

# **Scientific English Education System Using Haptic Rendering**

2014

HoangMinhPhuong

**Master's Thesis**

**Scientific English Education  
Using Haptic Rendering**

**Hoang Minh Phuong (황민푸엉)**

**Department of Computer Science Engineering**

**Pohang University of Science and Technology**

**2015**

영어와 햅틱 피드백을 활용한 과학 교육

**Scientific English Education  
Using Haptic Rendering**

# **Scientific English Education Using Haptic Rendering**

by

Hoang Minh Phuong

Department of Computer Science Engineering

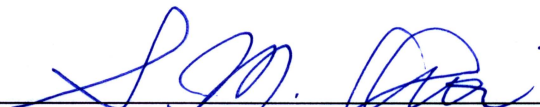
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

A thesis submitted to the faculty of  
Pohang University of Science and Technology  
in partial fulfillment of the requirements  
for the degree of Master of Science  
in the Department of Computer Science Engineering

Pohang, Korea

2014 . 12 . 16 .

Approved by

  
Seungmoon Choi, Academic Advisor

# **Scientific English Education Using Haptic Rendering**

Hoang Minh Phuong

The undersigned have examined this thesis  
and hereby certify that it is worthy of acceptance  
for a master's degree from POSTECH

12. 16. 2014

**Committee Chair**      **Seungmoon Choi (Seal)**

**Member**                **Seungyong Lee (Seal)**

**Member**                **Gary Geunbae Lee (Seal)**

MCSE 20131089 황민푸엥, Hoang Minh Phuong, 영어와 햅틱 피드백을 활용한 과학 교육,  
Scientific English Education Using Haptic Rendering,  
Department of Computer Science Engineering, 2015, 46 p,  
Advisor: 최승문 (Seungmoon Choi). Text in English

## **Abstract**

Formal language and scientific education methods have traditionally focused on visual and auditory sensory feedback to learners. The main issue of these approaches is how to encourage learners to participate actively in the learning process. Recently, multimodal sensory feedback including haptics has shown new opportunities to provide entirely different learning experiences. This research is in line with this recent research thrust aiming at learning improvement in scientific English for elementary students by using haptic feedback. We propose a scientific English education system utilizing a spoken dialogue technology and haptic force feedback rendering algorithms to teach physics phenomena. The system allows students to study in four science-related scenarios including a gravity game, a texture exploration scenario, a solar system and a kite simulation. In addition, we utilize 3-DOF and voxel-based 6-DOF haptic rendering algorithms to provide force feedback and torque feedback in these scenarios to learners to improve the realism of virtual environments. As a result, students can better understand and remember the related physics when the education involves movement and touch.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Organization . . . . .	2
<b>2</b>	<b>Haptic Rendering</b>	<b>4</b>
2.1	Haptic Rendering Method . . . . .	4
2.2	Voxel Sampling Method for 6-DOF Haptic Rendering . . . . .	7
2.2.1	Collision Detection . . . . .	7
2.2.2	Collision Response . . . . .	10
2.2.3	Rigid Body Dynamics . . . . .	13
2.2.4	Virtual Coupling . . . . .	17
2.3	Implementation in CHAI3D . . . . .	19
<b>3</b>	<b>Integration of Haptic Rendering Algorithms into Scientific English Education System</b>	<b>23</b>
3.1	Scientific English Education System . . . . .	23
3.2	Integration . . . . .	24
<b>4</b>	<b>Implementation</b>	<b>27</b>



---

4.1	Apparatus . . . . .	27
4.2	Haptic-enabled Gravity Game . . . . .	28
4.2.1	Goal and Design . . . . .	28
4.2.2	Implementation and Results . . . . .	29
4.3	Haptic-enabled Texture Exploration Scenario . . . . .	30
4.3.1	Goal and Design . . . . .	30
4.3.2	Implementation and Results . . . . .	31
4.4	Haptic-enabled Solar System . . . . .	33
4.4.1	Goal and Design . . . . .	33
4.4.2	Implementation and Results . . . . .	33
4.5	Haptic-enabled Kite Simulation . . . . .	35
4.5.1	Goal and Design . . . . .	35
4.5.2	Implementation and Results . . . . .	36
4.6	Maze Park Scenario . . . . .	38
<b>5</b>	<b>Discussion</b>	<b>40</b>
<b>6</b>	<b>Conclusion</b>	<b>41</b>
	<b>Bibliography</b>	<b>43</b>
	<b>감사의 글</b>	<b>47</b>
	<b>PUBLICATIONS</b>	<b>49</b>

# List of Figures

1.1	Scientific English Education System . . . . .	2
2.1	Illustration of 3-DOF God-object method. . . . .	5
2.2	Illustration of virtual proxy method. . . . .	5
2.3	Direct haptic rendering archituctre . . . . .	6
2.4	Simulation-based haptic rendering pipeline using virtual coupling	7
2.5	Surface voxels of a virtual gear model. Blue points represents surface voxels. . . . .	9
2.6	Interior voxels of a virtual gear model. Red points represents interior voxels. . . . .	9
2.7	Voxel tree data structure: 512-tree of a virtual gear model. . . . .	10
2.8	Pointshell representation of a virtual end-effector model. Blue dots represents the position of pointshells. Small red lines from blue points represents the inward surface normal vectors. . . . .	11
2.9	Tagent Plane force model for a Point. . . . .	12
2.10	A rigid block subject to an interaction force and torque . . . . .	15
2.11	Virtual coupling integration scheme . . . . .	18
2.12	Six-DOF haptic rendering test case of a bunny model. . . . .	19

---

2.13 Six-DOF haptic rendering test case between a gear model and the virtual haptic tool. . . . .	20
2.14 A flow of an test case implementation in CHAI3D. . . . .	22
3.1 Science room scenarios . . . . .	24
3.2 Screenshot of virtual science room. . . . .	25
3.3 Interprocess communication channels . . . . .	25
4.1 Barret WAM arm. . . . .	28
4.2 Six-DOF haptic rendering using WAM arm . . . . .	29
4.3 Haptic-enabled Gravity Game. . . . .	30
4.4 Haptic-enabled Texture Exploration Scenario. . . . .	32
4.5 Illustration of texture rendering algorithm. . . . .	32
4.6 Illustration of torque on a sphere by an interaction force . . . . .	34
4.7 Haptic-enabled Solar System. . . . .	35
4.8 Archimedes experiment scenario. . . . .	35
4.9 Illustration of forces on a kite. The center of pressure is abbreviated as CP. The center of gravity is abbreviated as CG. . . . .	37
4.10 Modelling a control line as a spring damper system. . . . .	38
4.11 Haptic-enabled kite simulation using WAM arm device. . . . .	39
4.12 Illustration of maze park for a quiz game. . . . .	39

# List of Tables

2.1 Average haptic update frequency. . . . .	20
--	----

# Chapter 1

## Introduction

### 1.1 Motivation

The concurrent use of English as a global language has brought increased interests in English education. Governments have invested huge expenses to provide more effective English education [20]. Many computer-aided English education systems have been proposed by recognizing utterances and providing educational feedback [14, 17, 23]. However, these systems can only take advantage of two sensory channels (visual and auditory) to convey information to learners. Therefore, a medium for interactive learning is necessary. Haptic feedback can provide more immersive and interactive environments to users by directly delivering innate tactual feedback to a user's body. Several researchers employed haptic feedback to develop scientific education systems [11, 12, 26]. In these systems, visual and aural information is augmented or replaced with haptic information. Hamza-Lup et al. [12] allowed students to interact with physics experiments that demonstrate static and kinetic friction. In chemistry education, haptic devices create virtual models of molecules that students can manipulate [26]. In this way, students can develop a deeper understanding of abstract concepts.

