Representation of Force in Object Manipulation

Koichi Hirota*, **Masaki Hirayama***, **Atsuko Tanaka****, **Toyohisa Kaneko***

*Department of Information and Computer Sciences,
Toyohashi University of Technology
hirota@tutics.tut.ac.jp
**Central Research and Development Laboratory,
OMRON Corporation

Abstract

Manipulation of objects is a most fundamental operation in the virtual environment. In the computation of feedback force in object manipulation, we need to simulate the status of contact between the user's finger and the object. In this paper, we propose a method to simulate physical manipulation of objects with force feedback. The interaction force is computed based on the constraint that is caused by the object on the user's finger.

Key words: physical simulation, manipulation, virtual object, force feedback, virtual environment

1. Introduction

Manipulation of the object is a most fundamental operation in the virtual environment [1]. Various approaches have been proposed to implement manipulation. We can categorize these approaches into gesture based, geometric, and physical manipulations.

The gesture based manipulation is often employed for rough handling of objects using glove type devices. In this approach, the intention of the user to grasp or to release is conducted to the system by gestures (i.e. the posture of hands and fingers). However, it is difficult to apply this approach for the precise manipulation using fingers.

In the geometric approach, collision between fingers and the object is detected and the status of grasp and release is determined based on the status of geometric contact [2, 3]. In this approach, we need to artificially define criteria to interpret whether the object is grasped or not [4]. Also, we need to define the motion of the object according to the motion of fingers while the object is grasped. For example, the configuration of fingers may change while grasping an object, and the position and orientation of the object must be updated according to the change. However, in spite that these criteria and definitions have important effect on naturalness of manipulation, the discussion on the issue is not made enough.

From the aspect of physical phenomenon, we can not clearly define the status of grasp and release. When an object is manipulated by a hand, the force appears between the object and the hand or fingers, and the object moves based on the dynamics. We are just regarding that an object is grasped when the object moves with the hand. In the physical approach, this physical phenomenon is simulated. A merit of this approach is that we can simulate the slipping contact, which is frequently observed in real manipulation. There is an investigation on the physically based manipulation [5]. However, in the study, only objects of simple shapes such as sphere are dealt.

In both the geometric and physical approaches, it is essentially important to haptically feedback the constraint on the motion of fingers [6]. In the geometric approach, such constraint has effect to avoid the change of finger configuration. Also, in the physical approach, the feedback force informs the physical constraint directly to the user.

Representation of contact force is one of most basic issue in the study of haptic rendering[7]. For the computation of contact force for haptic representation, the idea of the god-object method[8, 9] is frequently used. In the computation of interaction in virtual environment, we usually define the model of user's hands or fingers. However, since the user's action is directly reflected on the model, it is inevitable that the model violate constraints defined in the environment. In the god-object method, we define another model of user's hands or fingers, and simulate the constrained behavior of the model in the environment.

In the implementation of the god object method for haptic interaction, we usually regard that the interaction occurs on a haptic interface point (HIP). In this case, the constrained motion of HIP is simulated by defining ideal haptic interface point (IHIP). For example, even when the HIP penetrates a surface, the IHIP is bound on the surface. Also, in this approach, the interaction force on the HIP is computed from the disparity between the HIP and IHIP.

In this paper, we propose a method to simulate physical manipulation of objects with force feedback. The interaction force is computed based on the constraint that is caused by the object on the user's finger. The behavior of the object is simulated by solving the equation of motion. To compute the distribution of force on the surface of fingers, we define plural HIPs on the surface. We expand the concept of the god-object method so that we can simulate physical constraints during the spatial motion of HIPs.

Table 1: State Transition

Before	After				
	Tet.	Sur.	Edg.	Ver.	Out.
Tet.	o	o	x	x	x
Sur.	o	o	o	x	o
Edg.	o	o	o	x	o
Ver.	o	o	o	o	o
Out.	x	o	x	x	o

(o: possible, x: impossible)

2. Simulation of Constraint

2.1 Object Model

The interaction force between the user's finger and the object is derived from physical constraints on the finger caused by the object. Such physical constraint changes according to the spatial position. For example, the air around the object cause little constraint force except the viscous drag, while the material of the object requires larger force for the finger to enter into or move in the volume. To represent the spatial difference of the material, we dived the space into tetrahedral cells, and defined the property of materials on each cell. We also examined the division of space by voxels or other polyhedra. However, in the case of voxel model, it is a serious problem that the boundary of the material becomes uneven. In the case of polyhedron, the increase in the number of surface polygons also increase the complexity of geometric computation. From the discussion, we applied the tetrahedron cell that has surface polygons of minimum number.

