EUROGRAPHICS 2004 / M.-P. Cani and M. Slater
(Guest Editors)

Volume 23 (2004), Number 3

Real-time Rigid Body Simulation for Haptic Interactions Based on Contact Volume of Polygonal Objects

S. Hasegawa and M. Sato

Tokyo Institute of Technology, Tokyo, Japan

Abstract

This paper proposes a new method for real-time rigid body simulations for haptic interactions based on a penalty method regarding contact volume. Analytical methods for calculation of contact forces require too much time to maintain fast update rates for haptic controls. In addition, they prohibit direct connection of haptic interfaces.
Penalty methods, which employ spring-damper models for calculation of contact forces, offer a very rapid rate of iterations. In addition, they permit direct connection of haptic interfaces. Penalty methods are good for haptic interactions. However, previous penalty methods do not regard distribution of contact forces over the contact area. For that reason, they can't calculate normal and friction forces on face-face contacts correctly.
We propose a distributed spring-damper model on a contact area to solve these problems. We analyze three-dimensional geometries of the intersecting portion on the polyhedral objects. Then, we integrate forces and torques of distributed spring-damper models.
We implement a proposed simulator and compare it with a point-based penalty method and constraint method. The comparison shows that the proposed simulator improves accuracy of the simulation of face-face contact and friction forces and the simulation speed. In addition, we attach a six degree-of-freedom (6-DOF) haptic interface to the simulator. Users can feel 6-DOF force feedback and input 6-DOF motions.

Categories and Subject Descriptors (according to ACM CCS): I.6.5 [Model Development]: Modeling methodologies, H.5.2 [User Interfaces]: Haptic I/O, I.3.6 [Methodology and Techniques]: Interaction techniques

1. Introduction

Recent progress of computational power encourages the use of virtual environments. In addition to visual and audio sensations, sensation of forces acting on our hands during touch enhances the intuitiveness of interactions. Physics-based modeling, which simulates motions of virtual objects based on physics laws, creates realistic motion of virtual objects. The combination of these techniques realizes haptic interaction with realistic virtual environments.

We create a simulator and a framework for various realistic virtual environments through haptic interaction. Such virtual environments are anticipated to facilitate use of many applications in fields of entertainment, art, training, evaluation, and design.

1.1. Subjects to solve

We realize real-time rigid body motion simulation for haptic interaction by solving the following problems.

Update speed and computation time constancy. High frequency updates (300 Hz - 1 kHz or higher) are necessary to control force displays [LB95]. In addition, stability of computation time for each update is required for haptic interaction because excess computation time over the control periods leads to unstable force displays. This does not constitute a problem for offline purposes.

Direct connection of haptic interfaces. Conventional physics simulators do not allow direct connection of haptic interfaces. This limits sensation of presentation and system variation. (section 2.2 and 3.2)

Distribution of friction force and torque. Friction force is generated on all contact parts of objects. However, conven-

tional real-time motion simulators do not address the distribution of friction forces.

2. Previous work

Most of the early works come from combination of simulators for graphics and haptic interfaces. Some works of haptic rendering mention physical motion of virtual objects. We summarize them below.

2.1. Contact force estimation

Main problem in rigid body motion simulations is to find contact force. Once forces, which act on objects are found, motions of objects can be determined from numerical solutions of the equation of motion. Gravity forces, spring forces and dynamic friction forces are determined by fields, positions and velocities of objects. Such forces are easy to calculate. On the other hand, it is not very easy to calculate forces from partial constraints on relative positions or velocities such as contact forces and static friction forces.

Analytical method. Baraff *et al.* [Bar89] [Bar94] and Sauer and Schoemer [SS98] propose to solve contact forces from conditions of constraints and equations of motion. Mirtich and Canny [MC95] propose to calculate normal forces from the impact force of two objects. Their method finds time of impacts and solves impacts of two objects paired sequentially. These methods solve constraints completely and motions of objects are very crisp.

However, computation time for these methods changes sharply depending on the complexity of the simulated scene. The former requires computation time of $o(n^3)$ for number of contacts n and the latter must execute many iterations if many impacts occur in a short term. Therefore, it is difficult to maintain haptic update rates for these methods.

In addition, analytical methods require equations of motion to compute contact forces. They can't treat objects whose dynamic properties are not predefined, such as haptic pointers. For such cases, they require special models such as virtual couplings.

Penalty methods. Moore and Wilhelms[MW88], McKenna and Zeltzer [MZ90], Keller *et al.* [KSZ93] employed penalty methods for computation of contact forces. Penalty methods give forces that are proportional to violations of constraints to solve the violations; usually, these forces are calculated by spring-damper model. These methods require small simulation time steps to make crisp motions. Calculation of contact forces by penalty methods requires only computation time of $O(n)$ for the number of the contacts n. In addition, penalty methods calculate contact forces only from positions and velocities, which do not relate to the equation of motion. Hence, they can treat both haptic pointers and virtual objects directly.

However, there is a problem. Conventional real-time penalty-based simulators consider contacts between objects occur at points. They ignore contacts on large areas, such as contacts of a cube on a floor. They do not calculate normal and friction forces correctly in face-face contacts.

Terzopoulos *et al.*[TPBF87] and Snyder *et al.*[SWF*93] obtain normal force from the amount of penetration of many sampling points on surfaces of objects. Hippmann [Hip03] analyzes contact geometries of polyhedral objects to find contact forces. His contact analysis is based on polygon elements. These methods require a lot of computation and can't work in real-time.

2.2. Haptic rendering with 6-DOF

McNeely *et al.*[MPT99] models the haptic pointer in a point cloud for six-degree-of-freedom (6-DOF) haptic rendering. Kim *et al.* [KOLM03] models the haptic pointer in many convex polyhedra. They find normal force from the weighted average of penetration depths of each convex polyhedron for 6-DOF haptic rendering of complex shapes. The methods above do not address the presentation method for friction forces.

Chang and Colgate [CC97] calculate presentation forces from a virtual coupling. Their method produces a heavy sensation of manipulation because their haptic pointer model has mass.

Adachi *et al.* [AKO95], Hasegawa *et al.* [HIKS99] and Hollis *et al.* [BH00] represented an interfering structure of virtual objects and haptic pointers in a simple shape model (*i.e.* intermediate representations) and determined presentation forces from penetrations of haptic pointers to intermediate representations. Their methods employ slow update simulation. There is some delay between the input through the haptic interface and the motion of the virtual object. Therefore, the user feels virtual objects are heavier than real ones.