Inspired by the previous studies, we utilized multimodal sensory feedback

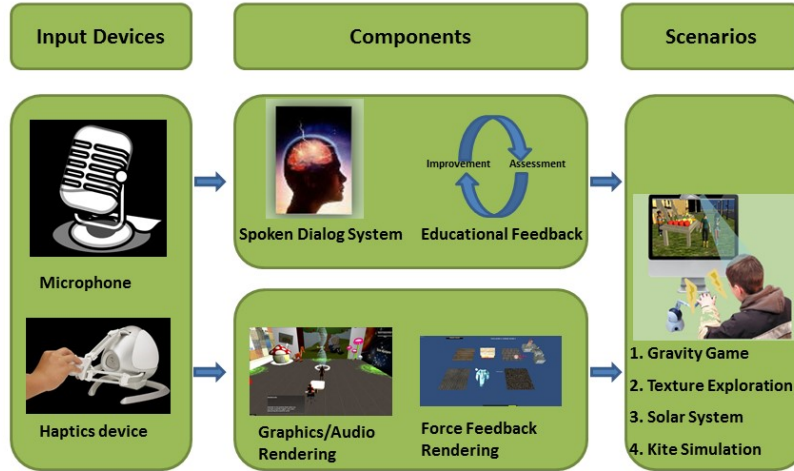


Figure 1.1: Scientific English Education System

including haptic feedback for our scientific English education system (as illustrated in Figure 1.1). Our system integrates a spoken dialogue system, graphical rendering and force feedback rendering to provide an active, intuitive and immersive learning environment to elementary students. The system allows students to study physics and English simultaneously in four haptic-enabled scenarios that are, a gravity game, a texture exploration scenario, a solar system and a kite simulation. In addition, a quiz game is designed to test the understanding of students.

## 1.2 Organization

This thesis structures as follows: the haptic force feedback rendering backgrounds are presented in Chapter 2. Integration of haptic rendering algorithms into a scientific English education is detailed in Chapter 3. the design and implementation of the four haptic-enabled scientific scenarios including gravity game, texture exploration scenario, a solar system and a kite simulation as well

as a quiz game are illustrated in Chapter 4. In Chapter 5, discussion about the developed system is provided, followed by conclusions in Chapter 6.

# Chapter 2

## Haptic Rendering

This chapter describes the background for 3-DOF and 6-DOF haptic rendering algorithms. First, Section 2.1 discusses the concept of 3-DOF and 6-DOF haptic rendering algorithms and related work. Then, section 2.2 provides details about a voxel sampling algorithm that is adopted in our 6-DOF haptic rendering implementation.

### 2.1 Haptic Rendering Method

Haptic rendering is the process of generating force and torque feedback from the interaction of haptic devices with virtual objects. In 3-DOF haptic rendering, a haptic tool is represented as a point interacting with virtual environment. Zilles et al. [28] proposed a God-object method to estimate a force vector to be exerted on the user. The term haptic interface point (HIP) is used to describe the position of the haptic device end-effector. In contrast, the term god-object is used to describe a proxy point that is not able to penetrate into polygonal models. Then, an energy minimization function is used to find the new best position of the god-object. Lastly, the resulting force is calculated by applying an ideal mass-less spring system between the god-object and HIP (as illustrated in Figure 2.1).



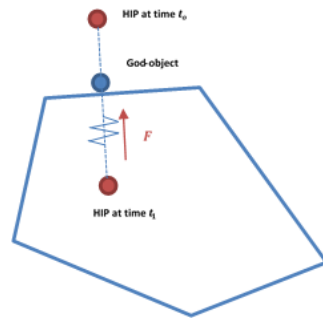


Figure 2.1: Illustration of 3-DOF God-object method.

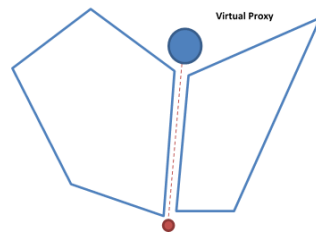


Figure 2.2: Illustration of virtual proxy method.

A drawback of god-object algorithm is that the user may get stuck inside a mesh when the polygonal models have small gaps. Ruspini et al. [25] proposed a virtual proxy method to overcome the limitation by representing of the gob-object as a sphere with a small radius (as illustrated in Figure 2.2).

However, many virtual scenarios require 6-DOF haptic rendering algorithms for interaction between more complicated objects. For example, in a haptic-enabled medical training task, the operator uses a scissor and a needle to interact with organs of virtual patients. These tools generate various types of haptic responses via force and torque feedback. Several 6-DOF haptic rendering approaches have been proposed ranging from direct rendering [10, 18] to simulation-based rendering [24, 22, 21].

In direct rendering method, the position and orientation of the haptic tool in a virtual environment is the same as the actual position and orientation of the haptic device (as illustrated in Figure 2.3). The force and torque are calculated

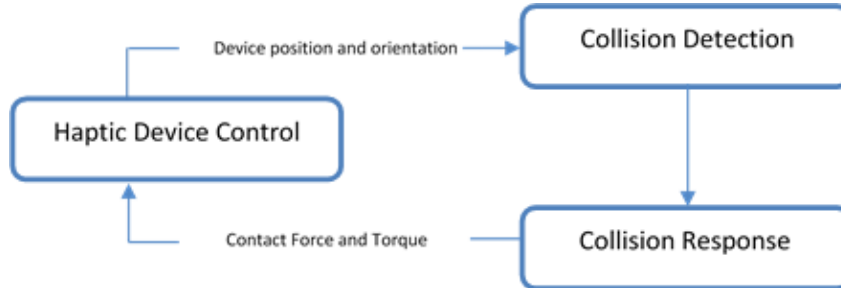


Figure 2.3: Direct haptic rendering architecture

to prevent penetration of the haptic tool into a virtual object. The advantage of direct rendering methods is that they are purely geometric. However, the drawback is the mechanical instability.

In contrast, many research groups have proposed different approaches based on simulations to generate more stable force and torque feedback by applying a virtual coupling method. In addition, the object penetration depth is typically much smaller than for direct rendering. Otaduy et al. [22] presented a stable 6-DOF haptic rendering method of complex polygonal models (ten thousands of triangles) using virtual coupling. Ortega et al. [21] extended the 3-DOF God-object method to 6-DOF using a constraint-based approach. This method prevents interpenetration and generates realistic and stable force feedback. McNeely et al. [15] proposed a voxel sampling method to represent the haptic tool and the virtual environment as different representations instead of polygonal models. This method is also extended for geometrically complex reduced deformable models [8]. Our 6-DOF haptic rendering follows the voxel sampling approach. Figure 2.4 shows the simulation-based haptic rendering pipeline using virtual coupling.

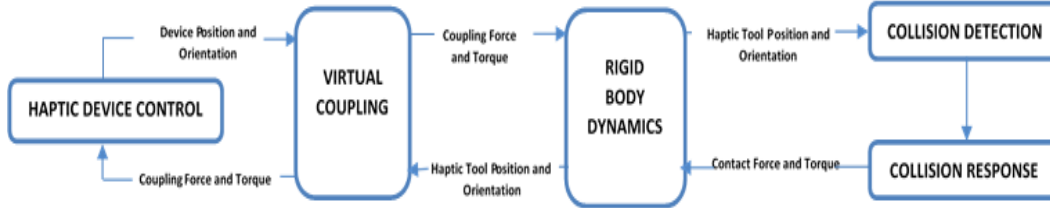


Figure 2.4: Simulation-based haptic rendering pipeline using virtual coupling

## 2.2 Voxel Sampling for 6-DOF Haptic Rendering

To the best of our knowledge, the voxel sampling method was one of the first 6-DOF haptic rendering methods applicable to commercial purposes. The main idea of the method is to represent the virtual haptic tool and the virtual environment as a set of surface points and voxels, respectively. This method is not geometrically accurate but its implementation is not excessively complex and can achieve a reasonably high haptic update rate. The voxel sampling method can consist of collision detection, collision response, virtual coupling and rigid body dynamics as illustrated in Figure 2.4.