2.2 Computation Algorithm

In the simulation, the HIP moves in the tetrahedron model, and the constrained motion of the IHIP in the model is simulated. The simulation algorithm consists of both the computation of IHIP's motion in a cell and the transition of the constraint state among cells. The motion of IHIP in a cell is defined by a function (i.e., motion function). The function determines the position of IHIP based on the position of HIP, material of the space, the state of constraint (discussed below).

Since we assume that the material in each cell to be homogeneous, the position of IHIP computed by the function is valid so long as the IHIP is in an identical cell. However, if the IHIP step out of the cell, we need to put back the IHIP on the boundary of the cell, and to re-compute the IHIP position using the material of the neighboring cell. If the IHIP can not enter the neighboring cell (i.e., the IHIP do not move from the boundary), we need to explore the possibility that the IHIP moves constrained on the boundary surface. Similarly, it is possible that the IHIP moves constrained on edges and vertices. To systematically implement the change of the constraint state, we introduce the concept of state transition. The possible transition of state is summarized in Table 1.

Before describing the details of the state transition, we declare the motion function as follows:

$$\boldsymbol{q}' = move(\boldsymbol{q}, con, \boldsymbol{p}, \boldsymbol{p}_c, mat, mat_c, fric);$$

where $\boldsymbol{p}$ and $\boldsymbol{q}$ are the position of HIP and IHIP, respectively. The constraint state of IHIP is passed to the function through con, which is one of tetrahedron, surface, edge, vertex constraints (abbreviated to Tet, Sur, Edg, Ver, respectively). Also, mat indicate the material of the space in which the IHIP moves. In the case where the IHIP is constrained on the surface or edge, the IHIP tries to move not toward the HIP but toward a point closest to the HIP on the constraining plane or line, respectively. Such a goal of the IHIP is passed to the function using $\boldsymbol{p}_c$. Also, in this case, the IHIP travels on the boundary of two or more different materials. For example, when the user is tracing on the surface of an object, the IHIP moves in the volume of air on the boundary between the air and the object's material. The material that cause constraint on the IHIP is passed to the function through mat_c. Finally, $fric$ indicates the contact state, which is used to simulate friction (discussed later).

This function is generally used to test if IHIP can move under various constraint conditions. Using the motion function, the transition from each state is computed as follows:

(1) Tetrahedron Constraint

Firstly, we test if IHIP moves in the cell:

$$\boldsymbol{q}' = move(\boldsymbol{q}, Tet, \boldsymbol{p}, \boldsymbol{p}, mat_p, -, fric);$$

where, mat_p indicates the material of the belonging cell. If the resulting position $\boldsymbol{q}'$ is inside of the cell, the position of IHIP is moved to $\boldsymbol{q}'$, and the constraint state is kept unchanged.

If the resulting position $\boldsymbol{q}'$ step out of the cell, the cross point on the surface of the cell is computed, position of IHIP is moved to the cross point, and the constraint state is changed to the constraint on the surface in the present cell (i.e., the belonging cell).

(2) Surface Constraint

If the position of HIP is interior side of the surface plane of the belonging cell, we test if the IHIP can move into the cell:

$$\boldsymbol{q}' = move(\boldsymbol{q}, Tet, \boldsymbol{p}, \boldsymbol{p}, mat_p, -, fric);$$

where mat_p is the material of the cell. If IHIP can move, the position of IHIP is kept unchanged and the constraint state is changed to the tetrahedron constraint in the present cell.

Similarly, if the position of HIP is outer side of the surface plane of the belonging cell, we test if the IHIP can enter into the neighboring cell. If no cell is found as a neighbor, the position of IHIP is kept unchanged and the constraint state is changed to outside constraint. Otherwise, the motion of IHIP into the neighboring cell is tested:

$$\boldsymbol{q}' = move(\boldsymbol{q}, Tet, \boldsymbol{p}, \boldsymbol{p}, mat_n, -, fric);$$

where mat_n indicate the material of the neighboring cell. If IHIP can move, the position of IHIP is kept unchanged and the constraint state is changed to the tetrahedral constraint in the neighboring cell.