3. Proposal

We propose to determine contact forces from three-dimensional geometries of contacts between virtual objects. The proposal is a kind of penalty method. However, it considers spring-damper models distributing on a contact area generate the contact force. The followings are advantages of the proposed method.

3.1. Short and stable computation time for one update

As we mentioned in section 2.1, analytical methods require much computation time like $o(n^3)$ for the number of contacts n. The computation time increases sharply when the number of contacts increases. In contrast, the proposed method requires $o(n)$ for the number of contacts n. Thus, the computation time is more stable than that of analytical method.

3.2. Connection to haptic interfaces

Typical haptic renderings present forces that are proportional to penetrations of haptic pointers against virtual objects to present shapes of virtual objects. Contact forces are proportional to mutual penetrations of virtual objects in the proposed method. Therefore, any virtual object with various shapes can be used as a haptic pointer by presenting contact forces to the haptic interface and setting the position and velocity of the haptic interface to the virtual object. Moreover, the proposed method calculates 6-DOF normal and friction forces. This realizes 6-DOF haptic rendering including friction forces.

3.3. Calculation of contact force

The proposed method can also calculate correct contact forces including the case of face-face contacts because it integrates contact forces from contact areas. This ability was difficult for previous point-based penalty methods.

This section indicates drawbacks of the conventional point-based methods and advantages of the proposed contact-area-based method for both normal and friction forces.

Normal forces. Normal forces, which satisfy the non-penetrating constraint of any point on contact surfaces, work on the contacts.

Previous point-based methods address only the most penetrating point or vertices and ridges as contact points. Hence, contact points sometimes appear and disappear; consequently, normal forces change non-continuously. As a result, motions of objects in the previous methods do not converge in some cases.

To illustrate the problem, we consider a simulation of a cube on a floor in the previous method. It considers the most penetrating point as the contact point as in the method presented by Kim *et al.* Contact points are generated and disappear because the most penetrating point changes according to the rotating motion of the cube (Figure 3.3).

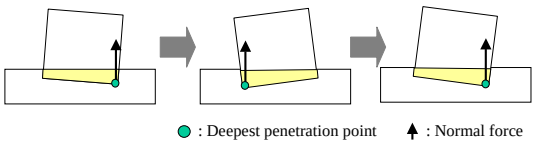

Figure 1: *Problem on normal forces*

The problem indicated above is peculiar to penalty methods. In analytical methods [Bar89][Bar94][SS98], normal forces are solved to satisfy constraints and the problem does not become a hindrance.

The proposed method puts a distributed spring-damper model on the entire area of the contact (Figure 2). This creates a continuous change of normal forces and their application points; it also enables convergence of the motion with proper spring and damper coefficients. We can find proper spring and damper coefficient from mass of the objects.

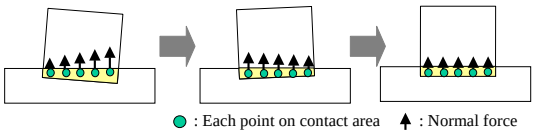

Figure 2: *Solution on normal forces*

Friction forces. The dynamic friction force and the maximum static friction force on a small area of contact surface are proportional to the normal force acting on the small area. The friction forces acting on the entire contact area are their sum.

Previous methods, which consider the most penetrating point or vertices and ridges as the contact points, can't calculate exact friction forces. For example, if friction force acts on one point, there is no friction torque for rotation. Figure 3 is a top view of a cube slipping on a floor in a simulator in which friction forces act on a point.

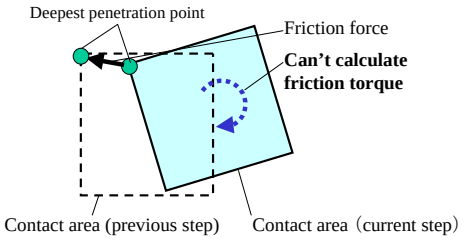

Figure 3: *Problem of friction forces: Top view of a cube slipping on a floor.*

On the other hand, the proposed method considers that the Coulomb friction models are distributed on entire contact area and friction forces are generated from them (Figure 4). Therefore, the proposed method can calculate both friction forces and torques.

4. Implementation of a proposed rigid body motion simulator

Motions of rigid bodies are represented by an equation of motion (1), (2).

$$m\dot{\boldsymbol{v}} = \boldsymbol{F}(t) \tag{1}$$

$$I\dot{\boldsymbol{\omega}} = \boldsymbol{N}(t) \tag{2}$$

Here, $\boldsymbol{v}$ and $\boldsymbol{\omega}$ represent the velocity and angular velocity of the rigid body. $\boldsymbol{F}(t)$ and $\boldsymbol{N}(t)$ represent the force and torque acting on the rigid body.

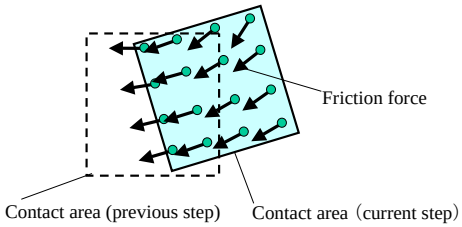

Figure 4: *Solution for friction forces: Top view of a cube slipping on a floor.*

We can calculate positions and orientations of rigid bodies from numerical integration of the equation of motion. We analyze contact states and find contact geometries to calculate contact forces. The proposed simulator treats rigid bodies whose shapes are represented by polyhedra. For efficiency of analysis on the contact status of polyhedra, we decompose polyhedra into convex polyhedra preliminarily.

We process the following procedures to find contact forces (*i.e.* normal and friction forces) for each combination of two convex polyhedra.

1. Contact detection. If contacts are present, proceed to 2.
2. Find the contact normal and the contact plane.
3. Find the geometry of the contact volume.
4. Find the normal and friction force from the geometry.

The followings are the explanation of each step.

4.1. Contact detection

We represent shapes of virtual objects as unions of convex polyhedra. We use GJK algorithm [GJK88] for the contact detection because it is fast and uses a simple data structure. GJK algorithm determines the closest point pair or a point on the contact volume for a given pair of convex shapes.

4.2. Contact plane

The proposed simulator processes the following procedure for each convex pair to decide contact planes (the direction and application point of normal forces).

1. Separate the contact pair of convexes in the direction of contact normal, which was calculated by a previous iteration.
2. Find the closest point pair to find the new contact normal.
3. If convexes still contact each other, double the distance of separation, then repeat from (1).
4. Return the position of the pair of convexes before (1). Then, find the contact plane which includes the middle point of the closest point pair and whose normal is equal to the contact normal.

We put spring and damper models for normal and friction forces on this contact plane.