### 2.2.1 Collision Detection

This section outlines the creation and usage of voxel-based representations of the virtual haptic tool and objects in a virtual environment. Thus, data structure generation is the first important initialization step. Then, the usage of these voxel-based structures to determine pairs of a colliding point and a colliding voxel is illustrated.

#### Voxel-based Data Structures

All objects in the virtual environment are voxelized to surface voxels and interior voxels. A voxel is defined as a 3D regular cube with a voxel size  $s$ . The voxel size determines the accuracy of the voxel-based representation. The smaller the

value of voxel size, the greater the voxel-based representation accuracy. However, as the number of voxels increases dramatically, the performance of the collision detection module decreases rapidly. To voxelize the objects in the virtual environment, we can use a flood fill algorithm. From a starting voxel (for example the center of mass of the object), the flood fill algorithm uses a queue or a stack to store the six axis-aligned neighbor voxels of the starting voxel. Then, a line between the starting voxel and a neighbor voxel is set up to determine whether the neighbor voxel is surface voxel or interior voxel by checking intersection with the object mesh. If there is an intersection between the line and the triangles of the mesh object, the neighbor voxel is marked as a surface voxel. Otherwise, it can be marked as an interior voxel. Then, the interior voxel is inserted to a queue to process its neighbors in the next step. Figure 2.5 and 2.6 illustrate the surface voxels and interior voxels found by the voxelization process. Then, these voxels are stored in a hierarchical data structure for memory efficiency. McNeely et al. [15] used a 3-level hierarchy based on a 512-tree, where leaf nodes are voxels. The voxel tree consists of axis-aligned cubical volumes in the local object frame. In our implementation, the leaf node has a number of voxels. Figure 2.7 demonstrates the voxel tree for a gear model.

In contrast, the virtual haptic tool is represented by a set of points known as pointshell with an inward pointing surface normal vector for each point. In our implementation, generating pointshell is done by using the center position of surface voxels for positions of points and using inward pointing normal vectors at pointshell position. An example is given in Figure 2.8. Then, the next step is how to detect intersection between voxel-based representations of the virtual haptic tool and objects in the virtual environment.

### Pointshell vs Voxel Tree

McNeely et al. [16] illustrated their approach for collision detection between moving objects and a dynamic virtual tool. In our implementation, determining

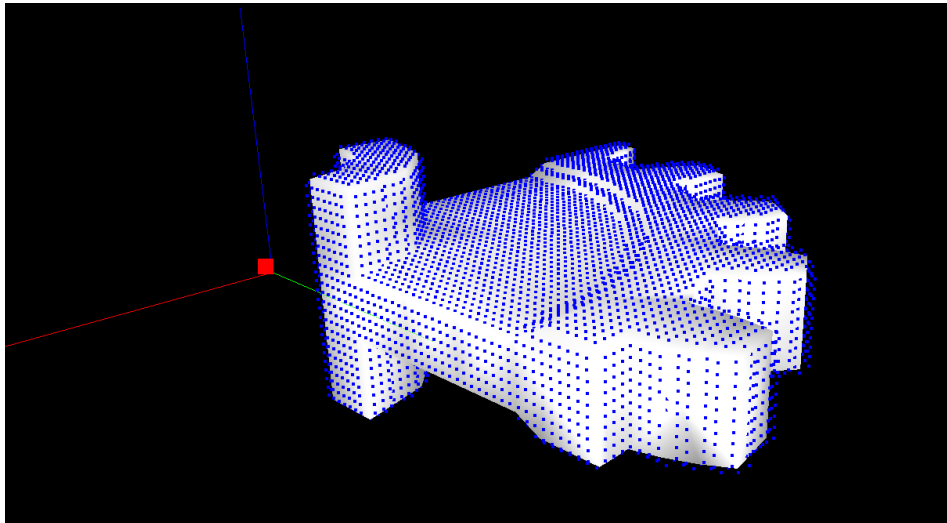


Figure 2.5: Surface voxels of a virtual gear model. Blue points represents surface voxels.

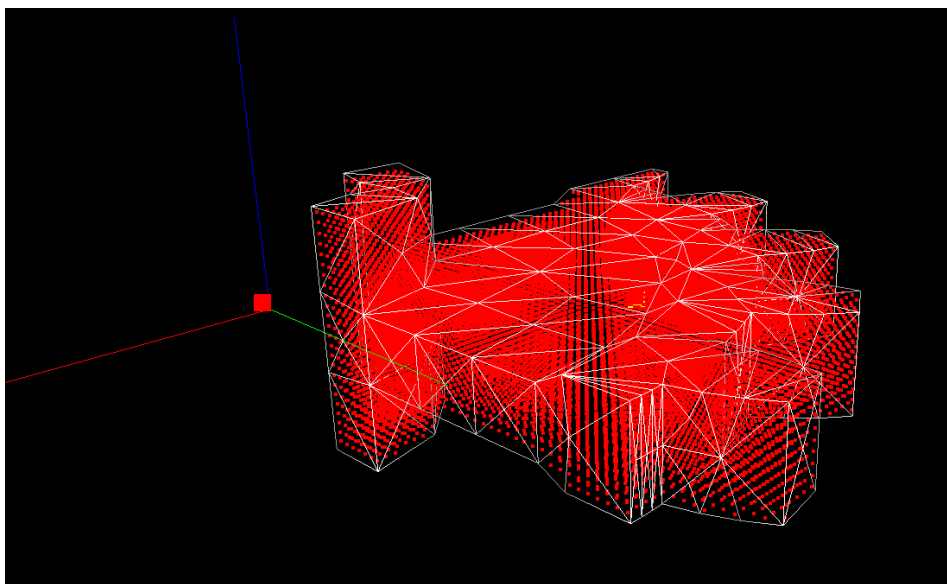


Figure 2.6: Interior voxels of a virtual gear model. Red points represents interior voxels.

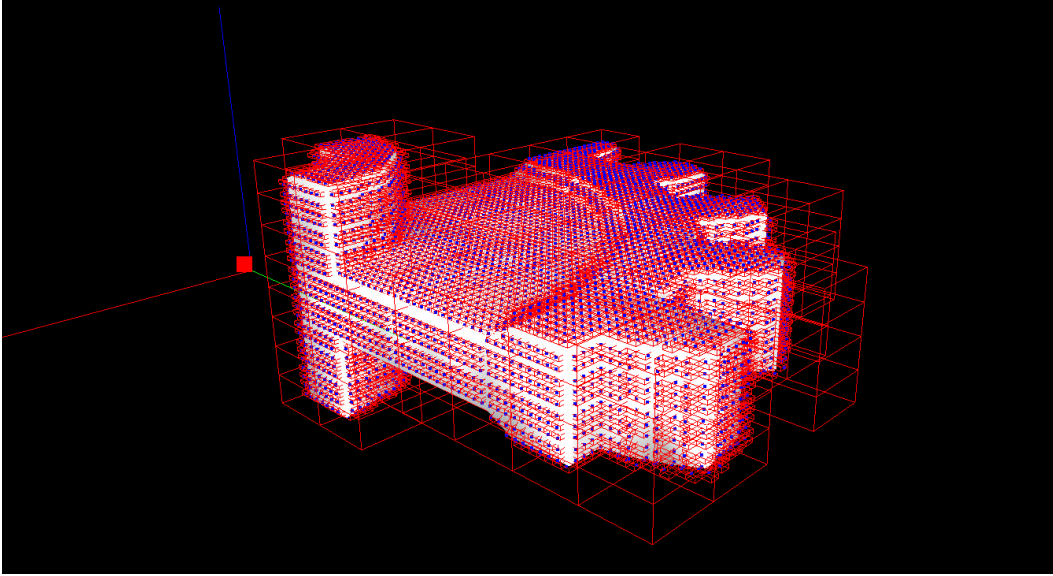


Figure 2.7: Voxel tree data structure: 512-tree of a virtual gear model.