If both of the above test fail, we test the motion of IHIP on the surface plane:

$$\boldsymbol{q}' = move(\boldsymbol{q}, Sur, \boldsymbol{p}, \boldsymbol{p}_f, mat_p, mat_n, fric);$$

where $\boldsymbol{p}_f$ is the foot of the perpendicular from HIP to the constraining surface plane. Also, mat_p and mat_n indicate the material of the belonging and neighboring cells.

If the resulting $\boldsymbol{q}'$ is inside of the triangle surface area, the position of IHIP is moved to $\boldsymbol{q}'$ and the constraint state is kept unchanged. If $\boldsymbol{q}'$ step out of the triangle area, the cross point on the boundary edge of the area is computed, position of IHIP is moved to the cross point, and the constraint state is changed to the constraint on the edge in the present cell.

(3) Edge Constraint

Firstly, we test if IHIP can move toward HIP. For this purpose, we find the cell in the orientation of HIP from cells sharing the edge. If no cell is found, the position of IHIP is kept unchanged and the constraint state is changed to outside constraint. Otherwise (i.e., a cell in the orientation of HIP is found), the motion of IHIP into the cell is tested:

$$\boldsymbol{q}' = move(\boldsymbol{q}, Tet, \boldsymbol{p}, \boldsymbol{p}, mat_h, -, fric);$$

where mat_h indicates the material of the cell found above. If IHIP can move, the position of IHIP is kept unchanged and the constraint state is changed to the tetrahedral constraint in the found cell.

If the test fail, we test the possibility to move into other cells sharing the edge. In this case, the IHIP moves along a boundary surface of cells sharing the edge. On each of the boundary surfaces of each cell, we compute $\boldsymbol{p}_i$ as the foot of the perpendicular from the HIP on the plane. If the foot is on a half-plane bounded by the edge, we regard the $\boldsymbol{p}_i$ as a valid goal of IHIP, and test if the IHIP can move on the surface:

$$\boldsymbol{q}' = move(\boldsymbol{q}, Sur, \boldsymbol{p}, \boldsymbol{p}_i, mat_i, mat_{ni}, fric);$$

where mat_i and mat_{ni} indicate the material of remaking cell and the neighboring cell on the other side of the boundary surface, respectively. Among all the cases where the IHIP can move, we apply the case that maximize the distance $|\boldsymbol{p}_i - \boldsymbol{q}|$ (i.e., the case where the orientation of motion is closest to the orientation of HIP) as the result. In this case, the position of IHIP is kept unchanged and the constraint state is changed to the surface constraint on the boundary surface in the applied cell.

Otherwise (i.e., the IHIP does not move in all of the cases), the motion of IHIP on the edge is tested:

$$\boldsymbol{q}' = move(\boldsymbol{q}, Edg, \boldsymbol{p}, \boldsymbol{p}_f, mat_p, mat_h, fric);$$

where mat_p indicates the material of the present belonging cell. If the resulting $\boldsymbol{q}'$ is inside of the edge (i.e., $\boldsymbol{q}'$ do not step out of the end point), the position of IHIP is moved to $\boldsymbol{q}'$ and the constraint state is kept unchanged. If $\boldsymbol{q}'$ step out of an end point, position of IHIP is moved to the end point, and the constraint state is changed to the constraint on the vertex in the belonging cell.

(4) Vertex Constraint

In the beginning, we test if IHIP can move toward HIP. For this purpose, we find the cell in the orientation of HIP from cells sharing the vertex. If no cell is found in the orientation, the position of IHIP is kept unchanged and the constraint state is changed to outside constraint. Otherwise, the motion of IHIP into the found cell is tested:

$$\boldsymbol{q}' = move(\boldsymbol{q}, Tet, \boldsymbol{p}, \boldsymbol{p}, mat_h, -, fric);$$

where mat_h indicates the material of the found cell. If IHIP can move, the position of IHIP is kept unchanged and the constraint state is changed to the tetrahedral constraint in the cell.