4.3. Analysis of the contact geometry

Muller and Preparata [MF78] proposed an algorithm, which finds a common part of two convex polyhedra for a given common point of two convex polyhedra. Mullar's algorithm uses geometric dual transformation (Figure 5). This transformation converts a plane of $ax+by+cz=1$ into a point of (a,b,c), and a point of (a,b,c) into a plane of $ax+by+cz=1$.

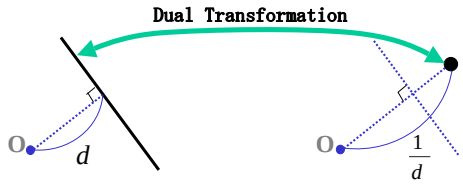

Figure 5: *Dual transformation*

The following is Muller's algorithm (see Figure 6).

1. Considering a given common point as the origin, convert planes of convex polyhedra into a vertex by dual transformation.
2. Find minimum convex polyhedra that include all vertices (*i.e.* convex hull).
3. Convert the convex polyhedra found in 2. by dual transformation to determine the geometry of the common part.

This algorithm yields the geometry of contact volume (vertices and faces). The proposed simulator employs QuickHull[BDH96] to implement Muller's algorithm. We use a common point found by the GJK algorithm for procedure 3.

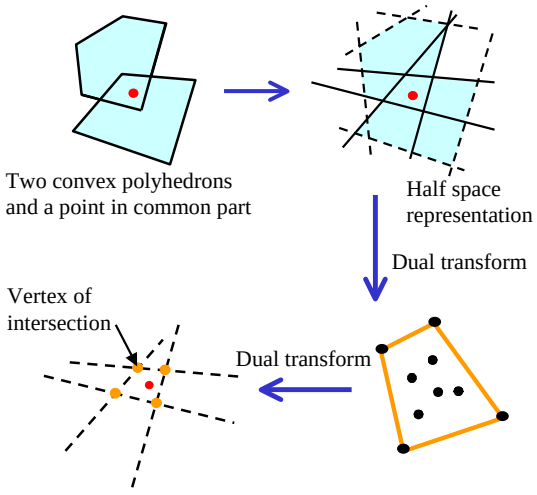

Figure 6: *Common part of two convex polyhedra*

4.4. Preliminary contact force calculation

Any force acting on any point on a rigid body can be represented as a force acting on the origin and a torque. The proposed simulator integrates normal and friction forces divided into forces working on the origin and torques to find the total amount of force and torque working on the rigid body. A force of $\boldsymbol{f}$ working on any point of $\boldsymbol{p}$ on the contact area generates a force of $\boldsymbol{f}_p$ working on the origin and a torque of $\boldsymbol{m}_p$.

$$\boldsymbol{f_p} = \boldsymbol{f} \tag{3}$$
$$\boldsymbol{m_p} = \boldsymbol{p} \times \boldsymbol{f} \tag{4}$$

The sums of the $\boldsymbol{f_p}$ and $\boldsymbol{m_p}$ are amounts of force and torque acting on the rigid body.

The contact volume becomes a convex polyhedron. We integrate forces and torques for each triangle, which consist of the convex polyhedron of contact volume. Then we sum to find the amount of force and torque working overall rigid body. This section shows notation about the contact plane and a triangle on the contact volume.

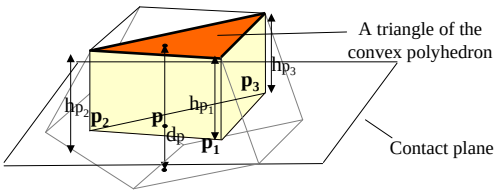

Figure 7: *Triangular decomposition*

For a triangle of a convex polyhedron (Figure 7), we define:

- $\boldsymbol{n}$: normal vector of the contact plane.
- $\boldsymbol{p}$: coordinates of a point on the contact plane.
- i: vertex ID (1,2,3).
- $\boldsymbol{p_i}$: coordinates of vertices of the triangle projected on the contact plane.
- $\overline{*}$: average of values on three vertices. For example, $\overline{\boldsymbol{p}}$ represents $(\boldsymbol{p_1}+\boldsymbol{p_2}+\boldsymbol{p_3})/3$.
- h_p: distance between vertices on the plane of the convex polyhedron and contact plane.
- $\boldsymbol{v_p}$: relative velocity of two rigid bodies at $\boldsymbol{p}$. $\boldsymbol{v_p} = (\boldsymbol{v_b}+\boldsymbol{\omega_b}\times(\boldsymbol{p}-\boldsymbol{c_b})) - (\boldsymbol{v_a}+\boldsymbol{\omega_a}\times(\boldsymbol{p}-\boldsymbol{c_a}))$ for the velocities of $\boldsymbol{v_a}, \boldsymbol{v_b}$, the angular velocities of $\boldsymbol{\omega_a}, \boldsymbol{\omega_b}$, and the position of the center of gravities of $\boldsymbol{c_a}, \boldsymbol{c_b}$ for the two rigid bodies.
- $\boldsymbol{v_p^N}$: component of $\boldsymbol{v_p}$, which is orthogonal to the contact plane $((\boldsymbol{v_p}\cdot\boldsymbol{n})\boldsymbol{n})$.
- $\boldsymbol{v_p^T}$: component of $\boldsymbol{v_p}$, which is parallel to the contact plane $(\boldsymbol{v_p}-(\boldsymbol{v_p}\cdot\boldsymbol{n})\boldsymbol{n})$.
- $\boldsymbol{\omega_r}$: relative angular velocity. $(\boldsymbol{\omega_b}-\boldsymbol{\omega_a})$
- $\boldsymbol{v_r}$: relative velocity. $(\boldsymbol{v_b}-\boldsymbol{v_a}-(\boldsymbol{\omega_b}\times\boldsymbol{c_b}-\boldsymbol{\omega_a}\times\boldsymbol{c_a})$
- S_t: the area of the triangle $(\|(\boldsymbol{p_2}-\boldsymbol{p_1})\times(\boldsymbol{p_3}-\boldsymbol{p_1})\|/2)$.
- S_c: the area of the contact.
- $\boldsymbol{f^N}$: normal force.
- $\boldsymbol{f^D}$: dynamic friction force.
- $\boldsymbol{f^S}$: static friction force.
- $\boldsymbol{f^M}$: maximum static friction force.
- $\boldsymbol{f^{*_s}}$: force from spring model. ($\boldsymbol{f^{N_s}}$:normal force from spring model)
- $\boldsymbol{f^{*_d}}$: force from the damper model.
- $\boldsymbol{f^*_{tri}}$: force acting on the triangle.

