Real-time Rigid Body Simulation Based on Volumetric Penalty Method

Shoichi HASEGAWA
Precision and Intelligence Laboratory
Tokyo Institute of Technology
4259 Nagatsuta-cho, Midori-ku, Yokohama, Japan
hase@hi.pi.titech.ac.jp

Nobuaki FUJII
nfujii@hi.pi.titech.ac.jp

Yasuharu KOIKE
koike@pi.titech.ac.jp

Makoto SATO
msato@pi.titech.ac.jp

Abstract

This paper proposes a new method for real-time rigid body simulations based on a volumetric penalty method. The penalty method, which employs spring-damper model, is a simple and useful method for real-time simulation of multi-bodies. However, simple penalty method cannot handle face-face contact, because simple penalty method cannot find application point of reflection force.

We suppose distributed small spring-damper model to solve the problem. We analyze intersecting part of bodies and integrate forces and torques from distributed spring-damper models. We implement the simulator and compare our simulator with simple penalty method. It showed that our simulator solve the face-face contact problem. In addition, we attach haptic interface to the simulator for interaction. It shows that we were able to interact with virtual world by haptic interfaces.

1. Introduction

This paper proposes novel real-time rigid body motion simulator based on volumetric penalty method, which has stable update cycle.

Recent progress of computer technology encourages development of applications, which create and use virtual worlds. Natural interactions between users and virtual worlds are important to extend application of the virtual worlds. Haptic interfaces display haptic sensation as a response of touch. Therefore, haptic interfaces help natural manipulation of virtual object and extend application area of virtual world. For example, Nahvi et al.[1] created simple mechanism in the virtual world to aid the design process. In their system, the user can manipulate the mechanism with a haptic interface.

In such systems, the interaction of the user must be reflected to the virtual world in real-time. Therefore, the system requires real-time update of virtual world.

On the other hand, objects in the real world obey laws of motion. Simulations of laws of motion create natural motion of virtual object. Therefore, we develop a real-time rigid body simulator with stable update rate.

2. Background

Baraff [2] [3] proposes analytical methods, which treats contacts as constraints. His method solves contact forces, which prevent penetration, regarding momentum conservation law. His method processes multi-contacts at once and the time step of simulation can be constant. Therefore, his method can be used for real-time simulator like *Open Dynamics Engine* [4]. However, Baraff's method takes $O(n^3)$ (where n represents number of contacts) computation time to solve the contact forces. Therefore, when many contacts occur at the same moment, the computation takes a lot of time. Therefore, Baraff's method is not suitable for simulators for haptic displays.

Mirtich [5] proposes impulse-based simulations. His method searches the time of the impact rather than treating concurrent impacts. In his method, individual steps of the simulation don't require much computation, but when many collisions occur within a short period, the simulation steps become very small. Therefore, sometimes the simulation speed becomes very slow. Therefore, it is difficult to create real-time simulator with Mirtich's method.

Chang and Colgate [6] proposes real-time impulse-based simulation for haptic display. In their simulator, the time step is constant and the simulation speed is enough for haptic display. However, their simulator is two-dimensional and has a problem in face-face contact.

McKenna and Zeltzer[7] and Keller et al. [8] calculates contact forces from spring-damper models. In these method, a contact force calculates from the amount of penetration (= penalty). Therefore, these method called 'penalty method'. Penalty methods are simple and useful method for real-time applications. Penalty methods put multiple spring and damper models for multiple contact points and solves multiple contact forces at once. Because the contact force is calculated from spring-damper model directly, penalty methods take computation time of liner order. Therefore, penalty methods are suitable for simulators with haptic displays.

Baraff [2] wrote that penalty methods for rigid bodies are often computation-ally expensive, give only approximate results, and may require adjustments for different simulation conditions.

However, for a real-time simulation, small computation time for each time step is much more important than total computation time and the adjustments for simulation conditions can be automated. In addition, the adjustment of the condition of the simulation such as a spring-coefficient can be automated to a certain extent by using the information such as the mass of the object . Moreover, Baraff's method gives also approximate results, because his method does not regard the reflection coefficient.

Above is the reason why we choose the penalty method.

When faces of objects parallelly contacts each other, previous penalty method can't decide the position of the application point on which penalty force acts.

In this paper, we integrate penalty and its moment over the contact area. The integrated penalty and moment tells us the position of the application point.