If the test fail, we test the possibility to move into other cells sharing the vertex. In this case, the IHIP moves along a boundary surface or an edge of cells sharing the vertex.

On each of the boundary surfaces of each cell, we compute the foot of the perpendicular from the HIP on the plane ($\boldsymbol{p}_i$). If the foot is on a half-plane bounded by two edges of the triangle surface sharing the vertex, we regard $\boldsymbol{p}_i$ as a valid goal, and test if the IHIP can move on the surface:

$$\boldsymbol{q}' = move(\boldsymbol{q}, Sur, \boldsymbol{p}, \boldsymbol{p}_i, mat_i, mat_{ni}, fric);$$

where mat_i and mat_{ni} indicate the material of remaking cell and the neighboring cell on the other side of the boundary surface. Also, on each of the boundary edges of each cell, we compute the foot of the perpendicular from the HIP on the edge line ($\boldsymbol{p}_j$). If the foot is on a half-line bounded by the vertex, we regard $\boldsymbol{p}_i$ as a valid goal, and test if the IHIP can move on the edge:

$$\boldsymbol{q}' = move(\boldsymbol{q}, Edg, \boldsymbol{p}, \boldsymbol{p}_{fj}, mat_j, mat_h, fric);$$

where mat_j indicate the material of remaking cell. Among all the cases where the IHIP can move, we apply the case that maximize the distance $|\boldsymbol{p}_i - \boldsymbol{q}|$ or $|\boldsymbol{p}_j - \boldsymbol{q}|$ (i.e., the case where the orientation of motion is closest to the orientation of HIP) as the result. In this case, the position of IHIP is kept unchanged and the constraint state is changed to the surface or edge constraint on the boundary surface or edge in the applied cell.

Otherwise (i.e., the IHIP does not move in all of the cases), the position of IHIP and the constraint state is kept unchanged.

(5) Outside

We test intersection between the trajectory of the HIP and all of surface triangle planes of the model. The trajectory is defined by linearly connecting the past and present positions of the HIP. If a cross point is found, the position of IHIP is moved to the cross point and the constraint state is changed to surface constraint in the cell that owns the surface triangle. Otherwise, the IHIP is moved to the HIP and the constraint state is kept unchanged.

2.3 Definition of Motion Function

As is described above, the motion function defines the rule of the IHIP's motion. By changing the definition of the function, various constraints on the IHIP is represented. A most simple geometric constraint is represented by the definition as follows:

$$\boldsymbol{q}' = \begin{cases} \text{if } mat = 0: \\ \quad \boldsymbol{p}_c \\ \text{if } mat = 1: \\ \quad \boldsymbol{q} \end{cases} \tag{1}$$

According to the function, the IHIP moves freely in material 0, while it never moves in material 1. Since the test to enter into material 1 always fails, IHIP is constrained on the boundary, although it can move without friction on the boundary.

If we define the force on the IHIP, it is possible to represent physical constraint on the IHIP. Suppose the force on the IHIP is defined by the following equation:

$$\boldsymbol{f} = k(\boldsymbol{q} - \boldsymbol{p}). \tag{2}$$

where k is a coefficient that is understood as the stiffness of the finger. As is described above, it it a common approach in previous studies to compute the interaction force proportionally to the disparity between the HIP and the IHIP. Based on this relationship, we can represent the frictional force using the motion function as follows:

$$\boldsymbol{q}' = \begin{cases} \text{if } mat = 0: \\ \quad \boldsymbol{p}' \\ \text{if } mat = 1: \\ \quad \begin{cases} \text{if } k|\boldsymbol{p}_c - \boldsymbol{q}| < F: \\ \quad \boldsymbol{q} \\ \text{if } k|\boldsymbol{p}_c - \boldsymbol{q}| \geq F: \\ \quad \boldsymbol{p}_c + \frac{F}{k}\frac{(\boldsymbol{q}-\boldsymbol{p}_c)}{|\boldsymbol{q}-\boldsymbol{p}_c|} \end{cases} \end{cases} \tag{3}$$

In material 1, the position of IHIP is kept unchanged if the force affecting on IHIP is smaller than a given threshold (F). Otherwise, the IHIP approaches to HIP until the force becomes equal to the threshold. Consequently, a constant frictional force is fed-back when IHIP is moving in the material.