4.5. Calculation of normal forces

We consider spring-damper models distribute on the contact area; then, we define normal forces as the forces from the models. The force from the spring model and the damper model are proportional to the depth of the two objects and the normal component of the relative velocity between the two objects respectively.

Normal forces from the spring models. The normal force $\Delta\boldsymbol{f_p^{N_s}}$ and torque $\Delta\boldsymbol{m_p^{N_s}}$ from the distributed spring model on a small area ΔS around a point $\boldsymbol{p}$ are

$$\Delta\boldsymbol{f_p^{N_s}} = k^N\frac{\Delta S}{S_c}d_p\boldsymbol{n} \tag{5}$$
$$\Delta\boldsymbol{m_p^{N_s}} = k^N\frac{\Delta S}{S_c}\boldsymbol{p}\times d_p\boldsymbol{n} \tag{6}$$

for the penetration depth of d_p. k^N is the spring coefficient for the spring model before distribution on contact area.

Dividing penetration d_p into that of front-sides and back-sides of the contact plane h_ps, we integrate them for each triangle:

$$\boldsymbol{f_{tri}^{N_s}} = \frac{k^N}{S_c}\int_{\boldsymbol{p}\in S_t} h_p\boldsymbol{n}dS = k^N\frac{S_t}{S_c}\overline{h}\boldsymbol{n} \tag{7}$$
$$\begin{aligned}\boldsymbol{m_{tri}^{N_s}} &= \frac{k^N}{S_c}\int_{\boldsymbol{p}\in S_t}\boldsymbol{p}\times(h_p\boldsymbol{n})dS \\ &= k^N\frac{S_t}{S_c}\left(\frac{3}{4}\overline{h}\overline{\boldsymbol{p}}+\frac{1}{12}(h_{p_1}\boldsymbol{p_1}+h_{p_2}\boldsymbol{p_2}+h_{p_3}\boldsymbol{p_3})\right)\times\boldsymbol{n}.\end{aligned} \tag{8}$$

Normal forces from the damper models. The force $\Delta\boldsymbol{f_p^{N_d}}$ and torque $\Delta\boldsymbol{m_p^{N_d}}$ from the distributed damper model on small area ΔS are

$$\Delta\boldsymbol{f_p^{N_d}} = b^N\frac{\Delta S}{S_c}\boldsymbol{v_p^N} \tag{9}$$
$$\Delta\boldsymbol{m_p^{N_d}} = b^N\frac{\Delta S}{S_c}\boldsymbol{p}\times\boldsymbol{v_p^N}. \tag{10}$$

b^N is the damper coefficient of the model before distribution.

We integrate them over the triangle:

$$\boldsymbol{f_{tri}^{N_d}} = \frac{b^N}{S_c}\int_{p\in S_t}\boldsymbol{v_p^N}dS = b^N\frac{S_t}{S_c}\boldsymbol{n}\cdot(\boldsymbol{v_r}+\omega_r\times\overline{\boldsymbol{p}})\boldsymbol{n} \tag{11}$$
$$\begin{aligned}\boldsymbol{m_{tri}^{N_d}} &= \frac{b^N}{S_c}\int_{\boldsymbol{p}\in S_t}\boldsymbol{p}\times\boldsymbol{v_p^N}dS \\ &= b^N\frac{S_t}{S_c}\left(\frac{3}{4}(\boldsymbol{n}\cdot\boldsymbol{v_r})\overline{\boldsymbol{p}}\times\boldsymbol{n}+\frac{1}{12}((\boldsymbol{p_1}\cdot(\boldsymbol{n}\times\boldsymbol{\omega_r}))\boldsymbol{p_1}+\right.\end{aligned}$$

$$(\boldsymbol{p_2}\cdot(\boldsymbol{n}\times\boldsymbol{\omega_r}))\boldsymbol{p_2}+(\boldsymbol{p_3}\cdot(\boldsymbol{n}\times\boldsymbol{\omega_r}))\boldsymbol{p_3})\times\boldsymbol{n}. \quad (12)$$

4.6. Calculation of friction forces

The proposed simulator employs the Coulomb friction model. The Coulomb friction model has two states: a dynamic state and a static state. A contact point $\boldsymbol{p}$ on an object in the static state has a constraint force (static friction force $\boldsymbol{f_p^S}$), which stops the object. If the static friction force exceeds the maximum static friction force $\boldsymbol{f_p^M}$, the object begins to move and comes to a dynamic state. A contact point on an object in the dynamic state gets a dynamic friction force $\boldsymbol{f_p^D}$, whose direction is identical to the normal component of the relative velocity $\boldsymbol{v_p^T}$ on the contact point. For the normal force of $\boldsymbol{f_p^N}$, the maximum static friction force becomes $\boldsymbol{f_p^M}=\mu_0\|\boldsymbol{f_p^N}\|(\boldsymbol{f_p^S}/\|\boldsymbol{f_p^S}\|)$ and the dynamic friction force becomes $\boldsymbol{f_p^D}=\mu\|\boldsymbol{f_p^N}\|(\boldsymbol{v_p^T}/\|\boldsymbol{v_p^T}\|)$. The proposed simulator considers that the friction force from a small area on the contact plane is given by the Coulomb friction model and that the sum of the friction forces acts on the rigid body.

Dynamic friction force. The dynamic friction force of $\Delta\boldsymbol{f_p^D}$ on a small area ΔS around a point $\boldsymbol{p}$ on the contact plane is

$$\Delta\boldsymbol{f_p^D} = \mu\|\Delta\boldsymbol{f_p^N}\|(\boldsymbol{v_p^T}/\|\boldsymbol{v_p^T}\|). \quad (13)$$

However, analytical integration of this equation is difficult. Therefore, we interpolate dynamic friction forces on the three vertices of the triangle:

$$\Delta\hat{\boldsymbol{f}}_{\boldsymbol{p}}^{\boldsymbol{D}} = a_1\Delta\boldsymbol{f}_{p_1}^D+a_2\Delta\boldsymbol{f}_{p_2}^D+a_3\Delta\boldsymbol{f}_{p_3}^D \quad for\ \ \boldsymbol{p}=a_1\boldsymbol{p_1}+a_2\boldsymbol{p_2}+a_3\boldsymbol{p_3}. \quad (14)$$

