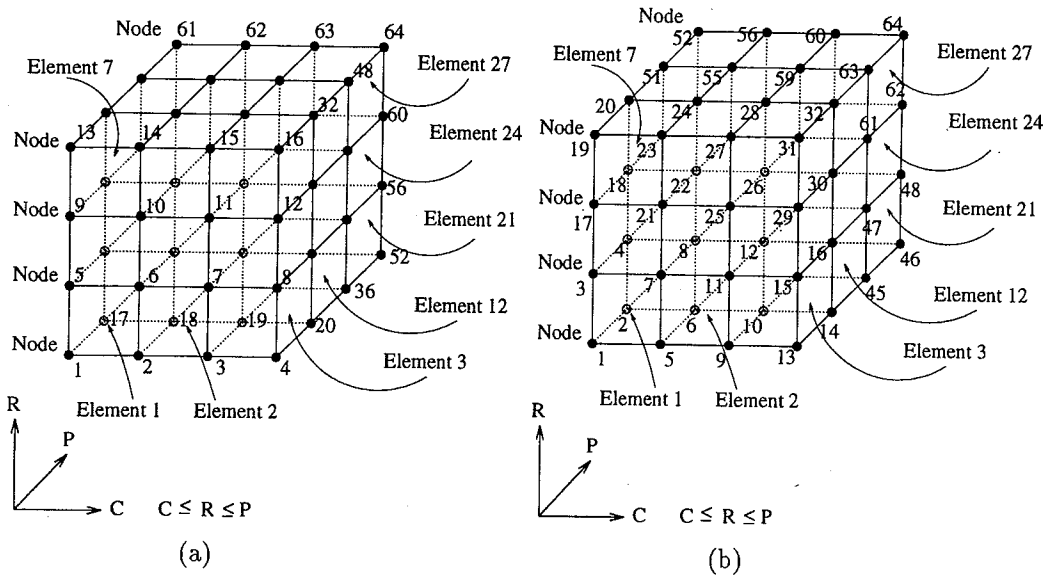


Figure 2 Different node indexing schemes for cubical meshing; (a) node indexing by sequential nodes, (b) node indexing by sequential cells

the cubic core, r the number of layers in the cylindrical outside, and p the number of layers in the axial direction. Note in particular that what we refer to as the cubic core of a cylindrical stock of unit dimensions is no different from a cubic stock shown in *Figure 2a*. For either the cubic stock or the cylindrical stock, all a user has to do is to specify the three parameters, c, r, and p. The system then automatically calculates the node coordinates from those parameters and indexes them appropriately. *Table 1* displays the expressions that yield the number of nodes, the number of faces, and the number of cubic elements for a given c, r, p for both cubic stock and cylindrical stock.

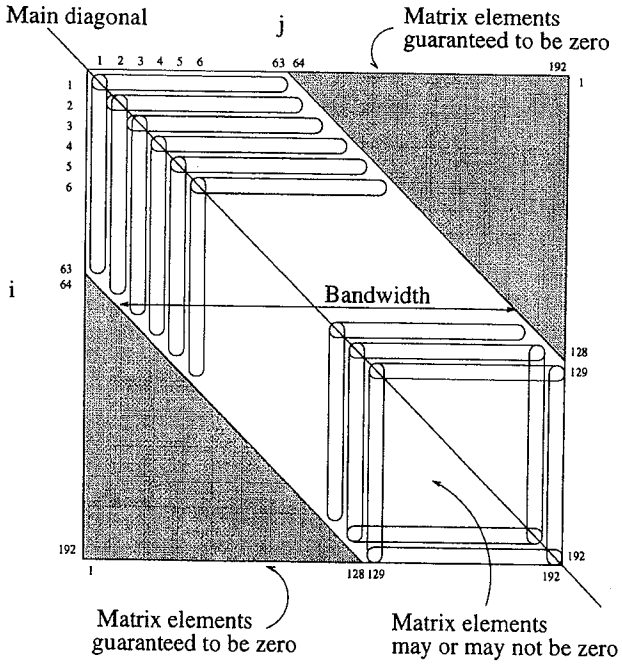


Figure 3 Symmetric and banded matrix [K] for the case of a cubic stock divided into $3 \times 3 \times 3$ cells

Plate-theory based FEM

We will now give the reader some insight into the fundamental assumptions that underlie the plate-theory based FEM[27, 28]. This discussion here will parallel our treatment of volumetric FEM.

For figuring out the deformations of thin plates, a plate of thickness h is considered to be a body that is bounded by two parallel planes, called its faces, whose lateral dimensions are large compared to the separation between these planes (*Figure 5a*). The plane parallel to and equidistance from the faces is called the midplane, or the median plane, of the plate.