intersection is done as follows. At every haptic cycle, all points in the pointshell are traversed linearly (point by point) to determine the contact between voxel tree and the pointshell. The positions and normal vectors of points are transformed from the local frame of the virtual haptic tool into the world frame. Secondly, an additional transformation of these values into the coordinate frame of the voxelized objects is done. Then, the position value is used to search the leaf node in the voxel tree recursively. Lastly, the voxels in the leaf node are used to determine an intersection pair of a colliding point and a colliding voxel. The great advantage of this scheme is that collision between moving objects can also be detected. However, as the number of points in the point set increases, the update rate can decrease very rapidly.

### 2.2.2 Collision Response

The goal of collision response is to calculate a contact force and a contact torque using a list of point-voxel intersections. For that McNeely et al. [15] used a tan-

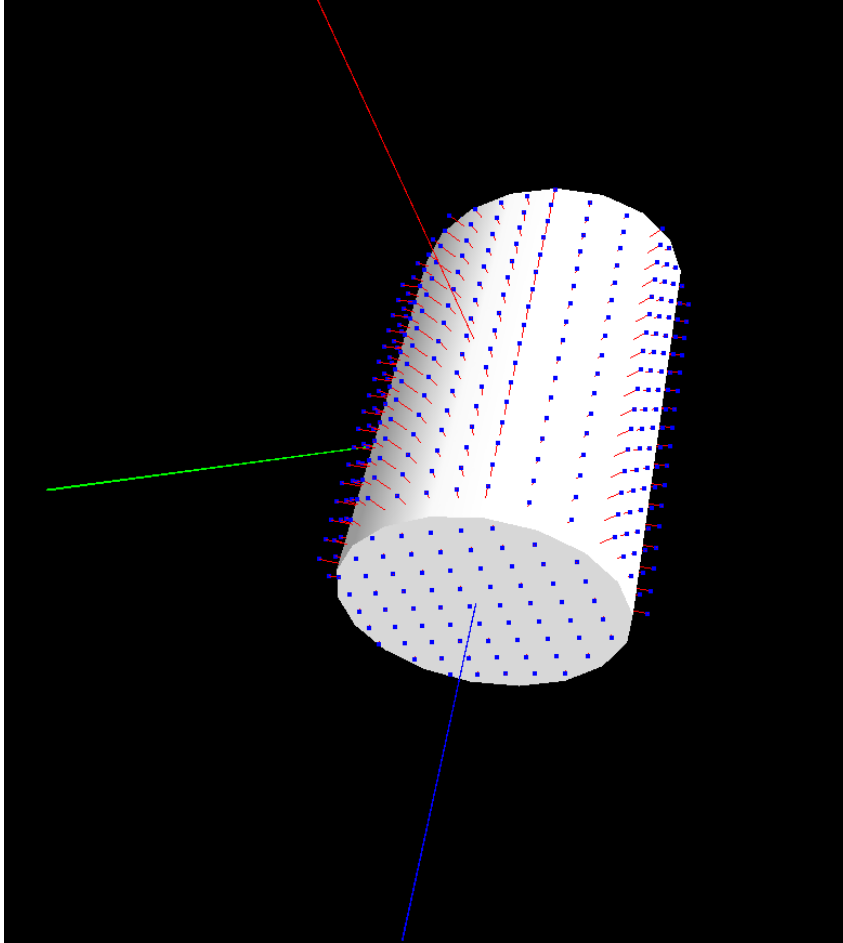


Figure 2.8: Pointshell representation of a virtual end-effector model. Blue dots represents the position of pointshells. Small red lines from blue points represents the inward surface normal vectors.

gent plane defined at the center position of the colliding voxel,  $p$  and an inward pointing surface normal vector of the colliding point,  $n$ . Then, the tangent plane force model calculates a contribution force  $F_p$  by applying a virtual spring between the colliding point and the tangent plane. The contribution force  $F_p$  is defined as:

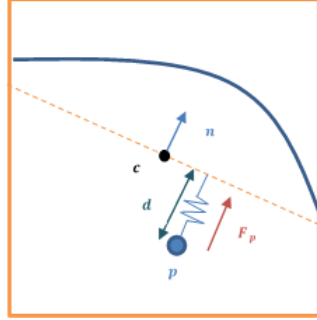


Figure 2.9: Tangent Plane force model for a Point.

$$\mathbf{F}_p = -k_s d = -k_s (n \cdot (p - c)) \quad (2.1)$$

where  $k_s$  is the stiffness of the spring and  $p$  is the position of the colliding pointshell. Figure 2.9 illustrates the tangent plane force model. The contribution force also results in a contribution torque  $\tau_p$  on the virtual haptic tool, which is defined as:

$$\tau_p = (p - x) \times \mathbf{F}_p \quad (2.2)$$

where  $x$  is the position of the center of mass of the virtual haptic tool. Thus, the total contact force and the total contact torque from all the intersection pairs are the sum of all contribution forces and contribution torques, respectively.

$$\begin{cases} \mathbf{F}_{\text{total}} = \sum \mathbf{F}_p \\ \tau_{\text{total}} = \sum \tau_p \end{cases} \quad (2.3)$$

A great benefit of this force model is simplicity. However, in the case that a large number of pointshell points collides with voxels, the total force  $\mathbf{F}$  may exceed the maximum force that can be exerted by the haptic device. McNeely et al. [15] proposed an averaging method to limit the total force when exceeding



a certain number of contacts (typically 10). Following this method, the total contact force applied to the virtual haptic tool is defined as:

$$\begin{cases} \mathbf{F}_{\text{net}} = \mathbf{F}_{\text{total}} & \text{if } n < 10 \\ \mathbf{F}_{\text{net}} = \frac{\mathbf{F}_{\text{total}}}{10} & \text{if } n \geq 10 \end{cases} \quad (2.4)$$

where  $n$  is the number of colliding points. In our work, we averaged the contact force for more than 70 colliding points.

### 2.2.3 Rigid Body Dynamics

This section describes the dynamics simulation of the virtual haptic tool using rigid body simulation. A rigid body is regarded as an object with a three-dimensional shape and a non-zero volume, and is assumed that it does not change its shape. At a particular time, the state of a rigid body is determined by four quantities: the position of its center of mass  $\mathbf{x}(t)$ , its orientation  $\mathbf{q}(t)$ , its linear velocity  $\mathbf{v}(t)$ , and its angular velocity  $\boldsymbol{\omega}(t)$ . Each of these quantities can be represented as a three-dimensional vector, except the orientation that uses a quaternion. In addition, the mass  $m$  and the moment of inertia  $I$  of the rigid body need be known. Then, under interaction of forces and torques, a new state of the rigid object in the next time should be determined by solving Newton-Euler equations of motion numerically.