The dynamic friction force $\boldsymbol{f_{tri}^D}$ and torque $\boldsymbol{m_{tri}^D}$ from the triangle can be written as following:

$$\boldsymbol{f_{tri}^D} = \int_{\boldsymbol{p}\in S_t}\hat{\boldsymbol{f}}_{\boldsymbol{p}}^{\boldsymbol{D}}dS=\frac{S_t}{S_c}\overline{\boldsymbol{f^D}} \quad (15)$$

$$\boldsymbol{m_{tri}^D} = \int_{\boldsymbol{p}\in S_t}\boldsymbol{p}\times\hat{\boldsymbol{f}}_{\boldsymbol{p}}^{\boldsymbol{D}}dS=\frac{S_t}{S_c}(\frac{3}{4}\overline{\boldsymbol{p}}\times\overline{\boldsymbol{f^D}} +\frac{1}{12}(\boldsymbol{p_1}\times\boldsymbol{f}_{p_1}^D+\boldsymbol{p_2}\times\boldsymbol{f}_{p_2}^D+\boldsymbol{p_3}\times\boldsymbol{f}_{p_3}^D)). \quad (16)$$

$$(\boldsymbol{f}_{p_i}^D\equiv\Delta\boldsymbol{f}_{p_i}^D/\Delta S) \quad (17)$$

Static friction forces. A static friction force is a constraint force like a normal force. The proposed simulator does not solve the constraint directly. Instead, the simulator employs penalty methods. The simulator presumes that a spring-damper model distributes on the contact plane. Each side of the spring is connected to each rigid body. Each rigid body gets a restitution force when it moves. We define the static friction force as this restitution force.

Static friction force $\Delta\boldsymbol{f_p^{S_s}}$ and torque $\Delta\boldsymbol{m_p^{S_s}}$ from a distributed spring model on a small area ΔS can be written as the following equations for the spring expansion of $\boldsymbol{l_p}$ and the spring coefficient of k^S:

$$\Delta\boldsymbol{f_p^{S_s}} = -\frac{k^S\Delta S}{S_c}\boldsymbol{l_p} \quad (18)$$

$$\Delta\boldsymbol{m_p^{S_s}} = -\boldsymbol{p}\times\frac{k^S\Delta S}{S_c}\boldsymbol{l_p} \quad (19)$$

Displacement of a rigid body can be represented by translation of $\boldsymbol{r}$ and rotation of $\boldsymbol{\theta}$ around the origin. Extension of the spring model $\boldsymbol{l_p}$ can be represented as:

$$\boldsymbol{l_p}=\boldsymbol{r}+\boldsymbol{\theta}\times\boldsymbol{p}. \quad (20)$$

From the above, the friction force $\boldsymbol{f_{tri}^{S_s}}$ and torque $\boldsymbol{m_{tri}^{S_s}}$ from each triangle of contact area can be written as following:

$$\boldsymbol{f_{tri}^{S_s}} = \int_{\boldsymbol{p}\in S_t}-\frac{k^S}{S_c}l_p dS=-k^S\frac{S_t}{S_c}(\boldsymbol{r}+\boldsymbol{\theta}\times\overline{\boldsymbol{p}}) \quad (21)$$

$$\boldsymbol{m_{tri}^{S_s}} = \int_{\boldsymbol{p}\in S_t}-\boldsymbol{p}\times\frac{k^S}{S_c}l_p dS = -k^S\frac{S_t}{S_c}(\boldsymbol{r}\times\overline{\boldsymbol{p}}+\boldsymbol{\theta}\frac{1}{6}(\boldsymbol{p_1^2}+\boldsymbol{p_2^2}+\boldsymbol{p_3^2} +\boldsymbol{p_1p_2}+\boldsymbol{p_2p_3}+\boldsymbol{p_3p_1}). \quad (22)$$

The total friction force $\boldsymbol{f}^{S_s}$ and torque $\boldsymbol{m}^{S_s}$ from spring models on the contact area C become

$$\boldsymbol{f}^{S_s} = -k^S(\sum_{tri\in C}\boldsymbol{r}\frac{S_t}{S_c}+\boldsymbol{\theta}\times\sum_{tri\in C}\overline{\boldsymbol{p}}\frac{S_t}{S_c}) \quad (23)$$

$$\boldsymbol{m}^{S_s} = -k^S(\boldsymbol{r}\times\sum_{tri\in C}\frac{S_t}{S_c}\overline{\boldsymbol{p}}+\boldsymbol{\theta}\frac{1}{6}\sum_{tri\in C}(\boldsymbol{p_1^2}+\boldsymbol{p_2^2}+\boldsymbol{p_3^2} +\boldsymbol{p_1p_2}+\boldsymbol{p_2p_3}+\boldsymbol{p_3p_1})\frac{S_t}{S_c} \quad (24)$$

.

The force $\Delta\boldsymbol{f_p^{S_d}}$ and torque $\Delta\boldsymbol{m_p^{S_d}}$ from the damper model on a small area ΔS around a point $\boldsymbol{p}$ on the contact plane are

$$\Delta\boldsymbol{f_p^{S_d}} = -\frac{b^S\Delta S}{S_c}\boldsymbol{v_p^T} \quad (25)$$

$$\Delta\boldsymbol{m_p^{S_d}} = -\boldsymbol{p}\times\frac{b^S\Delta S}{S_c}\boldsymbol{v_p^T} \quad (26)$$

for a damper coefficient of b^S. $\boldsymbol{v_p^T}$ can be written as

$$\boldsymbol{v_p^T} = \boldsymbol{v^T}+\boldsymbol{\omega^T}\times\boldsymbol{p}. \quad (27)$$

for $\boldsymbol{v^T}$ (component of the relative velocity which is parallel to the contact plane) and $\boldsymbol{\omega^T}$(component of the relative angular velocity which is parallel to the contact plane). The friction force $\boldsymbol{f}^{S_d}$ and torque $\boldsymbol{m}^{S_d}$ from the damper model on the contact area C become

$$\boldsymbol{f}^{S_d} = -b^S(\sum_{tri\in C}\boldsymbol{v^T}\frac{S_t}{S_c}+\boldsymbol{\omega^T}\times\sum_{tri\in C}\overline{\boldsymbol{p}}\frac{S_t}{S_c}) \quad (28)$$

$$\boldsymbol{m}^{S_d} = -b^S(\boldsymbol{v^T}\times\sum_{tri\in C}\frac{S_t}{S_c}\overline{\boldsymbol{p}}+\boldsymbol{\omega^T}\frac{1}{6}\sum_{tri\in C}$$

$$(p_1^2+p_2^2+p_3^2+p_1p_2+p_2p_3+p_3p_1)\frac{S_t}{S_c}). \quad (29)$$

The maximum static friction force. We consider that the maximum friction force acting on the contact area is the sum of the maximum friction forces from the distributed coulomb models.