3. Proposing Simulator

Motions of rigid bodies are represented by the initial state and equation of motion. Information of a rigid body such as position and velocity can be calculated by numerical integration of equation of motion from the initial state.

The term of external force of equation of motion changes the motion of the object. There are several external forces, which affect the motions of rigid bodies such as:

- Forces from fields such as gravity or air resistance.
- Forces from contacts of objects.

Forces from fields can be calculated directly from the nature of the virtual space. On the other hand, to determine the forces from the contacts, we have to detect the contact state of objects and calculate generated force from the contacts.

When two rigid bodies contact each other on some points or regions, each rigid body receives forces from some points or regions. Because the motion of a rigid body can be represented by translation and rotation, these forces can be represented by a translating force and a rotating force.

3.1. Problem of previous penalty methods

Previous penalty methods [7] [8] consider that contacts always occur on a point and apply penalty force on that point. These method does not consider a contact on large regions such as a contact of cube on a floor.

Terzopoulos et al. [9] samples many points on the surfaces of objects and calculate penalty force for each sampled points. Their method can treat large contact regions. However, their method requires a lot of computation time and is not suitable for real time simulators. Snyder et al. [10] formulated face-face contact problem of implicit curved surfaces into minimization problem with multiple solutions, and solved it with Interval Newton Method. However, their method doesn't work in real time.

To illustrate the problem, we'd like to take an example. Let's consider a simple penalty method, which put spring-damper model on the most penetrating point and a virtual world where a cube is on a floor. When the cube contacts floor, one of the vertices of the cube, which penetrates most to the floor, gets the penalty-force from the floor. Then, the cube begins to rotate and the most penetrating point changes. Therefore, the cube always rotates a little and does not stop (Fig.1). Keller et al. [8] points out this problem.

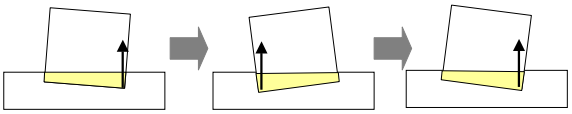

Figure 1. Problem of simple penalty method

3.2. Proposing penalty method

To calculate the force and the moment from the contact, Our method supposes that small spring-damper models distribute on surfaces of objects. our method analyses the shape of the intersecting part of two contact object and integrate the force and moment generated by the distributed small spring-damper models(Fig.2).

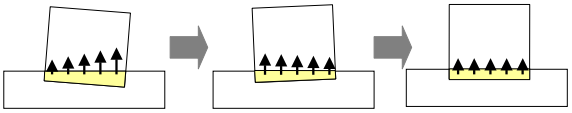

Figure 2. Solution by our method

4. Implementation

Our simulator detects contacts, analyses shape of intersecting part, and integrate penalty forces. In this section, we'll explain the details of these operations.

4.1. Contact analysis

Our simulator represents shapes of objects by convex polyhedrons. Because there are efficient algorithms for convex polyhedron to detect the contact and analyze the shape of the intersecting part. Objects, which have non-convex shape, can be represented by composition of convex polyhedrons.

There are some methods for contact detection [14] [15] [12]. We choose GJK algorithm [12] for contact detection, because we need a point located on the intersecting part of two convex. GJK algorithm finds closest points of two separate convex, or a point located on the intersecting part of two intersecting convex. SOLID [13] is an implementation of GJK algorithm. We use SOLID for the contact detection.

4.2. Analysis of intersecting part

Our simulator analyses intersecting part of objects. Because the objects are represented by convex polyhedrons, the intersecting part is represented by the common part of two convex polyhedrons.

Muller and Preparata [17] proposes an algorithm, which seeks the faces and vertices of common part from the planes of the convexes and a point located in the common part. Following is outline of their algorithm (Fig.4). Their algorithm represents convexes by half space representations. Half space representation is common part of half spaces, which represented by planes. The concatenation of two half-space representations results in the intersection of the two convex polyhedrons. However, the concatenated representation has redundant planes. Therefore, we have to find the minimum set of the planes, which represents the intersection part. To find the minimum set of the planes, their algorithm transforms planes via dual transformation. Dual transformation is a transformation which transform a vertex (a, b, c) into a face $(ax + by + cz = 1)$ and a face into a vertex (Fig.3). After the dual transformation, the problem of the finding of the minimum set of the planes results in a convex hull problem, which find the minimum convex from the vertices. There are fast algorithms to solve convex hull problem. Their algorithm transforms convex hull via dual transformation again. Then, the convex hull is transformed into the intersecting part. Now, we got the vertices and faces of the intersection.