#### Newton-Euler Equation of Motion

The most common way to describe the motion of a rigid body is to use Newton-Euler equations of motion. Derivations for the Newton-Euler equations of motion can be found in [7].

$$\begin{aligned} \mathbf{F} &= \frac{dp}{dt} = \frac{d(mv)}{dt} = m \frac{dv}{dt} = ma, \\ \boldsymbol{\tau} &= \frac{dL}{dt} = \frac{d}{dt}(I\boldsymbol{\omega}) = I \frac{d\boldsymbol{\omega}}{dt} = I\boldsymbol{\alpha} \end{aligned} \quad (2.5)$$

where  $a$  and  $\alpha$  are the linear and angular acceleration of the rigid body, respectively.

### Numerical Methods

Let  $\mathbf{Y}(t)$  represent the state vector of a rigid body at the time step  $t$ . The state vector  $\mathbf{Y}(t)$  is as follows:

$$\mathbf{Y}(t) = \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{q}(t) \\ \mathbf{p}(t) \\ \mathbf{L}(t) \end{pmatrix} \quad (2.6)$$

where  $\mathbf{p}(t)$  and  $\mathbf{L}(t)$  are the linear and angular momentum, respectively.

The Newton-Euler equations of motion can be rewritten in the form of the time derivative of the state vector as follows:

$$\frac{d}{dt}\mathbf{Y}(t) = \frac{d}{dt} \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{q}(t) \\ \mathbf{p}(t) \\ \mathbf{L}(t) \end{pmatrix} = \begin{pmatrix} \mathbf{v}(t) \\ \frac{1}{2}[0, \omega]\mathbf{q}(t) \\ \mathbf{F}(t) \\ \tau(t) \end{pmatrix} \quad (2.7)$$

To solve the Newton-Euler equations of motion, several numerical methods are used to estimate the state vector in the next time step  $\mathbf{Y}(t + \Delta t)$ , given the state vector at the current time step  $\mathbf{Y}(t)$ , the total interaction force  $\mathbf{F}$  and the total interaction torque  $\tau$  (see Figure 2.10).

$$(\mathbf{F}, \tau, \mathbf{Y}(t)) \rightarrow \mathbf{Y}(t + \Delta t) \quad (2.8)$$

**Explicit Integration Algorithm** One method to compute  $\mathbf{Y}(t + \Delta t)$  is to use explicit numerical integration algorithm. First, we look at the first-order Taylor expansion for  $\mathbf{Y}(t)$  around  $t$ :

$$\mathbf{Y}(t + \Delta t) = \mathbf{Y}(t) + \Delta t \dot{\mathbf{Y}}(t) + O(\Delta t^2) \quad (2.9)$$

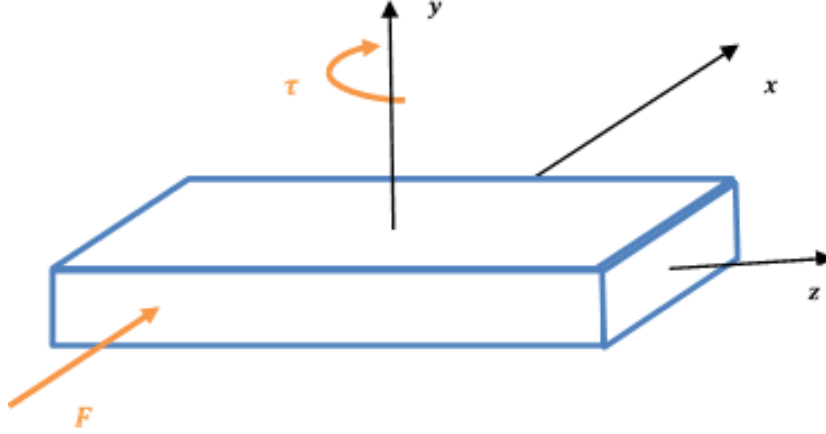


Figure 2.10: A rigid block subject to an interaction force and torque

The state vector in the next time step can be approximated from its time derivative  $\dot{\mathbf{Y}}(t)$  up to the order of precision of  $O(\Delta t)^2$ . The numerical algorithm [9] to compute the next state vector is as follows :

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \Delta t \mathbf{v}(t) \quad (2.10)$$

$$\mathbf{q}(t + \Delta t) = \mathbf{q}(t) + \Delta t \frac{1}{2} [0, \omega(t)] \mathbf{q}(t) \quad (2.11)$$

$$\begin{cases} \mathbf{p}(t + \Delta t) = \mathbf{p}(t) + \Delta t \mathbf{F} \\ \mathbf{v}(t + \Delta t) = \frac{\mathbf{p}(t + \Delta t)}{m} \end{cases} \quad (2.12)$$

$$\begin{cases} \mathbf{L}(t + \Delta t) = \mathbf{L}(t) + \Delta t \tau \\ \omega(t + \Delta t) = I^{-1} \mathbf{L}(t + \Delta t) \end{cases} \quad (2.13)$$

The algorithm is easy to implement and fast to compute, but neither very accurate or stable.

**Implicit and Semi-Implicit Integration Algorithm** Otaduy et al. [22] implemented an implicit integration for 6-DOF haptic rendering. The implicit integration

method can be defined as follows:

$$\mathbf{Y}(t + \Delta t) = \mathbf{Y}(t) + \frac{d}{dt}\mathbf{Y}(t + \Delta t)\Delta t \quad (2.14)$$

The main issue in this method is that the derivative term cannot be directly evaluated. A set of algebraic equations need to be solved. Thus, in our implementation, we use a semi-implicit integration method with reasonably stable simulation and less effort. The difference between the semi-implicit algorithm and the explicit algorithm is the order of computations. In the semi-implicit method, the new position and orientation are estimated using the new values of linear and angular velocity.

$$\begin{cases} \mathbf{p}(t + \Delta t) = \mathbf{p}(t) + \Delta t\mathbf{F} \\ \mathbf{v}(t + \Delta t) = \frac{\mathbf{p}(t + \Delta t)}{m} \end{cases} \quad (2.15)$$

$$\begin{cases} \mathbf{L}(t + \Delta t) = \mathbf{L}(t) + \Delta t\boldsymbol{\tau} \\ \boldsymbol{\omega}(t + \Delta t) = I^{-1}\mathbf{L}(t + \Delta t) \end{cases} \quad (2.16)$$

Then, the new position and orientation are computed as:

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \Delta t\mathbf{v}(t + \Delta t) \quad (2.17)$$

$$\mathbf{q}(t + \Delta t) = \mathbf{q}(t) + \Delta t\frac{1}{2}[0, \boldsymbol{\omega}(t + \Delta t)]\mathbf{q}(t) \quad (2.18)$$

**Verlet Integration Algorithm** In order to have a more accurate numerical method, our implementation also uses the Verlet integration algorithm [27]. Let's expand the two third-order Taylor expansions for the state vector  $\mathbf{Y}(t)$  at  $t + \Delta t$  and  $t - \Delta t$

$$\mathbf{Y}(t + \Delta t) = \mathbf{Y}(t) + \Delta t\frac{d\mathbf{Y}(t)}{dt} + \frac{\Delta t^2}{2}\frac{d^2\mathbf{Y}(t)}{dt^2} + \frac{\Delta t^3}{6}\frac{d^3\mathbf{Y}(t)}{dt^3} + O(\Delta t^4) \quad (2.19)$$

$$\mathbf{Y}(t - \Delta t) = \mathbf{Y}(t) - \Delta t \frac{d\mathbf{Y}(t)}{dt} + \frac{\Delta t^2}{2} \frac{d^2\mathbf{Y}(t)}{dt^2} - \frac{\Delta t^3}{6} \frac{d^3\mathbf{Y}(t)}{dt^3} + O(\Delta t^4) \quad (2.20)$$

