ICAT'98

Representation of Soft Objects in Virtual Environment

Koichi HIROTA, and Toyohisa KANEKO

Department of Information and Computer Sciences
Toyohashi University of Technology
1-1 Hibarigaoka, Tempaku-cho, Toyohashi, Aichi 441-8580, Japan
hirota.kaneko@mmip.tutics.tut.ac.jp

Abstract

In this paper, a method to haptically represent an elastic object in a virtual environment is proposed. The elasticity of an object is defined by a linear FEM (finite element method) model. By calculating the inverse stiffness matrix in advance, the displacement of nodes according to the applied force is obtained with a low calculation cost.

The pre-calculated inverse stiffness matrix must remain constant for it to be used during the operation. To preserve the matrix, we converted the displacement boundary condition induced by the user's finger into the force boundary condition.

In our prototype implementation, we employed the god-object method to determine the contact point on the object and to calculate the displacement at that point. We applied the proposed method to two objects, a cube and an object shaped like a mouse, which consist of 125 and 1713 nodes, respectively. By measuring the computation time for those objects, we confirmed that the method is applicable to the real-time representation of the motion of elastic objects.

Key words: FEM, Elastic Model, Deformable Object, Force Feedback, Virtual Object

1. Introduction

The representation of force sensation has become an important topic of research for the improvement of reality in a virtual environment. In particular, the stiffness of an object is information that is transmitted only by the sensation of force. Also, the implementation of deformable objects is important for such an application as surgical simulation[1].

Various approaches have been proposed for the visual representation of geometrical deformation. However, such visually motivated deformation models do not provide information on haptic sensation related to the deformation. We need models to represent a physically deformable object. There are two typical models available: spring-network model[2], and FEM (finite element method) elastic model[3].

The FEM is an excellent approach in that both the model and the calculation method are physically well defined. However, FEM requires a high computation cost; therefore, it is not suitable for real-time computation. An investigation was made on the real-time simulation based on FEM using high-performance computers[4]. However, at present, this approach is not feasible for conventional virtual reality systems.

The spring-network model requires less time for one cycle of deformation calculation. This is the reason why many studies use this model to represent deformable objects. However, this model has a problem that the meaning of parameters such as spring constants are not clear in the physical sense, because the stiffness of the object depends not only on those spring constants but also on the topological structure of the network. Another problem is that the transient behavior (i.e., the speed of deformation) of the model is relatively slow. Consequently, the model is not suitable to applications requiring quick reactions of the object during a deformation operation.

In this study, we investigated a method of applying linear FEM model to the real-time representation of stiffness. As we described above, FEM is generally not suitable for real-time simulation. In the FEM, most of the computation cost is consumed by solving the simultaneous equations defined by the 'global stiffness matrix'.

However, even when using the FEM, if we apply a linear elastic model and preserve the displacement boundary condition, we can reduce the computation cost significantly. Namely, under the above assumption, the global stiffness matrix becomes a constant matrix; therefore, we can calculate the inverse matrix in advance. Using the inverse matrix, we can easily obtain the displacement of nodes from the force boundary condition.

Since the linear elastic model is an approximate representation that is applicable only to deformations with a small displacement, the model gives an unnatural result when it is applied to the simulation of large deformations. However, for the haptic representation of stiffness, this problem does not arise, because stiff-

ness is usually recognized by deformation operations with relatively small displacements.

Most of the available force feedback devices provide the position of the user's finger with respect to the system. Therefore, the contact of the finger with the object causes an additional displacement boundary condition in the model. As was mentioned above, we must preserve the displacement boundary condition so that we can use the precalculated inverse matrix. We avoid the change of the displacement boundary condition by converting the additional displacement boundary condition into the equivalent force boundary condition.

2. Calculation

2.1. Force-Displacement Relationship

In the FEM[5], an object is represented by a set of nodes and elements, and the deformation of the object is described by the displacement of those nodes. The force-displacement relationship at those nodes is defined by the 'global stiffness matrix'. In the case of the linear-elastic model, the global stiffness matrix becomes a constant matrix.