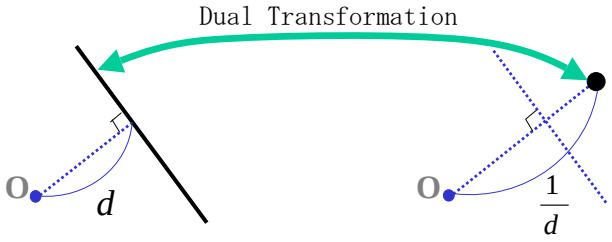

Figure 3. Dual transformation

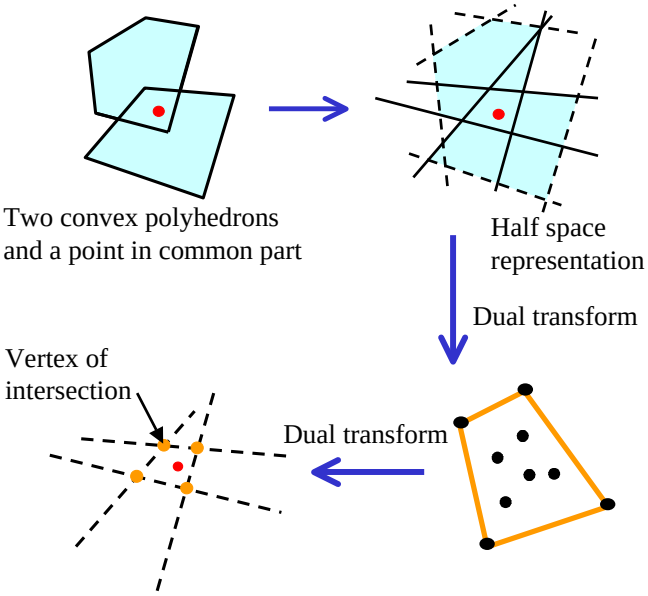

Figure 4. Common part of two convex polyhedrons

4.3. Contact normal estimation

Our simulator estimates contact normals before the calculation of penalty force. The faces of intersecting part of two convex polyhedrons A and B come from one of the convex polyhedron A or B. We sum the face normals over the faces from each convex polyhedron, regarding the area of each face. The faces are triangles or can be divided into triangles. We estimate the contact normal n as following:

$$n = \sum_{triangles\ on\ A}(\boldsymbol{p_3} - \boldsymbol{p_1}) \times (\boldsymbol{p_2} - \boldsymbol{p_1}) - \sum_{triangles\ on\ B}(\boldsymbol{p_3} - \boldsymbol{p_1}) \times (\boldsymbol{p_2} - \boldsymbol{p_1}) \quad (1)$$

($\boldsymbol{p_i}$ represents position of the vertex i on the triangle.)

4.4. Integration of the penalty

Now we got the faces of intersecting part and contact normals. Next, we integrate penalty and its moment over the intersecting part to calculate the force and torque. Because we suppose distributed linear spring-damper models, the penalties are proportional to the depth of the intersecting part. The depth of intersecting part is height of upper bound minus height of lower bound. Therefore, we can calculate integration of penalty and its moment from the integration of height of upper bound minus height of lower bound (Fig.5). In addition, the boundary consists of triangles. Therefore, we integrate the height for each triangle (Fig.6). We calculate the height of each vertex of the triangle and integrate the height over the triangle.

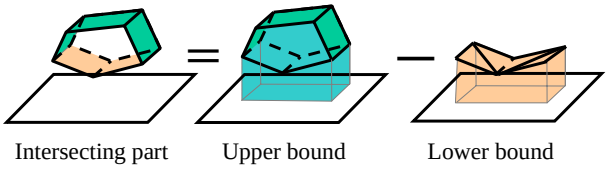

Figure 5. Upper bound and lower bound

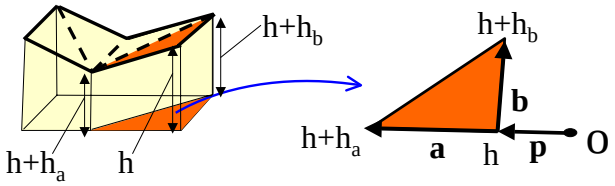

Figure 6. Triangular decomposition and notation

Following are integrations of the penalty and its moment over a triangle. p, $p+a$ and $p+b$ represents position of vertices. h, $h+h_a$ and $h+h_b$ represents height of each vertices from certain plane whose normal is parallel to contact normal (Fig.6).