2.4 Computation Process

As is obvious from the state transition processes, the motion of the IHIP from a cell to another neighboring cell requires at least three transitions of state (e.g., from tetrahedron constraint in a cell to surface constraint on the boundary, to tetrahedron constraint of the neighboring cell on the boundary, and to the volume constraint in the neighboring cell). To enable plural transition of state in one cycle, the state transition is repeated for each IHIP until both the constraint state and the position become kept unchanged. At the end of each cycle, the flag on the motion of each IHIP is updated. The flag indicates whether the IHIP has been moved in the cycle or not. The flag is passed to the motion function through *fric* parameter.

Also, in the implementation discussed below, to reduce the computation time, the normal and orientation vectors of all surfaces and edges are computed in advance. Also, the information of neighboring cells and the table of cells sharing vertices and edges are initially created.

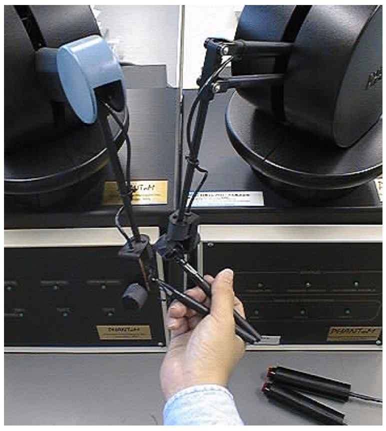
Fig.1: Haptic Interface Device

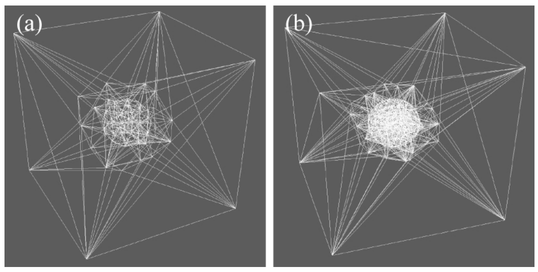

Fig.2: Tetrahedral Elements in the Model.

2.5 Implementation

We implemented the algorithm in a virtual environment. We used a PC (Pentium Pro 200MHz×2, Windows NT) for all of the computation, and two PHANToM force feedback devices (SensAble Technologies)[10] for haptic feedback (Figure 1). Also, we used the GHOST 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].

Figure 2 shows an example of the structure of of tetrahedron cells, where inside and outside spaces of a spherical object is divided into cells. The surface of the spherical object consists of 80 and 320 triangle polygons, respectively. The material of cells inside of the sphere is 1 and and outside 0. We applied the motion functions defined by Equation 1 and 3 on these two models. Figure 3 shows the result of the constraint force computed by the proposed algorithm, where fifty HIPs are placed on a line

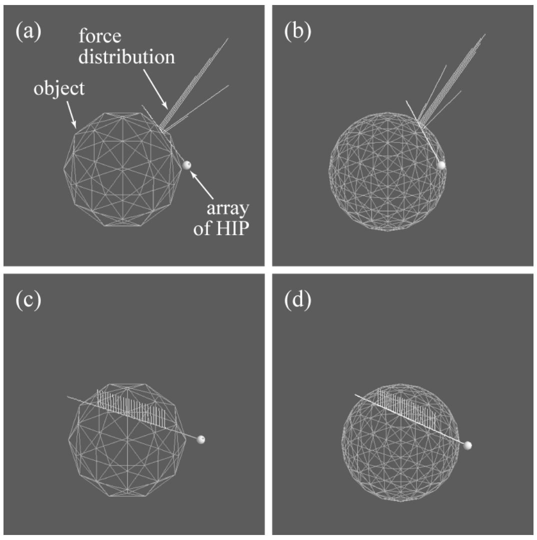

Fig.3: Representation of Constraint Force.

Table 2: Computation Time of State Transition

Before	After				
	Tet.	Sur.	Edg.	Ver.	Out.
Tet.	2.7	10.4	—	—	—
Sur.	0.6	2.6	7.2	—	0.5
Edg.	0.8	45.7	45.6	45.2	1.0
Ver.	4.0	207.7	209.3	194.4	6.2
Out.	—	15.2	—	—	20.9

(unit:μs)

with the interval of 1mm, and the simulation of constraint is executed on each HIP. In this example, the coefficient k is 0.1[N/mm]. Also, the parameter F in Equation 3 is 0.1[N].