Also, those nodes are classified based on the boundary condition. The displacement boundary condition is given when the node is fixed to the ground or to the user's finger (i.e., fixed node). On the other hand, the force boundary condition is applied when the node is free to move (i.e., free node). Taking this classification into consideration, the force-displacement relationship in the linear FEM model is described as follows:

$$\left(\begin{array}{c} F \\ \hline F_0 \end{array}\right) = \left(\begin{array}{c|c} K & K_0 \\ \hline K_1 & K_2 \end{array}\right)\left(\begin{array}{c} u \\ \hline u_0 \end{array}\right), \quad (1)$$

where (u) and (F) are the displacement and the force on free nodes, respectively. Also, (u_0) and (F_0) are the displacement and the force on fixed nodes, respectively. In the expression, we divide the global stiffness matrix into four parts (K, K_0, K_1, K_2). The displacement on free nodes is calculated as:

$$\left(\begin{array}{c} u \\ \hline u_0 \end{array}\right) = \left(\begin{array}{c|c} L & L_0 \\ \hline 0 & I \end{array}\right)\left(\begin{array}{c} F \\ \hline u_0 \end{array}\right), \quad (2)$$

where

$$L = K^{-1}, \quad (3)$$

$$L_0 = -K^{-1}K_0. \quad (4)$$

Furthermore, we assume that the displacements of all fixed nodes (u_0) are zero. Consequently, the displacements on free nodes are obtained as follows:

$$(u) = (L)(F). \quad (5)$$

In a special case where force $(\boldsymbol{F}_i)$ is applied on one node (i), the displacement on the node $(\boldsymbol{u}_i)$ is given as:

$$\left(\begin{array}{c} * \\ \hline \boldsymbol{u}_i \\ \hline * \end{array}\right) = \left(\begin{array}{c|c|c} & & \\ \hline & L_{ii} & \\ \hline & & \end{array}\right)\left(\begin{array}{c} 0 \\ \hline \boldsymbol{F}_i \\ \hline 0 \end{array}\right). \quad (6)$$

2.2. Calculation of Force

The force boundary condition equivalent to a displacement boundary condition is obtained by calculating the force that causes the same displacement given as the displacement boundary condition. In the case where the displacement boundary condition is given to one node, the equivalent force boundary condition on the node is obtained by the inverse transformation of Equation 6, as follows:

$$(\boldsymbol{F}_i) = (L_{ii})^{-1}(\boldsymbol{u}_i). \quad (7)$$

Generally, in the case where n nodes are involved, we need to calculate the inverse transformation of $3n \times 3n$ matrix.

2.3. Calculation of Deformation

The deformation of the whole object shape is obtained by calculating the displacements of all free nodes under the given force boundary condition (see Equation 5). In a special case where forces $(\boldsymbol{F}_i, \boldsymbol{F}_j, \boldsymbol{F}_k)$ affect three nodes (i, j, k) alone, the displacements on all free nodes are calculated as follows:

$$\left(\begin{array}{c} \\ u \\ \\ \end{array}\right) = \left(\begin{array}{c|c|c} & & \\ & L_iL_jL_k & \\ & & \end{array}\right)\left(\begin{array}{c} 0 \\ \hline \boldsymbol{F}_i \\ \boldsymbol{F}_j \\ \boldsymbol{F}_k \\ \hline 0 \end{array}\right). \quad (8)$$

3. Implementation

We implemented and tested the proposed method in a virtual environment with a force feedback device. To define the shape and the elasticity of the object, we employ a FEM model that consists of tetrahedral elements. Consequently, the surface of the object is represented by triangular patches, and the user's finger contacts with one of those triangular patches. To determine the contact point, we employ the god-object method [6], in which the 'god-object' indicates the contact point on the surface of the object. The displacement at the contact point $(\boldsymbol{u})$ is calculated as the disparity between the finger and the god-object positions.

From the displacement defined at one point of a triangular patch, we calculate the force on that point by interpolation. First, we calculate the force in the case where the displacement is made on each of the three nodes independently, by applying Equation 7, as follows:

$$\boldsymbol{f}_i = L_{ii}^{-1}\boldsymbol{u}, \quad \boldsymbol{f}_j = L_{jj}^{-1}\boldsymbol{u}, \quad \boldsymbol{f}_k = L_{kk}^{-1}\boldsymbol{u}. \quad (9)$$