$$\boldsymbol{P} = \int_{s=0}^{1}\int_{t=0}^{1-s}(h+sh_a+th_b)\,\boldsymbol{a}\times\boldsymbol{b}\,dtds \quad (2)$$

$$= (h+\frac{1}{3}h_a+\frac{1}{3}h_b)\boldsymbol{n} \quad (3)$$

$$\boldsymbol{M} = \int_{s=0}^{1}\int_{t=0}^{1-s}(\boldsymbol{p}+s\boldsymbol{a}+t\boldsymbol{b}) \times((h+sh_a+th_b)\boldsymbol{a}\times\boldsymbol{b})dtds \quad (4)$$

$$= \boldsymbol{p}\times\boldsymbol{P}+\boldsymbol{a}\times\boldsymbol{n}(\frac{1}{3}h+\frac{1}{6}h_a+\frac{1}{12}h_b) +\boldsymbol{b}\times\boldsymbol{n}(\frac{1}{3}h+\frac{1}{12}h_a+\frac{1}{6}h_b) \quad (5)$$

4.5. Setting of spring coefficient

We need penalty amount as a value, which is proportional to the penetration depth to apply spring coefficient. However the penalty $\boldsymbol{P}$ in section 4.4 is proportional to the volume of penetration. Therefore, we project the intersecting part on a plane whose normal is parallel to the contact normal and integrate projection area. Then we divide the penalty $\boldsymbol{P}$ and its moment $\boldsymbol{M}$ by the integrated area.

In the penalty method, a stiffer spring requires a smaller time step. For real-time applications, we can't choose time steps smaller than the computation time. Therefore, we choose spring stiffness from the time step and mass of the object [18]. The equation of motion of the spring mass damper system is

$$m\ddot{x}+b\dot{x}+kx=0 \quad (6)$$

The oscillation cycle is $T=2\pi\sqrt{m/k}$. To observe this oscillation, a sampling period of smaller than $T/2$ is necessary. Therefore, the time step must be at least smaller than $T/2$ for the simulation of the system.

Regarding this, we choose the spring coefficient k as:

$$m = m_1m_2/(m_1+m_2) \quad (7)$$

$$k = \frac{\pi^2 m}{(8T)^2} \quad (8)$$

where m_1 or m_2 represents mass of each object, T represents time step of the simulation.

4.6. Friction

Handling of friction is one of the most difficult problems for analytical methods. In some configuration, it becomes a NP-complete class problem [19]. On the other hand, penalty methods can easily treat the problem [20].

We can estimate the position of the application point where the penalty force acts from the penalty force and moment, . Therefore, we can choose the application point as the application point of the friction force.

We use a spring-damper model for static and dynamic frictions. When two objects contact each other initially, we put a spring-damper model to the application point of the penalty force(Fig.7-1). The anchors of the spring-damper model connect to two objects. The spring extends when the object is moved (Fig.7-2). Then we move the position of the spring-damper model to the application point of the penalty force. Then the spring gives the force to the application point of the penalty force(Fig.7-3). The procedure is repeated during the contact.

The friction force is given from this spring-damper model. If the force from the spring-damper model exceeds the limitation of the static friction ($f_{friction} < \mu_0 f_{normal}$),

we move the anchors of the spring to shrink the spring and to decrease the friction force to the dynamic friction force ($f_{friction} = \mu f_{normal}$).

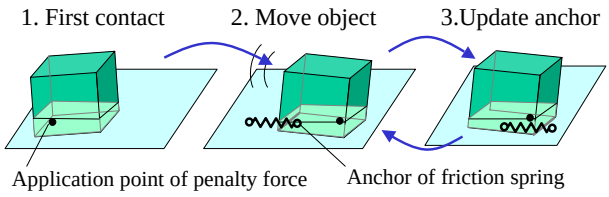

Figure 7. Procedure of the friction calculation

5. Evaluation

We do an experiment to evaluate if our simulator can treat face-face contact. Then show some example for our simulator.