The simplest and the most widely used plate theory is the *classical plate theory*, also known as the *Kirchhoff plate theory*. In this theory, a plate is assumed to be loaded transverse to the plane of the plate. It is assumed that the material points in the midplane can

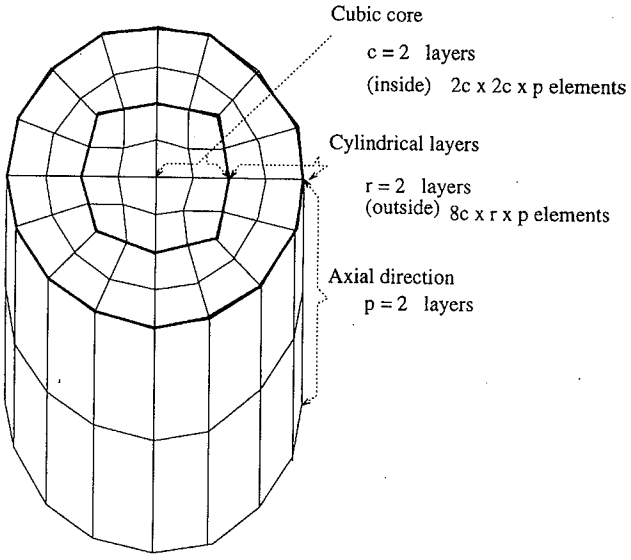


Figure 4 Cylindrical meshing

Table 1 Number of nodes, faces and cubes in a mesh

	Cube	Cylinder
Number of nodes	$(c+1)\times(r+1)\times(p+1)$	$(p+1)\times(r\times 8\times c+(2\times c+1)\times(2\times c+1))$
Number of faces	$(c\times r+r\times p+p\times c)\times 2$	$8\times c\times p+2\times r\times 8\times c+2\times 4\times c\times c$
Number of cubic cells	$c\times r\times p$	$8\times c\times r\times p+4\times c\times c\times p$

only be displaced along the transverse (or z) direction, so the midplane is a neutral plane (no stretch, no contract, no strain). It is also assumed that the straight material line elements that are perpendicular to the midplane in the undeformed state, such as corresponding to the line L in *Figure 5a*, remain straight and perpendicular to the midplane in the deformed state as the L' in *Figure 5b*, and that there is no thickness stretch of the plate. Further, it is assumed that the stresses normal to the midplane of the plate are neglible in comparison to the stresses in the plane of the plate, and the slope of the deflected plate in any direction is small. Since our interest in plate theory is confined to its use for the calculations of small deformations needed for surface detailing, etc., these assumptions are easily satisfied.

Let us consider the state of stress in a plate with an arbitrary small deflection $\mathbf{w}(x, y)$ at a point P in a plane at a distance z from the midplane. The slopes of the midplane are $S_x = \frac{\partial \mathbf{w}}{\partial x}$ and $S_y = \frac{\partial \mathbf{w}}{\partial y}$ so that the components of displacement vector $\mathbf{u}$, $\mathbf{v}$, $\mathbf{w}$ are given by

$$\mathbf{u} = -z\frac{\partial \mathbf{w}}{\partial x}$$
$$\mathbf{v} = -z\frac{\partial \mathbf{w}}{\partial y}$$
$$\mathbf{w} = \mathbf{w}(x, y) \tag{5}$$

In terms of the displacements, the strains are given by

$$\epsilon_x = \frac{\partial \mathbf{u}}{\partial x} = -z\frac{\partial^2 \mathbf{w}}{\partial x^2}$$

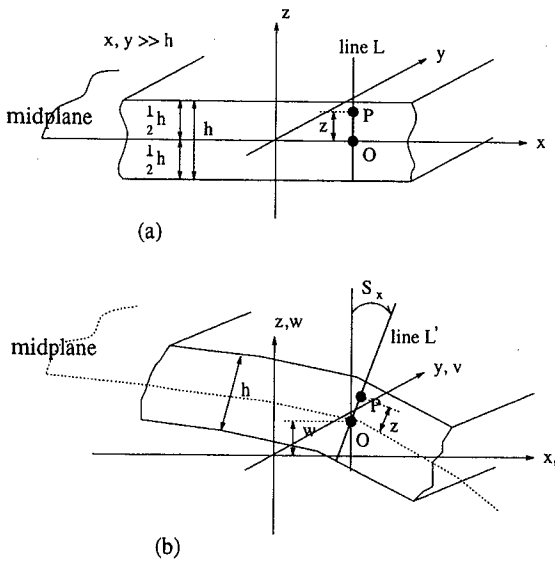


Figure 5 Deformation of a thin-plate; (a) plate before deformation, (b) plate after deformation

$$\epsilon_y = \frac{\partial \mathbf{v}}{\partial y} = -z\frac{\partial^2 \mathbf{w}}{\partial y^2}$$
$$\epsilon_{xy} = \frac{\partial \mathbf{u}}{\partial y} + \frac{\partial \mathbf{v}}{\partial x} = -2z\frac{\partial^2 \mathbf{w}}{\partial x \partial y} \tag{6}$$

and

$$\epsilon_z = 0,\ \epsilon_{yz} = 0,\ \epsilon_{xz} = 0 \tag{7}$$

in keeping with the assumptions listed previously.