Sum these equations together:

$$\mathbf{Y}(t + \Delta t) + \mathbf{Y}(t - \Delta t) = 2\mathbf{Y}(t) + \Delta t^2 \frac{d^2\mathbf{Y}(t)}{dt^2} + O(\Delta t^4) \quad (2.21)$$

$$\mathbf{Y}(t + \Delta t) = -\mathbf{Y}(t - \Delta t) + 2\mathbf{Y}(t) + \Delta t^2 \frac{d^2\mathbf{Y}(t)}{dt^2} + O(\Delta t^4) \quad (2.22)$$

An important advantage of this method is that its precision is proportional to  $\Delta t^4$ . The position vector in the next time step is calculated as:

$$\mathbf{x}(t + \Delta t) = -\mathbf{x}(t - \Delta t) + 2\mathbf{x}(t) + \Delta t^2 \frac{\mathbf{F}(t)}{m} \quad (2.23)$$

The quaternion for the rigid body orientation is updated using a second-order Taylor expansion.

$$\mathbf{q}(t + \Delta t) = \mathbf{q}(t) + \Delta t \dot{\mathbf{q}}(t) + \frac{\Delta t^2}{2} \ddot{\mathbf{q}}(t) + O(\Delta t^3) \quad (2.24)$$

where  $\dot{\mathbf{q}}(t)$  and  $\ddot{\mathbf{q}}(t)$  are the first and second time derivative of quaternion.

#### 2.2.4 Virtual Coupling

This section describes how to integrate the virtual coupling scheme into 6-DOF haptic rendering algorithm. The goal of virtual coupling is to stabilize force and torque feedback by connecting a virtual haptic tool to the haptic device by a spring-damper system that is composed of a linear spring-damper subsystem and a rotational spring-damper subsystem. The virtual coupling force  $\mathbf{F}_v$  and torque  $\tau_v$  of the spring-damper system can be described by the following equations:

$$\begin{cases} \mathbf{F}_v = k_l \Delta p - b_l v \\ \tau_v = k_r \Delta \theta - b_r \omega_r \end{cases} \quad (2.25)$$

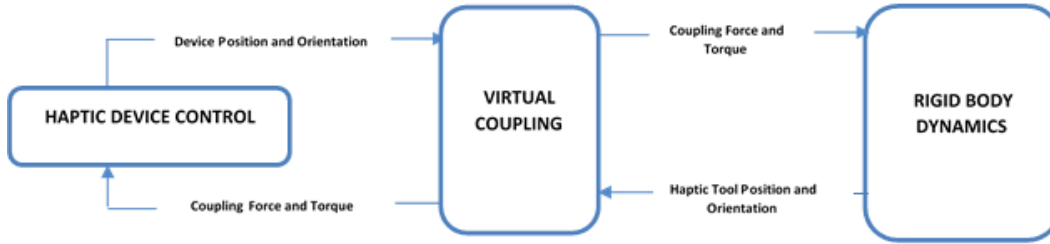


Figure 2.11: Virtual coupling integration scheme

where  $k_l$ ,  $b_l$ ,  $k_r$  and  $b_r$  represents the linear stiffness and damping constants and the rotational spring and damper, respectively.  $\Delta p$ ,  $v$ ,  $\Delta\theta$ , and  $\omega$  represents the linear and angular displacement and velocity.

For the linear spring–damper system, if the damping ratio equals 1, the system is critically damped.

$$\xi_l = \frac{b_l}{2\sqrt{mk_l}} = 1 \Rightarrow b_l = 2\sqrt{mk_l} \quad (2.26)$$

Similarly, the rotational spring–damper constants should satisfy the condition for critically damped. Otherwise, the spring–damper system may be either under-damped or over-damped. The benefit of virtual coupling is to enhance the stability of the penalty-based method. Figure 2.11 illustrates how the virtual coupling module is integrated into the 6-DOF haptic rendering algorithm. The virtual haptic tool is under constraint of two interaction forces and torques including the contact force and torque from collision response and the virtual coupling force and torque. Then, the new position and orientation of the virtual haptic tool are computed from the rigid body simulation module. The virtual haptic tool should remain outside the surface voxels while the actual haptic device may be penetrated into the surface and interior voxels.

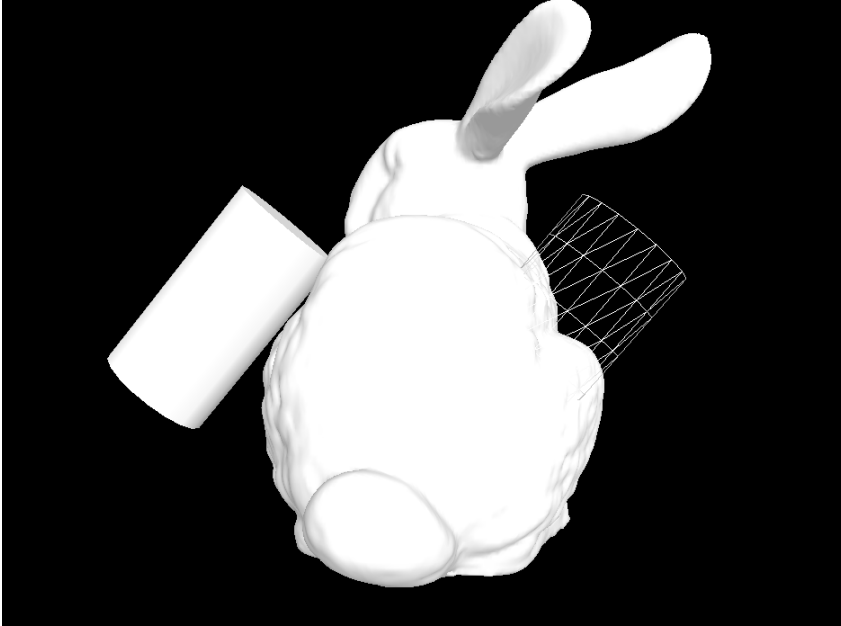


Figure 2.12: Six-DOF haptic rendering test case of a bunny model.

### 2.3 Implementation in CHAI3D

The voxel-based 6-DOF haptic rendering algorithm is implemented in CHAI3D, a open source C++ API to supporting a variety of haptic devices [2], to render force and torque feedback to a user. First, 3-D polygonal models are loaded to a virtual environment. Then, these 3D models are voxelized to construct voxel-based data structures. The main thread then uses a graphic thread and a haptic thread with a minimal update frequency of 1000 Hz (as illustrated in Figure 2.14). Several test cases of our implementation with a gear model and a bunny model are illustrated in Figure 2.13 and 2.12. The wire-frame cylinder represents the actual position and orientation of the WAM arm end-effector. The cylinder model represents the position and orientation of the virtual haptic tool. Although the haptic device penetrates into the gear model, the virtual haptic tool remains outside the gear model.