5.1. Procedure of the experiment

We prepare two simulator; one is our simulator and the other is a simple simulator, which we suppose in the section section 3.1. We simulate the motion of a cube dropped on a floor by each simulator. Then, we measure angular momentum of the cube around an axis. Following is other conditions of the simulation:

- The cube gets forces only from gravity and reflection.
- The floor does not move.
- The time step is 10ms.
- The size of cube is 2mx2mx2m.
- The mass of cube is 1kg.
- The cube is located above the floor and slanted 0.1 radian.

Fig.8 shows the initial state of the virtual world.

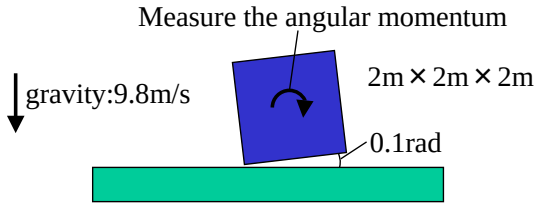

Figure 8. Simulated virtual world

5.2. Result of the experiment

The cube in our simulator stops soon. On the other hand, the cube in simple simulator vibrates and does not stop. Fig.9 shows the angular momentum of each cube. The angular momentum vibrates and does not converge on the simple simulator. On the other hand, the vibration of the cube stops after 70 steps on our simulator.

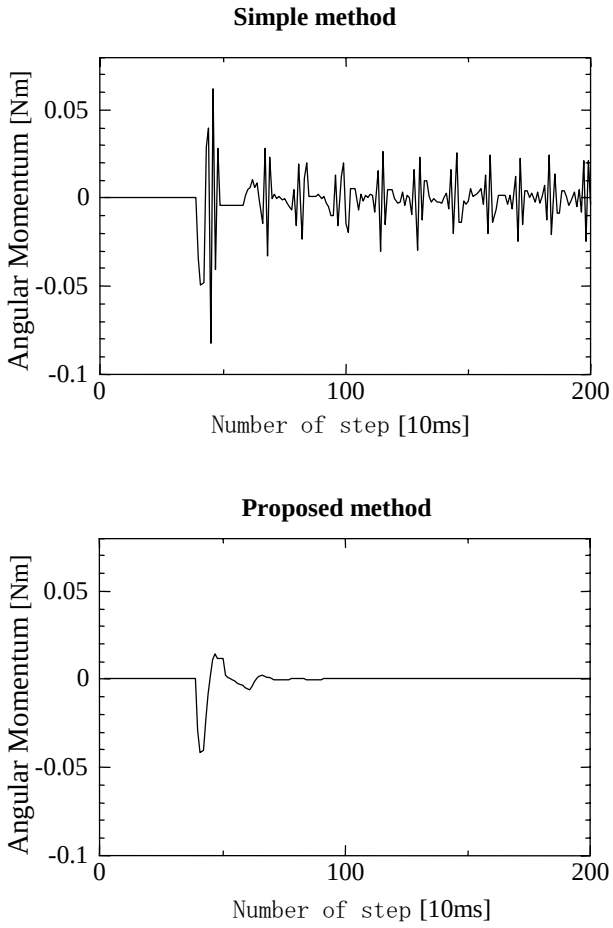

Figure 9. Angular momentum of the cube around the Z-axis

5.3. Simulation examples

Fig.10 is screen shot of simple and our simulator simulating same virtual world that have 4 blocks and a floor. The vibration of simple simulator collapses the piled block. Our simulator managed the update rate of 200Hz in this simulation on a PC with CPU of Pentium III 700MHz.