Friction forces $\Delta f_p^S = \Delta f_p^{S_s} + \Delta f_p^{S_d}$ working on a small area around a point on the contact area can be found from Eq. (18) Eq. (25). The direction of the static friction at each point can be calculated from $r, \theta, v, and \omega$.

The equation of the maximum static friction force is identical to the dynamic friction force, except for the friction coefficient and direction of the force. The maximum static friction force of Δf_p^M from a small area around a point p is given by the equation created from Eq. (13), changing $v_p^T/\|v_p^T\|$ into $f_p^S/\|f_p^S\|$, and μ into μ_0.

The maximum friction torque f_{tri}^M acting on the triangle can be approximated as the equation created from Eq. (15),Eq. (16) changing f_p^D into f_p^S.

State transition and calculation of friction forces. To find friction force working on a rigid body, we must address the state transition between static and dynamic states and find friction forces from the spring-damper models. Figure 8 shows states and conditions for transitions and extension of the spring model (r, θ) for each state. The procedure for

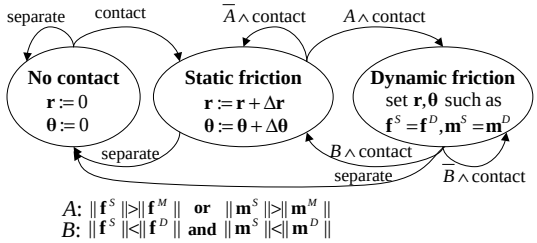

Figure 8: *State transition for friction*

iteration of the simulation can be written as following:

1. When two rigid bodies contact, we set relative displacements of r and θ at zero. The initial state is a static state.
2. In the static state, we add relative displacement of two rigid bodies $\Delta r, \Delta\theta$ to r, θ and calculate the static friction of f^S, m^S.
3. If the static friction force is smaller than the dynamic friction force, the state changes to the static state. If larger than the maximum friction force, the state changes to the dynamic state. In the static state, if the size of static friction $\|f^S\|, \|m^S\|$ exceeds the maximum static friction $\|f^M\|, \|m^M\|$, we update r, θ to satisfy $f^S = f^D, m^S = m^D$. We can find r and θ from Eq. (23), Eq. (24), Eq. (28) and Eq. (29).
4. We employ static friction of f^S, m^S for the static state and dynamic friction of f^D, m^D for the dynamic state.

5. Evaluation

We made three experiments to test whether the proposed simulator solved the problems of the previous method. In addition, we used the proposed method to produce two virtual environments to test practicality of our proposed method.

5.1. Simulators for experiments

We compared three simulation methods: the proposed method, a point-based penalty method and a constraint-based method. We used the Open Dynamic Engine [Smi00] for the simulator with the constraint-based method. We omitted contact analysis part from the proposed method and calculated contact forces from the deepest penetration to create the point-based simulator.

5.2. Experiment 1: simulation speed test

We compared computation time to test whether the proposed method was faster and better for haptic interactions.

We simulated stacked blocks as in Figure 9 and measured the relation between the number of blocks and the computation time. The period of each step of the simulation was 5 ms; the acceleration of gravity was set to $9.8 m/s^2$. The blocks were 1 m x 1 m x 2 m and each block was 1 kg.

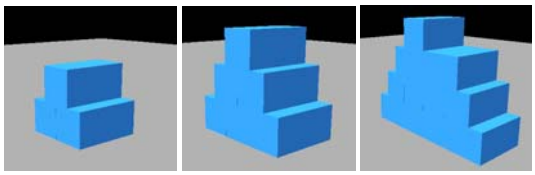

Figure 9: *Examples of simulated scenes for evaluation: 3, 6 and 13 blocks.*

5.3. The result of experiment 1

Figure 10 shows the relation between the number of blocks and computation time of one iteration of the simulation. The computation time of the constraint-based method increased sharply when the number of blocks increased. On the other hand, the computation time of proposed simulator and point-based simulator increased linearly. The point-based simulator was faster than proposed one. However, the point-based simulator was not stable and the blocks collapse in 1.8 seconds. We inferred that our proposed simulator was better for haptic interactions.

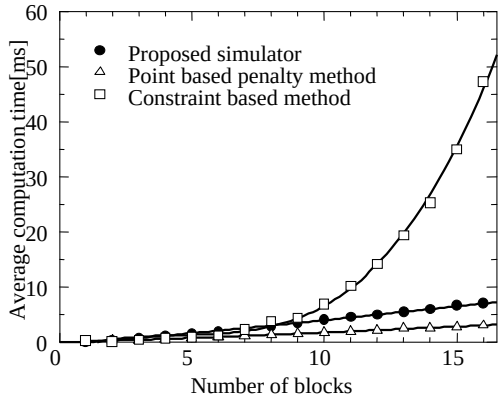

Figure 10: *Comparison of computation times*

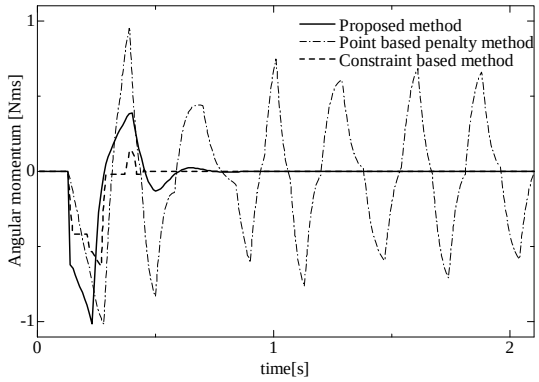

Figure 12: *Angular momentum of the cube around the z-axis*

5.4. Experiment 2: stability test

We performed the second experiment to examine whether the proposed method solved the vibration problem mentioned in section 3.

We simulated a cube falling on a floor to evaluate stability of simulations. We measured the angular momentum of the cube around the z-axis (Figure 11).

- The length of each edge of the cube was 2 m. The mass was 1 kg.
- Initial position was 1 m above the floor, tilting 0.1 rad.
- The period of one simulation step was 10 ms.
- The spring coefficient was 1500 [N/m] and the damper coefficient was 1.5 [Ns/m].

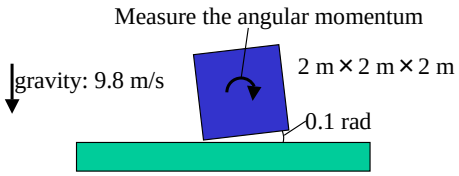

Figure 11: *Simulated scene*

5.5. The result of experiment 2

Figure 12 shows the angular momentum of the cube. The cube in the proposed simulator and constraint-based simulator stopped after landing. However, the cube in the point-based simulator vibrated continuously after landing.