The strains ϵ_x, ϵ_y, ϵ_{xy} are related to the stresses σ_x, σ_y, τ_{xy} by the relations

$$\epsilon_x = \frac{1}{E}(\sigma_x - \upsilon\sigma_y)$$
$$\epsilon_y = \frac{1}{E}(\sigma_y - \upsilon\sigma_x)$$
$$\epsilon_{xy} = \frac{1}{G}\tau_{xy} = \frac{2(1+\upsilon)}{E}\tau_{xy} \tag{8}$$

where E is Young's modulus, υ the Poisson ratio and G the shear modulus. Therefore, we can infer the following relationships between the stress components and the displacements:

$$\sigma_x = \frac{Ez}{1-\upsilon^2}\left(\frac{\partial^2 \mathbf{w}}{\partial x^2} + \upsilon\frac{\partial^2 \mathbf{w}}{\partial y^2}\right)$$
$$\sigma_y = \frac{EZ}{1-\upsilon^2}\left(\frac{\partial^2 \mathbf{w}}{\partial y^w} + \upsilon\frac{\partial^2 \mathbf{w}}{\partial x^2}\right)$$
$$\tau_{xy} = -\frac{Ez}{1+\upsilon}\left(\frac{\partial^2 \mathbf{w}}{\partial x \partial y}\right) \tag{9}$$

These stresses vary linearly through the thickness of the plate and are equivalent to moments per unit length acting on an element of the plate. Thus,

$$M_x = \int_{-h/2}^{h/2} z\sigma_x \mathrm{d}z = -D\left(\frac{\partial^2 \mathbf{w}}{\partial x^2} + \upsilon\frac{\partial^2 \mathbf{w}}{\partial y^2}\right)$$
$$M_y = \int_{-h/2}^{h/2} z\sigma_y \mathrm{d}z = -D\left(\frac{\partial^2 \mathbf{w}}{\partial y^2} + \upsilon\frac{\partial^2 \mathbf{w}}{\partial x^2}\right)$$
$$M_{xy} = \int_{-h/2}^{h/2} z\tau_{xy} \mathrm{d}z = -D(1-\upsilon)\frac{\partial^2 \mathbf{w}}{\partial x \partial y} \tag{10}$$

where the flexural rigidity D of the plate is defined by

$$D = \frac{Eh^3}{12(1-\upsilon^2)} \tag{11}$$

and M_x, M_y and M_z are the internal plate stresses or the bending moments. Note that because of the equality of complementary shears τ_{xy} and τ_{yx}, it follows that

$$M_{xy} = M_{yx} \tag{12}$$

Based on these relations, we can set up a system of equations relating displacements at a set of mesh points and the forces (and moments) in a manner similar to what was done for the volumetric case.

INTERPRETATION OF FORCE INPUTS

Although the user has the option of using the mouse for specifying forces, with regard to their locations, directions, and magnitudes the interaction with the SGI machine is much facilitated if the user uses our force-input device. Although simple in construction, it permits simultaneous specification of force magnitudes and directions in a couple of different modes. This device basically generates four scalar numbers in response to the user pressing on a pressure plate mounted on four position-sensing elements. It is possible to interpret these four scalar numbers in different ways for the purpose of deforming the object. In our experiments, the user can choose from three different interpretation modes.

In the first interpretation mode, the four scalar numbers are thought of as forces applied at four different mouse-selected points on an object face, the direction of the forces assumed to be perpendicular to the face, as illustrated in *Figure 6*. In part *a* of the figure, the four force values output by the device are assumed to be applied perpendicularly on face *F* at node points A, B, C, and D that are selected by the user with mouse clicks. The force values actually fed into the finite element algorithm are obtained from the four device-generated values by linear interpolation, as shown in part *b* of the figure. This capability, meaning the ability to apply non-uniform forces on the pressure plate, allows the system to simulate the application of shear forces. The applied force is assumed to be normal to the selected points or faces. The normal direction of a face is computed from the four edges in the face using well-known formulas and the normal direction associated with a point is the average of the normals of all the faces to which that point belongs. The latter definition is important for object points that are on the boundaries between different faces.

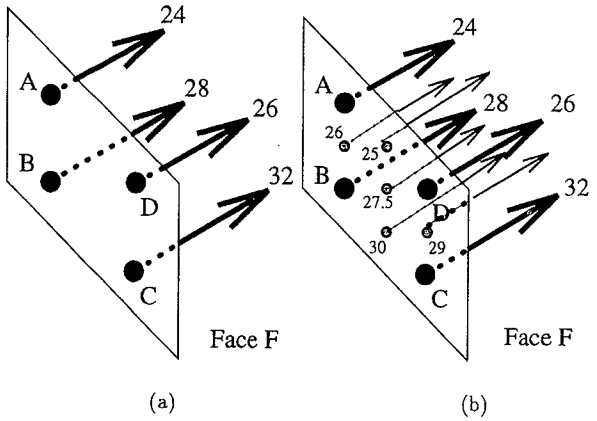


Figure 6 Shown here is the first of the three modes in which it is possible to interpret the output of the signals produced by the force-input device; (a) the four values produced by the device, (b) the interpolated values