We also measured the computation time of the proposed algorithm. Table 2 shows the computation time of all state transition. The computation time for the transition from the edge and the vertex state varies according to the number of tetrahedron cells sharing the edge and vertex, respectively. In the measurement, we supposed that an edge is shared by twelve cells, and that a vertex is shared by twenty cells. Since the computation time increases almost linearly to the number of cells, we can estimate the computation time even in the case where larger number of cells are concerned. Also, the computation time for the transition from outside state depends on the number of surface polygons in the model. In the measurement, the number of the surface triangle is supposed to be twelve, which is equal to the number of surface when the model has a boundary of rectangular box. As is observed from the result, the transition from nodes and edges requires relatively large time.

It is an important feature of the proposed algorithm that the computation time of state transition is independent of the number of cells and nodes in the whole model. The computation time depends on the number of boundaries of the cell that the IHIP passes in a cycle. In this sense, the computation time increases proportionally to the velocity of the user's finger. In the feedback of force, it is known that the update rate of force have serious effect on the sensation of force presented to the user[11]. In case of PHANToM, it is recommended to keep the haptic update rate to at least 1[kHz]. One of the worst case is edge-node-edge transition while tracing a concave shape, and the computation time required for the transition is estimated to be about 300 [μs]. In a cycle time of 1[ms], we can simulate the motion of one IHIP over three boundaries.

3. Object Manipulation

3.1 Representation of Friction

The friction on the surface has important effect on the manipulation of objects. One of the most simple approximation of friction is defined by a model in which the coefficients of static and kinetic frictions are defined independent of the velocity. To simulate the gripping force, we define the friction using the model. The constraint on the IHIP's motion caused by the friction on the surface is defined in a motion function as follows:

$$\boldsymbol{q}' = \left\{ \begin{array}{l} \text{if } con = Tet: \\ \quad \boldsymbol{p}_c \\ \text{if } con \neq Tet \text{ and } mat_c = 1: \\ \quad \left\{ \begin{array}{l} \text{if } |\boldsymbol{f}_n| < \mu|\boldsymbol{f}_t|: \\ \quad \boldsymbol{q} \\ \text{if } |\boldsymbol{f}_n| \geq \mu|\boldsymbol{f}_t|: \\ \quad \boldsymbol{p}_c + \mu|\boldsymbol{p}-\boldsymbol{p}_c|\frac{\boldsymbol{q}-\boldsymbol{p}_c}{|\boldsymbol{q}-\boldsymbol{p}_c|} \\ \left(\begin{array}{ll} \text{if } fric = 0: & \mu = \mu_s \\ \text{else}: & \mu = \mu_k \end{array} \right) \end{array} \right. \end{array} \right. \tag{4}$$

where $\boldsymbol{f}_n$ and $\boldsymbol{f}_t$ are normal and tangential components of force, and they are computed as follows:

$$\boldsymbol{f}_n = k(\boldsymbol{p}_c - \boldsymbol{p}), \quad \boldsymbol{f}_t = k(\boldsymbol{p}_c - \boldsymbol{q}). \tag{5}$$

As is described above, $fric$ indicates whether the IHIP is moving in the model ($fric = 1$) or not ($fric = 0$). Also, μ_s and μ_k are coefficients of static and kinetic frictions.

Figure 4 shows a result of representing friction using Equation 4. In the figure, the tangent/normal ratio of the force affecting on a IHIP is plotted. In the passage of time, the state of friction changes form 'grip' to constant 'slip', and consequently, the ratio is changing from almost 1.0 ($= \mu_s$) to 0.5 ($= \mu_k$). Also, we can observe a kind of stick-slip oscillation in the transition.