Next, we derive the force on each node by multiplying the value of the area coordinate:

$$\boldsymbol{F}_i = \alpha_i \boldsymbol{f}_i, \quad \boldsymbol{F}_j = \alpha_j \boldsymbol{f}_j, \quad \boldsymbol{F}_k = \alpha_k \boldsymbol{f}_k. \tag{10}$$

The coefficients $(\alpha_i, \alpha_j, \alpha_k)$ are given as follows:

$$\begin{aligned} \alpha_i &= \triangle V_f V_j V_k / \triangle V_i V_j V_k \\ \alpha_j &= \triangle V_f V_k V_i / \triangle V_i V_j V_k \\ \alpha_k &= \triangle V_f V_i V_j / \triangle V_i V_j V_k, \end{aligned} \tag{11}$$

where V_i, V_j, V_k and V_f indicate the vertices of the triangular patch and the point where the finger contacts the triangular patch, respectively.

Also, we calculate the force that is fed back to the user ($\boldsymbol{F}$) as the reaction of the force given in the force boundary condition:

$$\boldsymbol{F} = -\boldsymbol{F}_i - \boldsymbol{F}_j - \boldsymbol{F}_k. \tag{12}$$

This approach guarantees the continuous change of the force vector on the edges and vertices of triangular patches. Also, it has been proven that this kind of force interpolation method contributes to the representation of haptically smooth surfaces[7]. Finally, the deformation of the whole object is given using Equation 8.

We implemented our prototype environment using a DOS/V PC (Pentium Pro 200MHz×2, Windows NT) and a PHANToM force feedback device (SensAble Technologies)[8]. Also, we used PHANToM basic i/o programming library to control the device. The library provides the function of executing haptic and visual processes concurrently. It also schedules the execution of the haptic process at every 1[ms].

The force and deformation calculations are independent of each other except that the latter refers to the force boundary condition obtained in the former. Also, the force fed back to the user was obtained in the force calculation (Equation 9-12), while the deformation calculation (Equation 8) was committed only to the visual representation of the deformation. By taking advantage of this independence, we assigned the force and deformation calculations in haptic and visual processes, respectively. These two precesses were executed asynchronously and the the update rate of the haptic process was kept 1 [kHz] irrelevant to the load of the visual process.

We implemented two objects of different complexity: a cube (30 × 30 × 30 [mm]) and an object shaped like a mouse (length: about 120 [mm]) (see Table 1). The shape of the cube was defined and divided into tetrahedral elements for FEM calculation algorithmically. The fixed boundary condition was applied to the bottom nodes. The shape of the mouse was captured using a 3D scanner (Model3030MM,Cyberware) and manually divided into tetrahedral elements. The model was fixed to the ground at four pads of the mouse.

Table 1: Specifications of models

	Cube	Mouse
All Nodes	125	1713
Surface Nodes	98	1175
Free Surface Nodes	73	1036
Surface Patches	192	2346
Tetrahedral Elements	320	6312
Young's Modulus [Pa]	10000	1000
Poisson's Ratio [-]	0.49	0.3

Table 2: Calculation time

Calculation	Cube	Mouse
Force [μs]	15	12
Displacement [μs]	158	4181
Draw [Hz]*	97	2

* from the cycle time of the visual process

Figure 1 shows the mesh structure of the cube and the example of operations on the cube. In the figure, the position of the god-object, as well as the finger tip position, is indicated by a small sphere. Also, Figure 2 shows the mesh structure of the mouse and examples of operations on the model.

In the representation of force sensation, the update ratio of the haptic process is essential[9]. We measured the computation time using the proposed method (Table 2) and found that the force computation time described above is short enough compared with the cycle time of the haptic process. Also, the deformation computation time is short enough compared with the cycle time of video refresh interval of the PC(16.6[ms] at 60[Hz]).

4. Conclusion

In this paper, we proposed a method of visually and haptically representing a deformable object by a linear FEM model. By implementing this method in a virtual environment, we confirmed that it is applicable to the real-time deformation of an object.

A significant feature of the proposed method is that the force and the deformation are obtained by direct calculation (i.e., not by iterative calculation). Because of this feature, the method is capable of both visually and haptically representing the quick reaction of the object that is not adequately obtained by iterative calculation models.

One of our future works is to implement picking and grasping operation on the object that is defined by the proposed model. For this purpose, we need to introduce the friction model on the object surface and the dynamics model of the whole body, as well as the two-fingered interface environment.