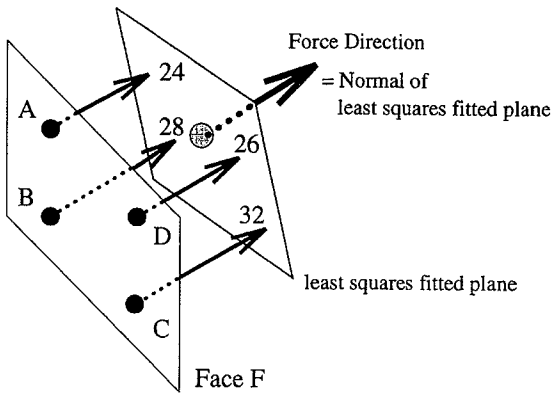


Figure 7 Shown here is another mode for interpreting the values output by the force-input device. Force direction now is a resultant of the four values output by the device

In the second interpretation mode, the four scalar numbers spit out by the force-input device are used for not only specifying the magnitude but also the direction of the force with respect to the local normals. However, the applied force is now assumed to be uniform over a mouse-selected cluster of nodes. To explain further, as shown in *Figure 7*, we interpret the four values as vectors perpendicular to the face at four neighbouring points surrounding a node. Using a least-squares fit, a plane is then drawn through the tips of these vectors. The normal to this plane into the object is considered to be the direction of the applied force, and the force itself considered constant in value over a selected cluster of nodes. Therefore, while the first interpretation allows us to simulate shear effects, the second allows application of deformation forces along arbitrary directions to a selected face.

In the third interpretation, the user mouse-selects a set of points that encircle the object. For example, if the user wishes to pinch a cylinder symmetrically around the axis, the user would select a set of point on a circle around the cylinder. The four signals generated by the force transducer are now averaged and this average is construed as a uniform force applied at all the mouse-selected points, the direction of the force at each point being along the local normal.

In each of the interpretaion modes, it is necessary that the user also mouse-select some of the nodes as boundary nodes where either no displacements are allowed or where the displacements must equal certain pre-set values. (If no nodes are specified as boundary nodes in this manner, the object will simply translate as opposed to deform in reponse to applied forces.) Since the user is allowed to rotate the object about any of the axes during this interaction process, the nodes that might otherwise be occluded in some view can also be selected for either force application or as boundary nodes by merely rotating the object.

As stated earlier, the force-input device is made up of four spring-loaded potentiometers attached to the four corners of a pressure plate. Each potentiometer of the device is connected to a different channel of the A/D converter on the M68HC11EVB board. On each

channel, the analogue voltage output by the potentiometer is converted into a digital number by taking four samples and averaging them. Four digital numbers from each channel are then transmitted from a serial port on the M68HC11EVB board to an SGI Personal Iris workstation. The four maximum numbers, one for each channel, are recorded and used as force input values for computation as explained previously. By calibrating the springs in the device with a scale, correct force scale is computed and multiplied to the force input values.

The user applies pressure on the pressure plate of the device; the user is free to press non-uniformly at the four corners of the pressure plate. As mentioned before, the resulting four signals may either be interpreted as four separate pressure values applied at four mouse-designated points on a mouse-selected face of the object, or as denoting the magnitude and direction of a uniform force applied to some mouse-selected cluster of nodes.

TWO-RESOLUTION IMPLEMENTATION OF FEM

The interactive nature of our research effort requires that the calculation of the deformations proceed on a real-time or near real-time basis. That rules out the use of many of the approaches for deformation calculations that have been published during the last several years in the graphics community. What has worked well for us is a two-resolution implementation of the time-honoured finite element method. As will be discussed here, we first use coarse resolution and a volumetric version of FEM for calculating the gross displacements of the nodes of the mesh. Then we use a finer resolution for the surface layers of the object and a plate-theory based version of FEM to determine the surface shape more precisely. The plate theory version of FEM can also be used for applying further forces to the object for what is known as surface detailing. For those readers who are not that conversant with FEM, our two-resolution implementation of FEM is necessitated by the fact that the usual volumetric FEM requires processing $3(\mathbf{N})^3 \times 3(\mathbf{N})^3$ matrices for an $\mathbf{N} \times \mathbf{N} \times \mathbf{N}$ meshing of an object. Notwithstnading the sparseness of these matrices, experience tells us that for real-time calculations $\mathbf{N}$ cannot be allowed to become large on typical laboratory sized workstations.