3.2 Computation of Distributed Force

It is difficult to compute the distribution of force on the contact area between the finger and the object in real-time. For the precise computation of the distributed force, we need to define deformable model of the finger and to simulate the deformation of the finger. On the other hand,

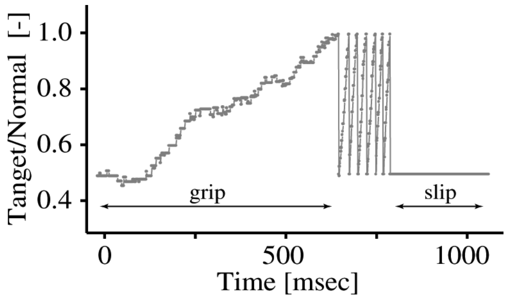

Fig.4: Representation of Friction

the distribution of force is essentially important in manipulation. Because, in case of point contact model, we can not affect torque around the normal axis of the contact surface. To avoid the problem, we approximately compute the distribution of force by defining multiple HIPs on the surface of finger. On each HIP, the simulation of constraint is performed independently.

3.3 Motion of Object

Based on the force obtained in the simulation of constraint, the motion of an object is computed using the equations of motion. In our model, we define the shape of a object in a local coordinate system (i.e., object coordinate system), and describe the relationship between object and world coordinate systems by a transformation matrix. We also define a gravity-center coordinate system, in which the equations of motion of the object is described.

Firstly, the force and torque around the gravity center of the object is computed. Next, the acceleration and angular acceleration of the object is obtained using Newton' and Eular's equations of motion. The effect of gravity and the effect of viscosity in translation and rotation are also considered in this computation. The velocity, angular velocity, translation and rotation angle are obtained by integrating the acceleration during each cycle time. Finally, the transformation matrix is updated.

In advance to the simulation of motion (i.e., in the initialization process), the position of gravity center, the inertia tensor, and the mass of the object is computed from the geometry and the material of the model. Also, the viscosity in translation and rotation is artificially determined.

3.4 Implementation

We implemented the algorithms of friction and motion in the virtual environment described above. In the experiment, eleven HIPs are defined on each of the fingertip (i.e., around the tip of stylus). The acceleration of gravity is 0.98 [m/s^2] downward, and the density of the material 0 and 1 are 0 and 500 [kg/m^3], respectively. Figure 5 shows an example of force distribution on the HIPs while holding a rectangular object. In the figure, the gravity center of the object is at the center of the rectangle

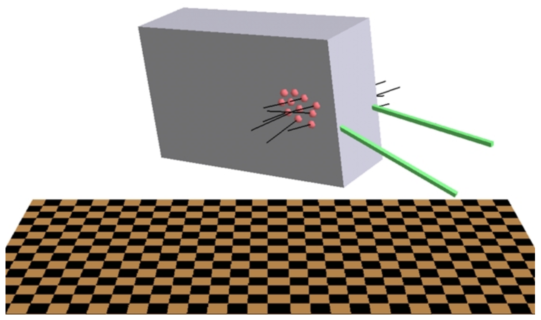
Fig.5: Manipulation of object using friction

shape, while the finger is holding the object near the edge. The twisting moment to support the object horizontally and the force to cancel the gravity are appearing on the contact points as the tangent component of force vectors. Although we have twenty two HIPs in the simulation, the computation is completed almost within 1[ms], probably because it scarcely occurs that the HIP is constrained on nodes in the interaction with convex shapes.

Figure 6 shows examples of manipulating objects of more complex shapes. The model consists of 1064 and 1534 cells inside and outside of the shape, respective, and the surface consists of 232. triangle polygons. Also, in this case, the computation is executed almost within a cycle time of 1[ms].

4. Conclusion

In this paper, we proposed a method to physically simulate the manipulating operation in the virtual environment. The interaction force is computed by simulating the constraint on HIPs caused by the object. By introducing physical laws in the simulation of IHIP's motion, the physically based force such as friction is represented.

One of the future work is to apply the proposed algorithm to feedback tactile sensations on the surface of fingertips. The proposed algorithm is capable of computing slip and grip state on each HIP, it is expected that the information is effectively fed-back to the user if tactile devices are integrate to the system.

Also we are interested in the manipulating operation using a whole hand. Figure 7 shows our current prototype of manipulation environment, where the model of a hand with three fingers are implemented using a spatial position sensor (Fastrak, 3SPACE). For the computation of constraints on whole hand, we need to represent the whole shape of a hand by large number of HIPs.