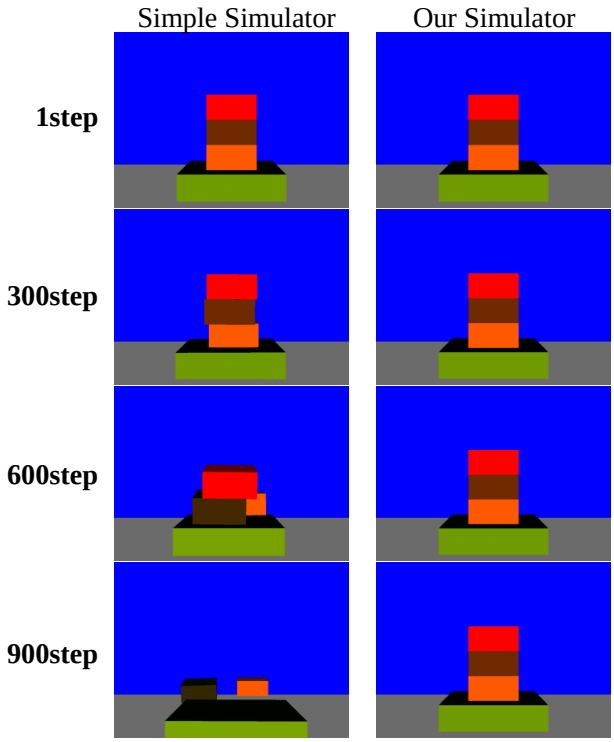

Figure 10. Simulation of piled blocks

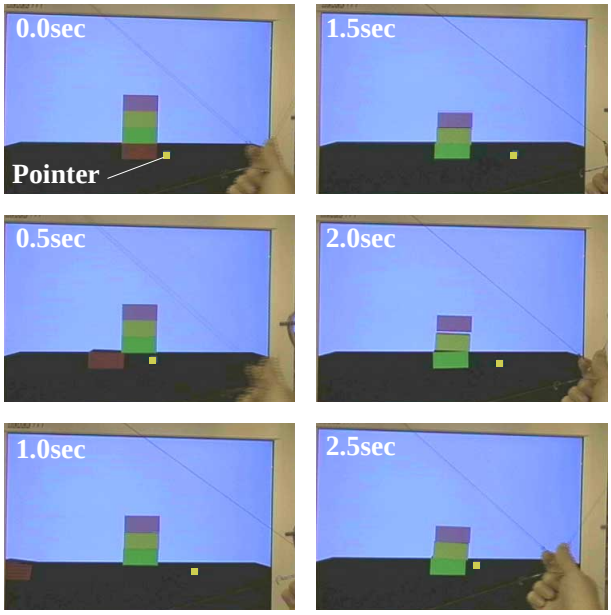

Figure 11. Interaction with haptic interface: Piled blocks

5.4. Interaction with haptic interface

We attach a string-based haptic interface [21][22] to interact with the virtual world in our simulator. We simulate two virtual environment, four piled blocks (Fig.11) and six bricks (Fig.12). In both cases, the simulation and haptic control is done at 200Hz, while graphics are rendered at 20Hz. The pointer of the haptic interface is associated to a cube in the virtual world. The posture of the cube is set to the posture of the pointer of the haptic interface. The force and torque, which the cube receives, is presented to the user via the haptic interface. Thus, the user can interact with blocks through the associated cube. Because the simulator computes both force and torque, user can feel the 6-DOF force feedbacks with a 6-DOF device.

6. Conclusion

We pointed out that previous penalty methods do not consider face-face contact and the vibrations of objects do not stop in the simple simulator, which put spring-damper model on the most penetrating point. Then we proposed a new penalty method, which integrate penetration over the intersecting part and solved the problem of previous penalty methods.

We did an experiment, which showed the effect of our method. In addition, we attached haptic interface to the simulator for interactions. We were able to interact with virtual world by haptic interface.

References

[1] A Nahvi, D. Nelson, J. Hollerbach, D. Johnson, “Haptic Manipulation of Virtual Mechanisms from Mechanical CAD Designs” Proc. IEEE International Conference on Robotics and Automation, pp.375-380, 1998

[2] D. Baraff, “Analytical methods for dynamic simulation of non-penetrating rigid bodies” Computer Graphics 23(3), pp.223-232, 1989

[3] D. Baraff, “Fast contact force computation for non-penetrating rigid bodies” Proc. SIGGRAPH 94, pp.23-34, 1994

[4] Russell L. Smith, “Intelligent Motion Control with an Artificial Cerebellum” PhD Thesis, University of Auckland, New Zealand, 1998

[5] Brian Mirtich, “Impulse-based Dynamic Simulation of Rigid Body Systems,” Ph.D. thesis, University of California, Berkeley, December, 1996