More specifically, the following steps outline our basic computational strategy for the calculations of the deformations:

Step 1: Mesh the stock at a coarse resolution.
Step 2: Set boundary conditions and apply forces to the object via a force-input device and calculate the gross deformations using volumetric FEM.
Step 3: Repeat Step 2 until the coarse shape is as desired (*Figure 8*).
Step 4: Remesh each group of surfaces of the deformed stock at a higher resolution*.
Step 5: Apply further forces as needed and use the low-resolution volumetric FEM to calculate the displacements at each of the low-resolution nodes. And, then, use the plate-theory version of FEM to calculate the displacements at the denser array of points on the face on which the forces are applied. Use linear interpolation for the other faces.
Step 6: Repeat Step 5 until the shape is as desired (*Figure 9*).
Step 7: For final surface detailing, assume the object is made of just the surface nodes. In other words, the object is now a shell consisting of many deformed faces. Apply additional forces as necessary to any desired face and use the plate-theory based FEM to calculate at high-resolution the deformations of that face (*Figure 10*).

Our two-resolution approach, with volumetric FEM applied at a coarse resolution and a plate-theory version of FEM applied at a finer resolution, raises the issue of 'connecting' the two separate solutions. Fortunately, this is not a real problem because the displacements produced by low-resolution computations are used as boundary conditions for the plate-theory based FEM calculations carried out at the finer resolution.

THE OVERALL SYSTEM

As shown in *Figure 11*, the overall system contains four modules: *mesh generator, matrix builder, solver* and *display*. The mesh generator accepts parameters from the user regarding the cell decomposition and then generates an appropriate mesh. The user is also allowed a choice of predefined meshes. In this way, we can start the deformation from not only the cubic or cylindrical meshes but also somewhat arbitrary meshes. Then the user chooses material properties and geometry. Using this information, the mesh generator and the matrix builder modules are then fired to build the mesh and the stiffness matrix of Equation 4. 3D views of the object and the mesh are then displayed in different windows. Next, the user mouse-selects nodes for force application and for boundary conditions. The user then deforms the object by pressing on the pressure plate of

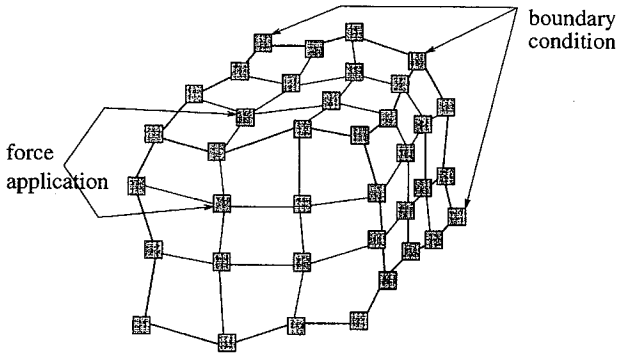


Figure 8 First phase of deformation corresponding to Steps 1–3

*It is important to note here that while the deformation of Step 2 will result in a single planar face deforming into what could be approximated by many planar faces, the original nodes are still grouped according to the original face assignments. Therefore, when we talk about remeshing a face at a higher resolution, we are referring to one of the original faces.

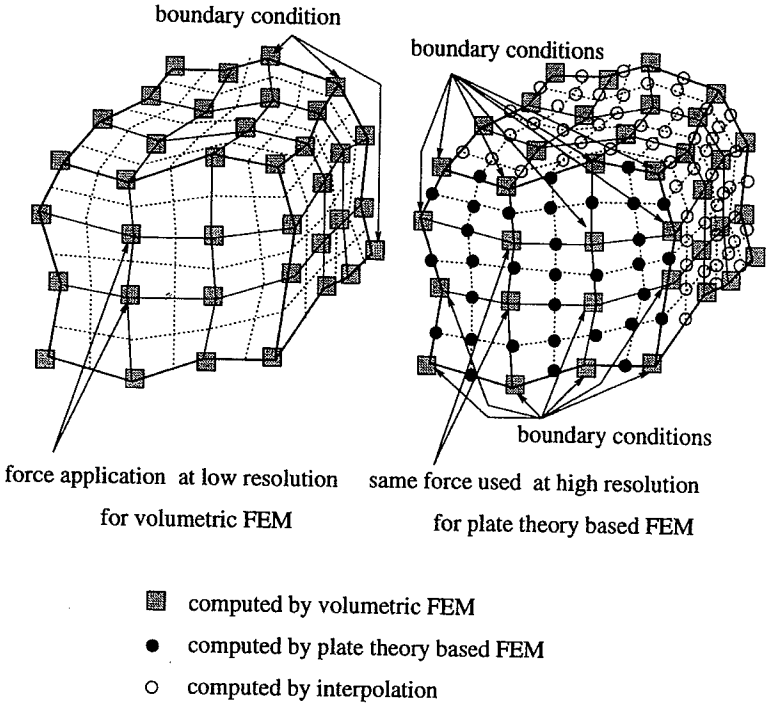


Figure 9 Second phase of deformation corresponding to Steps 4–6. On the left are the low resolution nodes, shown as black squares, whose displacements are obtained by solving volumetric FEM equations. The displacements at the higher resolution nodes, shown as black dots on the right, are the outputs of the plate-theory based FEM using as boundary conditions the displacements of the low resolution square nodes. At all the other nodes, shown as open circles, linear interpolation is used for the calculation of the displacements

the force-input device. The screen dump in *Figure 12* shows on the left the main menu available to the user. The lower right portion of the screen shows an information block that contains the values of various and sundry parameters. Parameter values are set by clicking on the menu items and choosing an appropriate value from a window that appears to the left of the information block. For example, the view direction is selected by using the sliders in the pop-up window labelled 'Moving Direction'.