5.6. Experiment 3: comparison of real and virtual stick-slip

We compared stick-slip motion of a weight in the real world and the simulators to test the accuracy of the simulation for friction forces. Figure 13 shows experimental setup in the real world. A weight was connected to a spring. The spring was pulled at a constant velocity. We chose surface materials and other parameters to cause stick-slip phenomena. We measured position of the weight and tension force and calculated friction force from them.

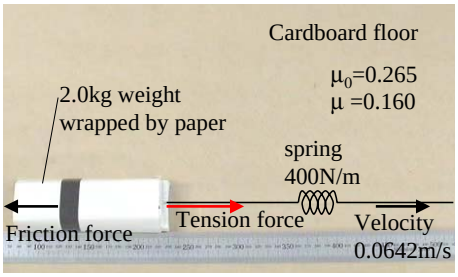

Figure 13: *Experimental setup for stick-slip motion in the real world*

5.7. The result of experiment 3

Figure 14 shows the motion of the weight and friction and tension forces. Stick-slip motion was observed in the real world and the proposed simulator. The friction force in the constraint-based simulator showed that the simulator treated only dynamic frictions. The point-based simulator did not maintain static friction state. The contact point moved around on the contact area and the contact was not stable.

In the real world, the stick-slip motion of the weight decreased little by little. Proposed simulator replicated this motion by adding a small damper model.

5.8. Example 1: direct manipulation environment

We created a direct manipulation environment with the proposed simulator and haptic interfaces. We used two 6-DOF

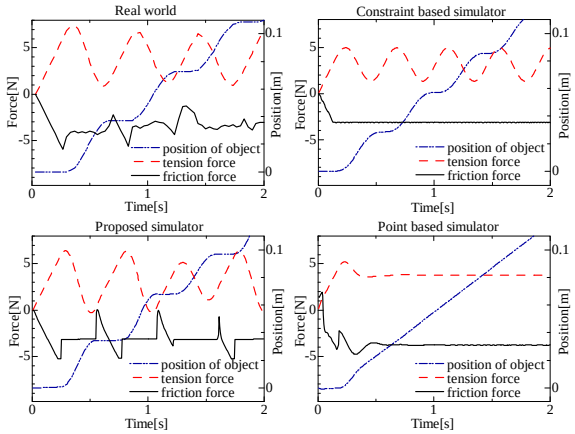

Figure 14: *Slip motion of the weight*

haptic interfaces named SPIDAR-G [KHKS02] for this example. The virtual environment had six blocks on a floor and two shovels, which were used as haptic pointers.

As a result, we confirmed that users were able to manipulate objects directly and intuitively. Figure 15 shows the images of direct manipulations and the computation time for the simulation. Users were able to hold, pick up, pass, and throw blocks with realistic 6-DOF force feedback including friction force and a collision impulse. The computation time for each iteration was less than 2.8 ms and less than 2 ms in most cases. That was fast enough for natural haptic interaction.

5.9. Example 2: the effect of the distributed friction forces

The proposed simulator addresses distribution of friction forces. To examine this effect, we simulated motion of a spinning top in the proposed simulator and the point-based simulator.

The proposed simulator simulated motion of the top from spinning by the tip to rolling on the floor and stopping without any special operation. However, the top in the point-based simulator moved unnaturally and did not cease rotating after falling down. Figure 16 shows images of the simulation.

6. Conclusions

In this study, we created a new rigid body motion simulator based on a penalty method regarding contact volumes to achieve responsiveness and correctness for haptic displays. We evaluated and confirmed that the proposed method was fast and stable enough for haptic interaction and found correct contact forces especially for friction forces.

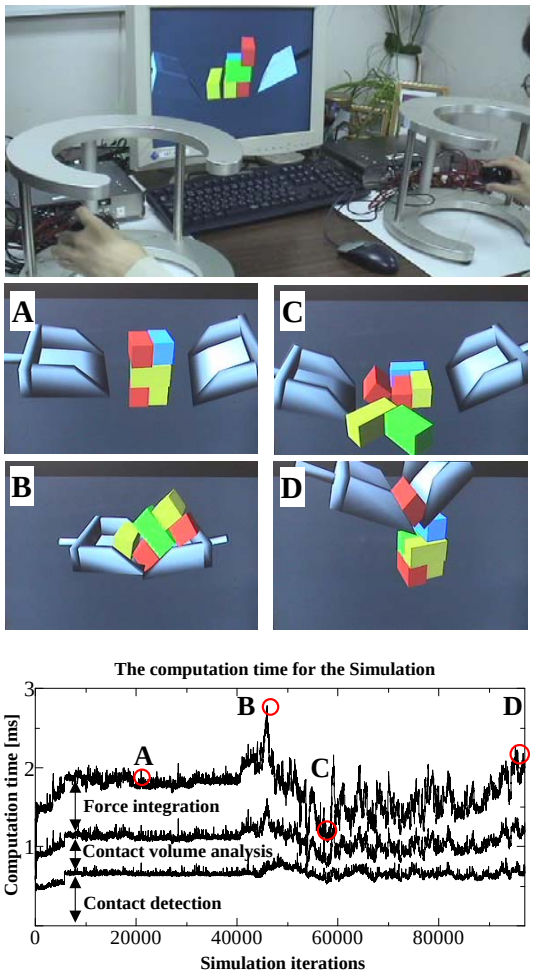

Figure 15: *Interaction with haptic interfaces: images and the computation time. The simulation was performed by a PC with a CPU of Pentium 4 2.8GHz.*

References

[AKO95] ADACHI Y., KUMANO T., OGINO K.: Intermediate representation for stiff virtual objects. *Proc. IEEE Virtual Reality Annual International Symposium '95* (1995). 2

[Bar89] BARAFF D.: Analytical methods for dynamic simulation of non-penetrating rigid bodies. *Computer Graphics 23* (1989). (SIGGRAPH 89). 2, 3

[Bar94] BARAFF D.: Fast contact force computation for nonpenetrating rigid bodies. *Computer Graphics* (1994). (SIGGRAPH94). 2, 3

[BDH96] BARBER C. B., DOBKIN D. P., HUHDANPAA H.: The quickhull algorithm for convex hulls.