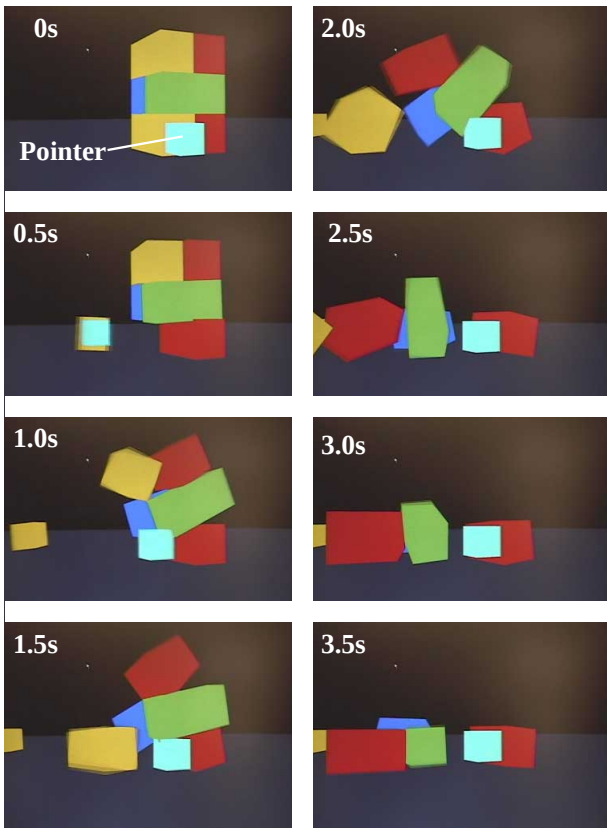

Figure 12. Interaction with haptic interface: Bricks

[6] B. Chang, J. Colgate, "Real-time Impulse-based Simulation of Rigid Body Systems for Haptic Display" Proc. ASME International Mechanical Engineering Congress and Exhibition, 1997

[7] M. McKenna and D. Zeltzer, "Dynamic simulation of autonomous legged locomotion", Computer Graphics (SIGGRAPH 90), Vol. 24, pp. 29-38, August 1990.

[8] H. Keller, H. Stolz, A. Ziegler: Virtual Mechanics : Simulation and Animation of Rigid Body Systems, http://citeseer.nj.nec.com/keller94virtual.html, 38 pages, 1993

[9] D. Terzopoulos, J. Platt, A. Barr, K. Fleischer: "Elastically deformable models", Computer Graphics, Vol. 21 (SIGGRAPH 87), pp.205-214,1987

[10] John M. Snyder, Adam R. Woodbury, Kurt Fleischer, Bena Currin, Alan H. Barr: "Interval Methods for Multi-Point Collisions between Time-Dependent Curved Surfaces" Proceedings of SIGGRAPH 1993, pp. 321-334, 1993.

[11] M. Moore, J. Wilhelms: "Collision Detection and Response for Computer Animation", Proc. SIGGRAPH 88, pp.289-298, 1988

[12] E. G. Gilbert, D. W. Johnson, S. S. Keerthi: "A fast procedure for computing the distance between complex objects in three-dimensional space" IEEE Journal of Robotics and Automation 4(2), pp.193 - 203, 1988

[13] G. van den Bergen: "A Fast and Robust GJK Implementation for Collision Detection of Convex Objects.", Journal of Graphics Tools 4(2), pp.7-25, 1999

[14] J. Cohen, M. Lin, D. Manocha, and K. Ponamgi: "I-COLLIDE: An Interactive and Exact Collision Detection System for Large-Scaled Environments", Proc. ACM Symposium on Interactive 3D Graphics, pp. 189-196, 1995

[15] B. Mirtich: "V-Clip Collision Detection Library", MERL

[16] Brian Mirtich: "Rigid Body Contact: Collision Detection to Force Computation", Workshop on Contact Analysis and Simulation, IEEE Intl. Conference on Robotics and Automation, May 1998

[17] D. E. Muller, F.P.Preparata: "Finding the intersection of two convex polyhedra", Theoretical Computer Science, 7(2), pp.217-236 1978

[18] Personal communication from T. Kano, http://cgi3.tky.3web.ne.jp/ tkano/, 2000

[19] D. Baraff: "Coping with friction for non-penetrating rigid body simulation", Computer Graphics 25(4), pp.31-40, 1991

[20] Peter R. Kraus and Vijay Kumar, "Compliant Contact Models for Rigid Body Collisions", 1997 Proc. of IEEE Intl. Conference on Robotics and Automation, April 1997, pp.1382-1387

[21] M. Ishii, M. Sato: A 3D Spatial Interface Device Using Tensed Strings, PRESENCE (MIT Press Journal), Vol.3, No.1, pp.81-86 (1994)

[22] K. Seahak, S. Hasegawa, Y. Koike, M. Sato, "Tension Based 7-DOF Force Feedback Device: SPIDAR-G", Proceedings of the IEEE Virtual Reality 2002