In keeping with our earlier discussion, the overall computation proceeds in three phases. The first phase consists of Steps 1–3 listed earlier, the second phase of Steps 4–6, and the final phase of Step 7. In the first phase, the phase in which gross displacements are calculated using a coarse resolution representation of the object, the user can apply forces to any node in any face. In the second phase, while the forces are still applied to the surface nodes corresponding to coarse resolution, the deformation calculations are carried out at both the coarse resolution for the entire object and using plate-theory based FEM for just the face to which the force was applied, the latter calculation using as boundary conditions the surface deformations obtained at the coarse level. During this phase, while the plate-theory based equations are not directly used for the other faces, at these other faces linear interpolation is used to calculate the deformations at the nodes corresponding to the finer resolution. Finally, in the last phase the calculations proceed under the assumption that the object is just a shell consisting of only the surface nodes. The user is allowed to apply additional forces to the high-resolution representation for one face at a time and the deformations of the nodes in that face computed again by using plate-theory based FEM.

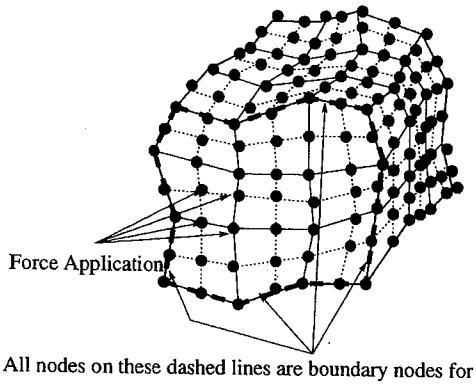


Figure 10 Final phase of deformation corresponding to Step 7

Each interactive force application is recorded into a script file that can be used to reproduce the same object later by executing the file as a batch job. Since each deformation is a function of the material properties used, we can create different objects very easily by just changing the material properties. Each object is also represented by 3D coordinates for each of the nodes in the mesh. The boundary representation of the outside surfaces is computed from these nodal coordinates for the surface nodes. By using bi-cubic spline interpolation on the surface nodes, the system can create smoother surface models. When the system initially creates a mesh, the surface nodes are grouped by six general viewing directions; top, bottom, left, right, front and back. This group of six surface nodes is interpolated separately. For creating smooth surfaces for polygonal objects, the system can also use natural bi-cubic splines if supplied with zero 2nd-order deriva-

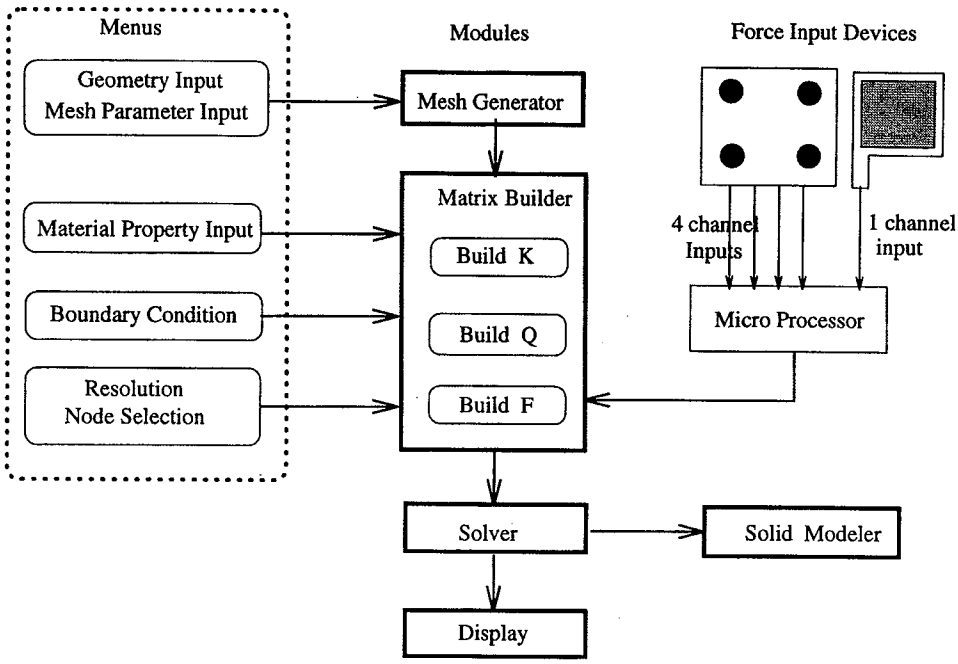


Figure 11 Overall flowchart for our system

tives on the boundary edges for each of the surfaces. For objects that are meant to round all around, specially computed 2nd-order derivatives at the boundary edges of the surfaces are used by the interpolation routine.