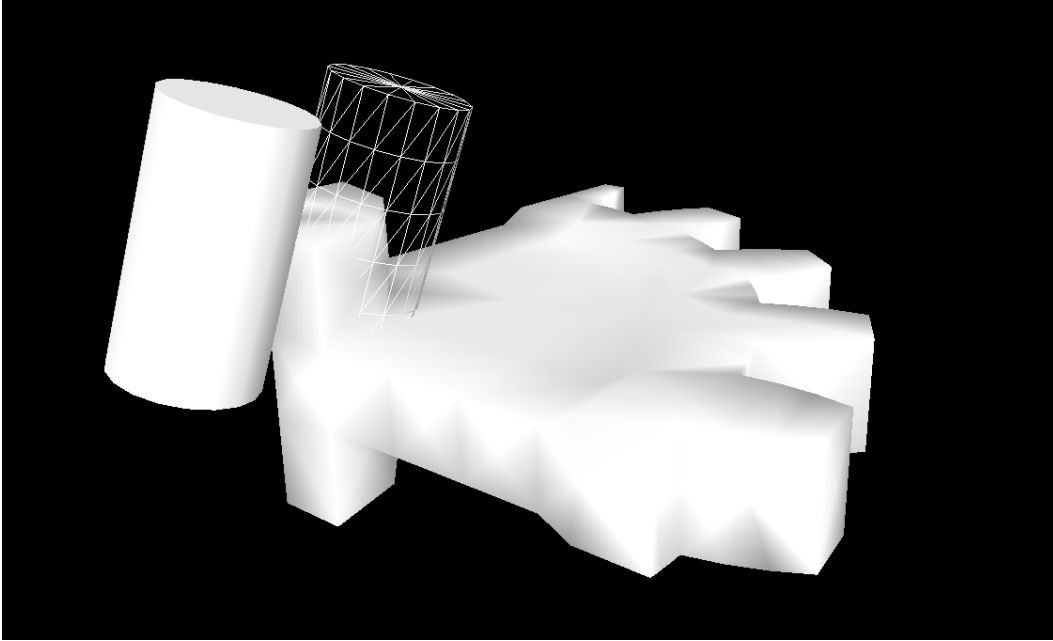


Figure 2.13: Six-DOF haptic rendering test case between a gear model and the virtual haptic tool.

**Table 2.1** Average haptic update frequency.

Number of points(voxelized resolution)	depth = 1	depth = 2
No = 614 (20x20x20)	8.5kHz	15.0kHz
No = 1454(30x30x30)	5.1kHz	8.0kHz
No = 2489(40x40x40)	2.5kHz	4.4kHz
No = 7782(70x70x70)	1.0kHz	1.5kHz

Table 2.1 summarizes the average haptic update frequencies with different number of points in a pointshell and different depths of the voxel tree in a test case with a gear model. The gear polygonal model was voxelized at a resolution of  $80 \times 80 \times 80$ . In every haptic cycle, all points in the pointshell are analyzed to determine collisions with the voxel tree. Thus, the haptic update frequency depends on the number of points in the pointshell. In particular, as the number of points in pointshells increases approximately twice, the update frequency



---

decreases almost twice with the same 512-tree ( $depth = 2$ ). In addition, the haptic update rate also depends on the depth of the voxel tree. In the case of tree  $depth = 1$  and  $depth = 2$ , the update rates were significantly different. In particular, the update rate of a voxel tree with  $depth = 2$  was approximately 1.6 times faster than the haptic update rate of a voxel tree with  $depth = 1$ . The main reason for this is that the number of voxels in a leaf node increases with a small tree depth. Thus, a point in the pointshell should be checked for collision with a larger number of voxels.

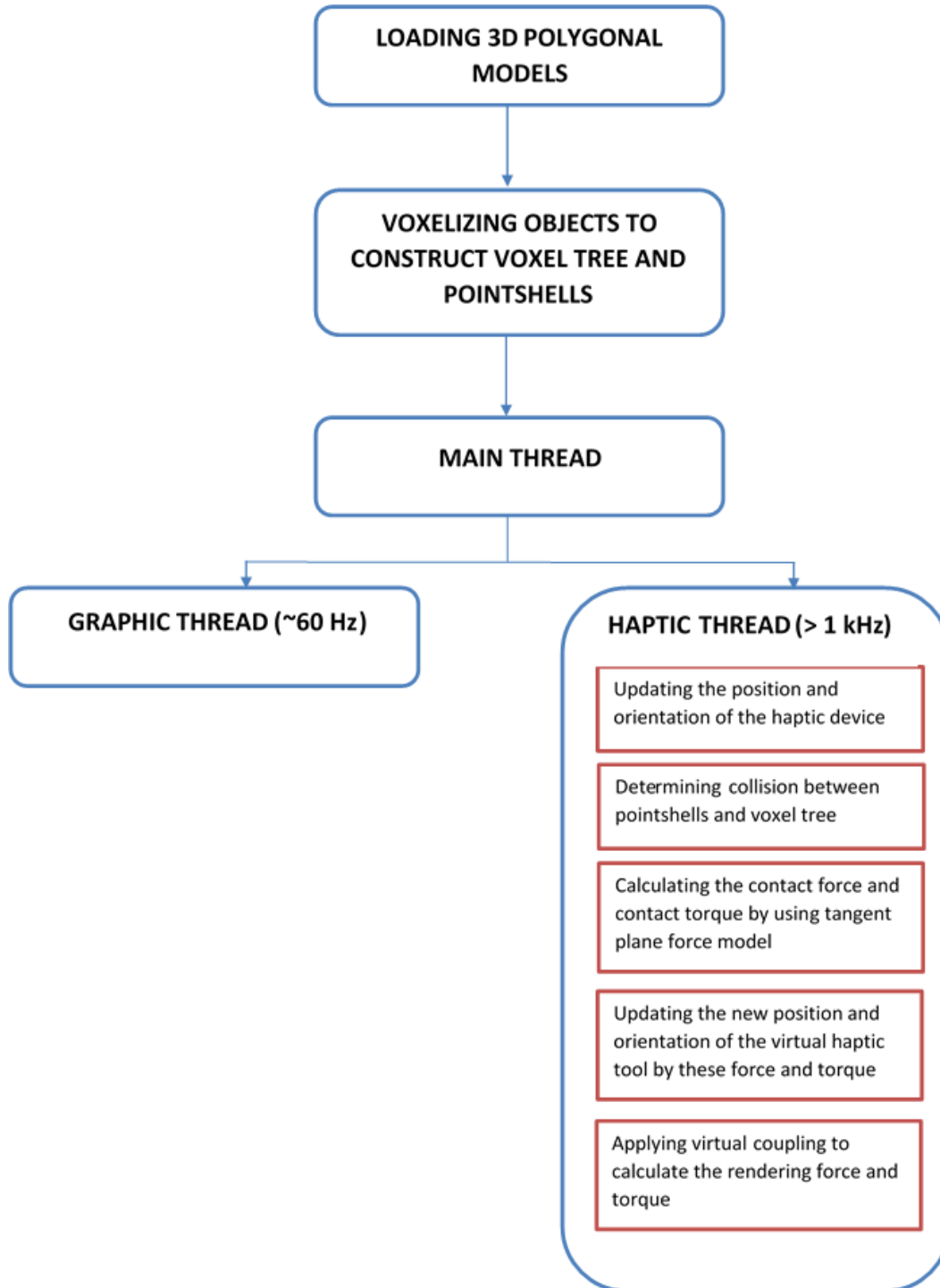


Figure 2.14: A flow of an test case implementation in CHAI3D.

# Chapter 3

## Integration of Haptic Rendering Algorithms into Scientific English Education System

In this chapter, first, the scientific English education system is presented in Section 3.1. Then, the integration of the haptic rendering algorithms described in Chapter 2 into the scientific English education system is described in Section 3.2.

### 3.1 Scientific English Education System

Our scientific English education system is a part of a project called POSTECH Immersive English Study (POMY) [19] that allows students to exercise their English skills with visual, aural and tactual feedback. Learners can increase their memory and concentration abilities to a greatest extent.