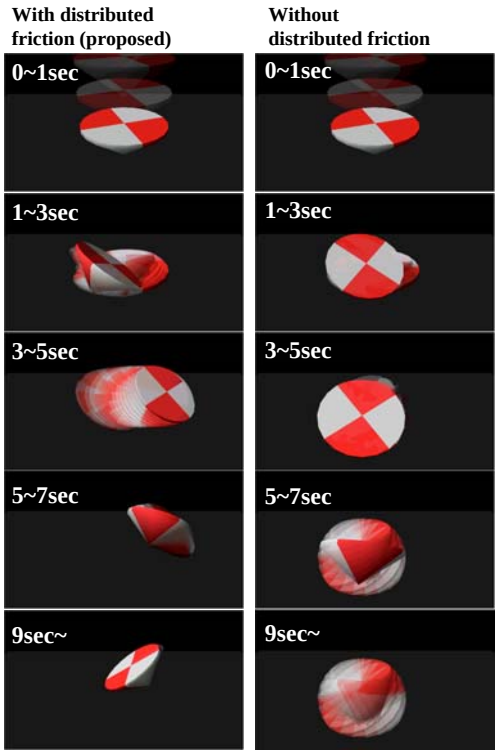

Figure 16: *Comparison of motions of spinning tops*

ACM Transactions on Mathematical Software 22 (1996). 4

[BH00] BERKELMAN P. J., HOLLIS R. L.: Lorentz magnetic levitation for haptic interaction: Device design, performance, and integration with physical simulations. *Int'l J. of Robotics Research 19*, 7 (July 2000). 2

[CC97] CHANG B., COLGATE J.: Real- time impulse-based simulation for haptic display. *Proc. of the 1997 ASME International Mechanical Engineering Congress and Exhibition* (1997). 2

[GJK88] GILBERT E. G., JOHNSON D. W., KEERTHI S. S.: A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal of Robotics and Automation 4*, 2 (1988). 4

[HIKS99] HASEGAWA S., ISHII M., KOIKE Y., SATO M.: Inter-process communication for force display of dynamic virtual world. *ASME Dynamic Systems and Control Division* (1999). 2

[Hip03] HIPPMANN G.: An algorithm for compliant contact between complexly shaped surfaces in multibody dynamics. *Proceedings of Multibody Dynamics 2003* (2003). 2

[KHKS02] KIM S., HASEGAWA S., KOIKE Y., SATO M.: Tension based 7-dof force feedback device: Spidar-g. *Proceedings of the IEEE Virtual Reality 2002* (2002), 283–284. 9

[KOLM03] KIM Y., OTADUY M., LIN M., MANOCHA D.: Six-degree-of-freedom haptic display using incremental and localized computations. *Presence 12*, 3 (June 2003). 2

[KSZ93] KELLER H., STOLZ H., ZIEGLER A.: Virtual mechanics: Simulation and animation of rigid body systems. *http://citeseer.nj.nec.com/keller94virtual.html* (1993). 2

[LB95] LOVE L., BOOK W.: Contact stability analysis of virtual walls. *Proc. of Dynamic Systems and Control Division ASME* (1995). 1

[MC95] MIRTICH B., CANNY J.: Impulse-based simulation of rigid bodies. *Proceedings of 1995 Symposium on Interactive 3D Graphics* (1995). 2

[MF78] MULLER D. E., F.P.PREPARATA: Finding the intersection of two convex polyhedra. *Theoretical Computer Science 7*, 2 (1978). 4

[MPT99] MCNEELY W., PUTERBAUGH K., TROY J.: Six degree-of-freedom haptic rendering using voxel sampling. *Proc. ACM SIGGRAPH 1999* (1999). 2

[MW88] MOORE M., WILHELMS J.: Collision detection and response for computer animation. *Computer Graphics 22* (1988). (SIGGRAPH 88). 2

[MZ90] MCKENNA M., ZELTZER D.: Dynamic simulation of autonomous legged locomotion. *Computer Graphics 24* (1990). (SIGGRAPH 90). 2

[Smi00] SMITH R.: Open dynamics engine. *http://opende.sourceforge.net/* (2000). 7

[SS98] SAUER J., SCHOMER E.: A constraint-based approach to rigid body dynamics for virtual reality applications. *Proceedings of the ACM symposium on Virtual reality software and technology 1998* (1998), 153–162. 2, 3, 11

[SWF*93] SNYDER J., WOODBURY A., FLEISCHER K., CURRIN B., BARR A.: Interval methods for multi-point collisions between time-dependent curved surfaces. *Proc. ACM SIGGRAPH 1993* (1993). 2

[TPBF87] TERZOPOULOS D., PLATT J., BARR A., FLEISCHER K.: Elastically deformable models. *Computer Graphics 21* (1987). (SIGGRAPH 87). 2

Appendix A: Extra video clips

Followings are extra video clips for sample simulation scenes.

A long video clip including everything

This includes all experiments and examples.

6DOF interactions (Figure 17)

This shows some examples of interactions with a 6DOF haptic interface.

Figure 17: *6DOF interactions*

Interaction with an insect (Figure 18)

An insect walking on a floor. The insect has 27 total DOF: 3 for head, 2 for each groin, 1 for each knee and ankle. The precise simulation of friction forces between the floor and the legs realizes a natural walking motion.

Figure 18: *Interaction with an insect*

A tippe top (Figure 19)

Simulations of mechanical toys are good evaluation for dynamics simulators. Sauer and Schoemer [SS98] used a tippe top for an evaluation example. We simulated the same toy. We modeled the shape of the top by 192 polygons. Yellow lines on the video clip represent contact forces.

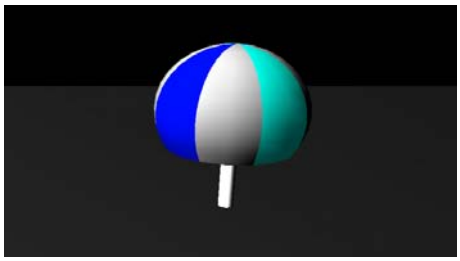

Figure 19: *A tippe top*

Gears and a cube (Figure 20)

The gear at the right side is self-actuated. The friction forces between the cube and gears decrease the speed of the rotation. Yellow lines on the video clip represent contact forces.

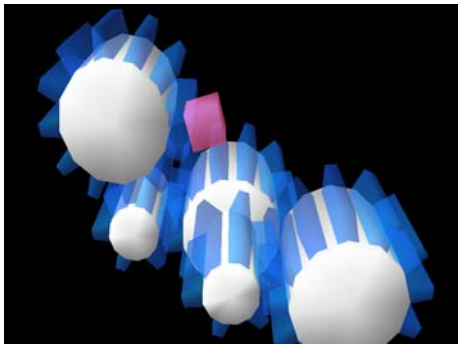

Figure 20: *Gears and a cube*

Appendix B: Source codes

We are creating an open source virtual environment development software named Springhead. All source codes related to this research are included to the software. Please visit the web site for source codes and more demonstrations.