After creating deformed objects, if desired, these boundary representations can be melded together through Boolean combinations for creating more complex objects (*Figure 13*). This procedure is very similar to actual object creation in real world: first make small parts and then assemble those parts into a larger and more complex object.

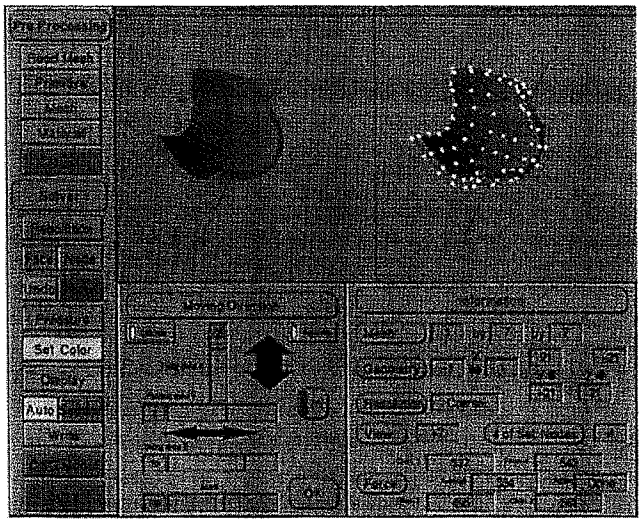


Figure 12 Shown here is a screen dump of our system. The main menu is in the column on the left. The top-left window displays a 3D view of the object. The top-right window displays the mesh of the object. The user can mouse-select nodes from this window. An information block in the lower-right shows the values of some of the more important parameters. The block between the main menu and the information block depends on which item is selected from the main menu. The block displayed here illustrates the sliders that can be used for viewing the object from different directions

CONCLUSIONS

We demonstrated a computationally feasible approach for interactive deformation of virtual 3D objects. We chose the framework of elastic theory to govern the deformations. This gives the human operator a comforting edge of predictability when deciding where to apply forces to the object so that a desired shape would ensue. Basing deformations on elastic theory necessitated that our computational approach be centred on finite element methods. Unfortunately, FEM as presented in the literature could not be used directly due to its enormous computational burden for mesh sizes that would be of interest for what we had in mind. We therefore developed a two-resolution approach that combines the volumetric version of FEM using coarse resolution with a plate-theory based FEM at a higher

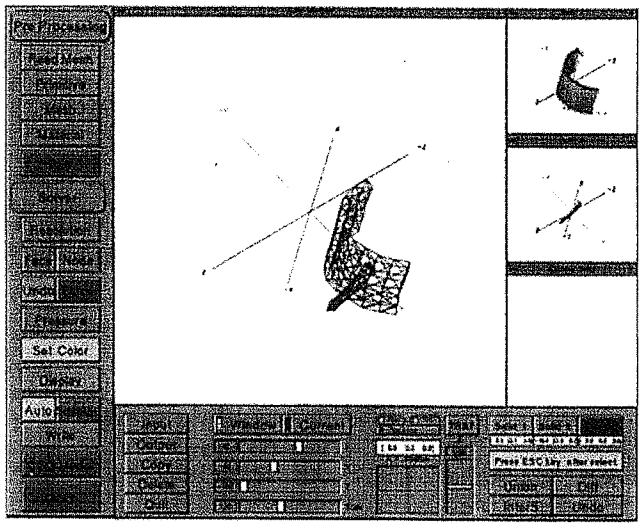


Figure 13 Shown here is a sample screen from our interactive solid modelling system showing how Boolean combinations are composed. The user can select two objects and construct their Boolean combinations interactively

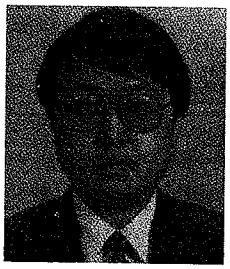
resolution. Linking the two solutions did not present any difficulties as we used the deformations produced by the volumetric FEM as boundary conditions for the plate-theory based deformations of the surface nodes. For fine surface detailing, we employed a third phase of computations in which the object is considered to be a shell consisting of just the surface nodes; now only the plate-theory based FEM was used for shape effects at a local level on the surface.

The reader might ask why not just use the plate-theory version of FEM. Although computationally that would certainly be feasible, it would alter fundamentally the nature of interaction between the system and the user and it would also prolong the length of time needed for creating a certain shape. If the object is considered to be merely a shell, as it will have to be for the plate-theory based FEM, the deformations produced by a force applied at any point on the surface would result in rather localized surface undulations.

For examples of our final results, the reader's attention is drawn again to *Colour Plates 1–4*. The information pertaining to the total CPU time needed and the session time is mentioned in the figure captions.