References

1. Hirose, M.: Virtual Reality; Sangyo-tosho (1993).

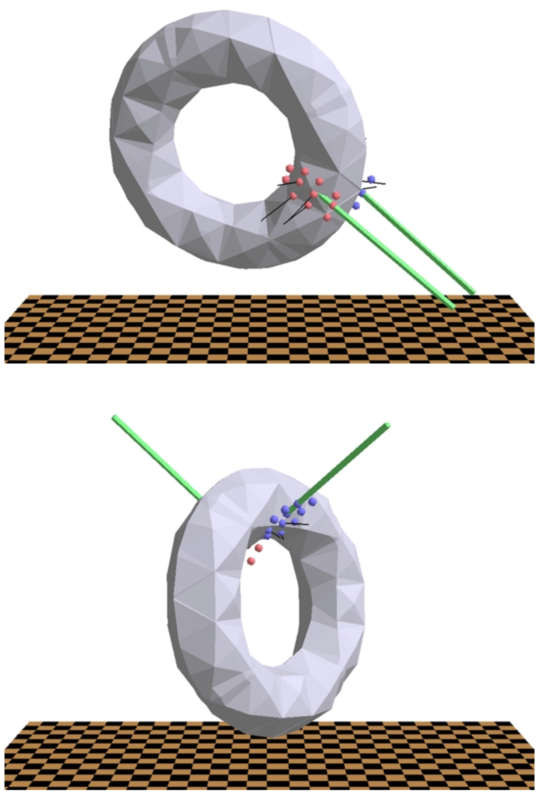

Fig.6: Manipulation with force feedback

2. Kitamura, Y., Smith, A., Takemura, H. and Kishino, F.: A Real-Time Algorithm for Accurate Collision Detection for Deformable Polyhedral Objects; *PRESENCE*, Vol. 7, No. 1, pp. 36–52 (1998).
3. Kitamura, Y., Higashi, T., Masaki, T., Kishino, F., Virtual Chopsticks: Object Manipulation using Multiple Exact Interactions; *Proc. VR99*, pp.198-203 (1999)
4. Funahashi, K., Yasuda, T., Yokoi, S., Toriwaki, J., A Model for Manipulation of Objects with Virtual Hand in 3-D Virtual Space; *Trans. IEICE*, Vol. J81-D-II, No. 5, pp.822-831 (1998)
5. Yoshikawa, T., Ueda, H., Display of 3-Dimensional Operating Feel of Dynamic Virtual Objects with Frictional Surface; *Progress in Human Interface*, Vol.2, pp.49-54 (1995)
6. G. Burdea: *Force & Touch Feedback for Virtual Reality*, A Wiley-Inter-Science Publication (1996).
7. Salisbury, K., Brock, D., Massie, T., Swarup, N., Zilles, C.: Haptic Rendering: Programing Touch Interaction With Virtual Objects; *Proc. Symp. Interactive 3D Graphics*, pp.123-130 (1995).
8. Zilles, C. B. and Salisbury, J. K.: A Constraint-based God-object Method For Haptic Display; *IROS'95*, pp. 145–151 (1995).

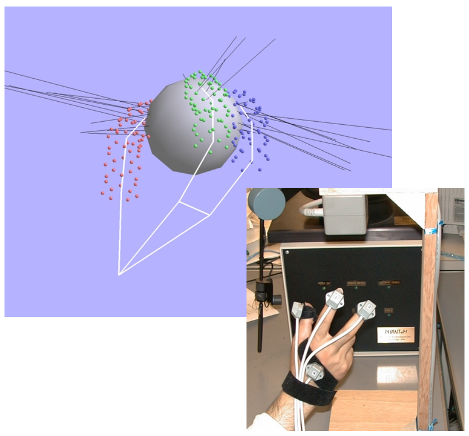

Fig.7: Manipulation using a Hand

9. Ho, C., Basdogan, C. and Srinivasan, M. A.: Haptic Rendering: Point- and Ray-Based Interactions; *PUG'97* (1997).
10. Massie, T. H.: Initial Haptic Explorations with the Phantom: Virtual Touch Through Point Interaction; *Master's thesis*, M.I.T. (1996).
11. Shimoga, K.: A study of perceptual feedback issues in dextrous tele-manipulation: Part I. Finger force feedback; *VRAIS'93*, pp. 263 – 270 (1993).