REFERENCES

1 Barr, A H 'Superquadrics and angle-preserving transformations' *IEEE Comput. Graph. Appl.* Vol 1 (1981) pp 11–23
2 Barr, A H 'Global and local deformations of solid primitives' *SIGGRAPH* Vol 18 (1984) pp 21–30
3 Sederberg, T W and Parry, S R 'Free-form deformation of solid geometric models' *SIGGRAPH* Vol 20 (1986) pp 151–160
4 Coquillart, S 'Extended free-form deformation: a sculpturing tool for 3D geometric modeling' *SIGGRAPH* Vol 24 (1990) pp 187–193
5 Lamousin, H J, Warren, N and Waggenspack, J 'NURBS-based free-form deformations' *IEEE Comput. Graph. Appl.* Vol 14 (1994) pp 59–65
6 Forsey, D R and Bartels, R H 'Hierarchical B-spline refinement' *SIGGRAPH* Vol 22 (1988) pp 205–212
7 Hsu, W M, Hughes, J F and Kaufman, H 'Direct manipulation of free-form deformations' *SIGGRAPH* Vol 26 (1992) pp 177–184
8 Yamashita, J and Fukui, Y 'A direct deformation method' *IEEE Virtual Reality Annual Int. Symp.* (1993) pp 499–504
9 Kass, M, Witkin, A and Terzopoulos, D 'Snakes: active contour models' *Int. J. Comput. Vis.* (1987) pp 321–331
10 Terzopoulos, D, Witkin, A and Kass, M 'Symmetry-seeking models and 3D object reconstruction' *Int. J. Comput. Vis.* (1987) pp 211–221
11 Terzopoulos, D, Witkin, A and Kass, M 'Constraints on deformable models: recovering 3D shapes and nonrigid motion' *Artifical Intelligence* Vol 36 (1988) pp 91–123
12 Terzopoulos, D, Platt, J, Barr, A and Fleischer, K 'Elastically deformable models' *SIGGRAPH* Vol 21 (1987) pp 205–214
13 Terzopoulos, D and Fleischer, K 'Deformable models' *Vis. Comput.* Vol 4 (1988) pp 306–331
14 Terzopoulos, D and Metaxas, D 'Dynamic 3D models with local and global deformations: deformable superquadrics' *IEEE Trans. PAMI* Vol 13 (1991) pp 703–714
15 Metaxas, D and Terzopoulos, D 'Dynamic deformation of solid primitives with constraints' *SIGGRAPH* Vol 26 (1992) pp 309–312
16 Terzopoulos, D and Qin, H 'Dynamic nurbs with geometric constraints for interactive sculpting' *ACM Trans. Graph.* Vol 13 (1994) pp 103–136
17 Pentland, A P and Williams, J 'Good vibrations: modal dynamic for graphics and animation *SIGGRAPH* Vol 23 (1989) pp 215–222
18 Pentland, A, Essa, I, Friedmann, M, Horowitz, B and Sclaroff, S 'The ThingWorld modeling system: virtual sculpting by modal forces' *SIGGRAPH* Vol 24 (1990) pp 143–144
19 Sclaroff, S and Pentland, A P 'Generalized implicit functions for computer graphics' *SIGGRAPH* Vol 25 (1991) pp 247–250
20 Galyean, T A and Hughes, J F 'Sculpting: an interactive volumetric modeling technique' *SIGGRAPH* Vol 25 (1991) pp 267–274
21 Piegl, L 'On NURBS: a survey' *IEEE CG & A* Vol 11 (1991) pp 55–71
22 Bathe, K-J *Finite Element Procedures in Engineering Analysis* Prentic Hall (1982)
23 Chandrupatla, T R and Belegundu, A D *Introduction to Finite Elements in Engineering* Prentic Hall (1991)
24 Kikuchi, N *Finite Element Methods in Mechanics* Cambridge (1986)
25 Boresi, A P, and Chong, K P *Elasticity in Engineering Mechanics* Elsevier (1987)
26 Grandin Jr., H *Fundamentals of The Finite Element Method* Macmillan Publishing (1986)
27 Reddy, J N *An Introduction to the Finite Element Method* McGraw-Hill (1993)
28 Reismann, H *Elastic Plates: Theory and Application* John Wiley (1988)



HoSeok Kang received a BS degree from Seoul National University, USA, in 1985, and an MS degree from the University of Florida, in 1988, both in electrical engineering. He received a PhD degree in electrical engineering at Purdue Universtiy, USA, in 1994. From 1989 to 1995, he was a member of the Robot Vision Laboratory in the School of Electrical Engineering, Purdue University, USA. He is currently with SAIT (Samsung Advanced Institute of Technology), Korea. His research interests are in the areas of computer graphics, solid modelling, virtual reality, game developing, computer vision and image processing. He is a member of Eta Kappa Nu. IEEE and ACM.

Avi Kak is currently serving a 2-year term as an IEEE Distinguished Lecturer in the Robotics and Automation Area. He is a coauthor of the widely used Digital Picture Processing *(Academic Press, 1982) and a coauthor of a forthcoming book entitled* Robotic Planning.