Our system consists of a spoken dialogue system and haptic rendering algorithms to provide an interactive, intuitive and immersive learning environment. First, our system motivates learners to be interactive by recognizing utterances and providing educational feedback with voice in specific mission-

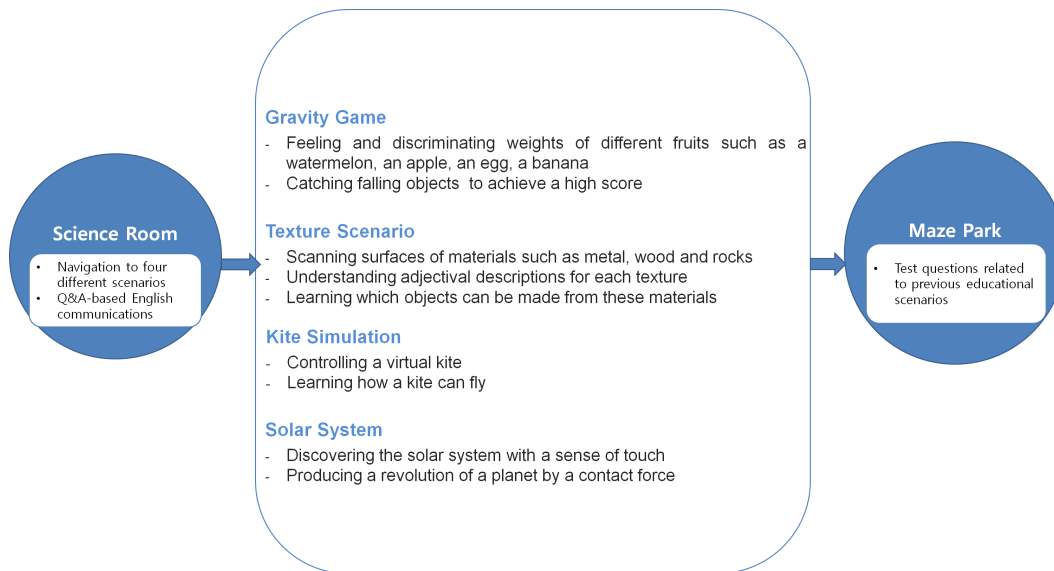


Figure 3.1: Science room scenarios

oriented scenarios. In addition, users would use a microphone to communicate with the system to get more information about English and physics.

Unlike the conventional English education systems, the system leverages force feedback technology to enable learners to interact with virtual objects to understand the basic concepts regarding the gravity law, texture of objects, force and torque of moving objects, and the solar system. A virtual science room (as shown in Figure 3.2) is constructed with the purpose of navigating to the four science-related scenarios. Detailed design and implementation of these scenarios are presented in the Chapter 4. Lastly, to revise the understanding of students, a maze park is designed for a quiz game. Figure 3.1 summarize the scenarios available in our science room.

## 3.2 Integration

Our project relies on multiprocessing model to convey haptic feedback, and graphics and audio rendering to students. The haptic rendering process is im-



Figure 3.2: Screenshot of virtual science room.

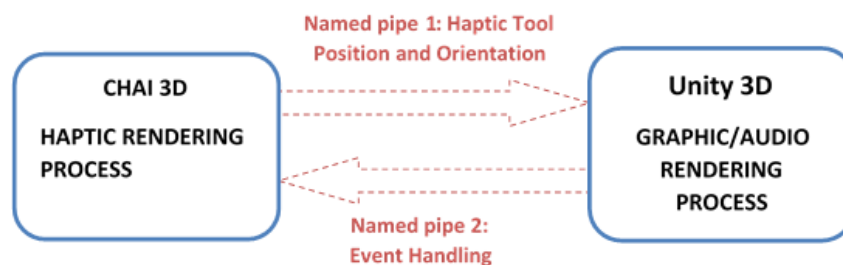


Figure 3.3: Interprocess communication channels

plemented in CHAI3D. In contrast, a game engine, Unity3D, is used to render virtual scenarios graphically. These processes communicate with each other through interprocess communication channels called Named Pipes [3]. A named pipe is a special kind of FIFO file on the local hard drive to allow two or more processes to read from or write to. We use two named pipes to synchronize graphics rendering and event handling between the CHAI3D process and a process constructed from Unity3D. The first named pipe is to send the position and orientation of the end-effector of our haptic device from CHAI3D to the Unity3D-based process for graphic rendering purposes. The second one from

Unity3D to CHAI3D is to send the controlling events e.g. switching scenarios. Using two named pipes enabled our program to avoid communication conflicts. These communication channels run at 60Hz. Figure 3.3 depicts the two communication channels.

# Chapter 4

## Implementation

In this chapter, the design and implementation of the haptic-enabled scenarios are provided.

### 4.1 Apparatus

The voxel-based 6DOF haptic rendering algorithm were implemented in CHAI3D. The computer running our haptic rendering implementation was equipped with a Intel Core-i7 3.4GHz CPU and 8 GB RAM. Then, the force feedback and torque feedback were rendered using a 7-DOF WAM arm device [6] (as shown in Figure 4.1) that is controlled by a Linux-based computer connected to the haptic rendering computer via internet connection with a UDP protocol. The position and orientation of the end-effector of the WAM arm are sent to our haptic rendering computer. Then, the rendering force and torque are calculated to be sent back to the WAM arm. Figure 4.2 shows the WAM arm working with a CHAI3D application.

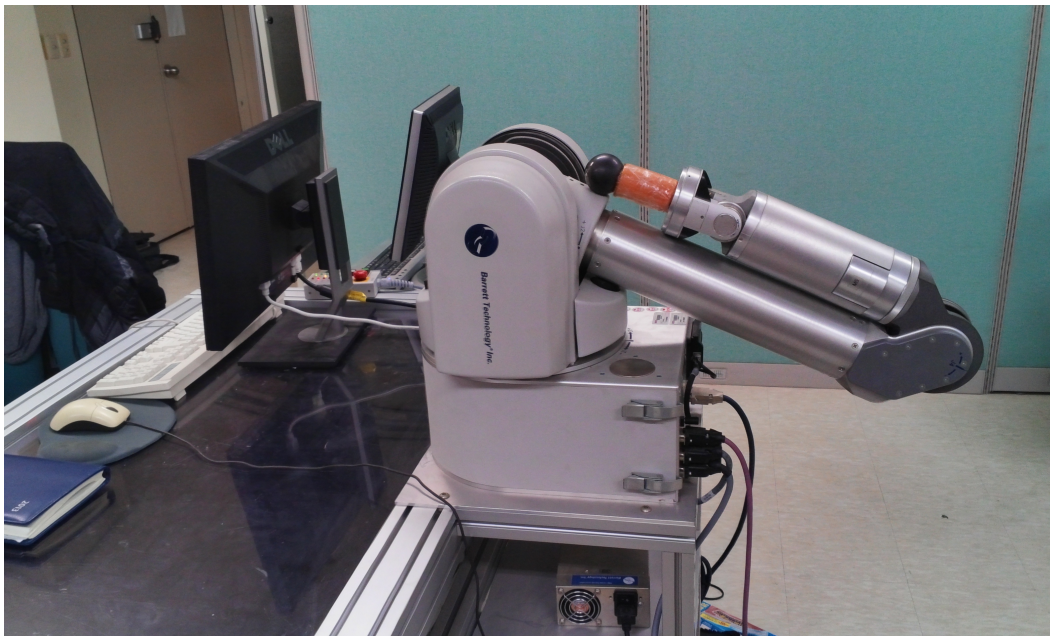


Figure 4.1: Barret WAM arm.

## 4.2 Haptic-enabled Gravity Game

### 4.2.1 Goal and Design

The goal of this scenario is to enable elementary students to understand the law of gravity in physics and to learn comparison sentences and superlatives form in English. The main issue in this scenario is the explanation about force for elementary students who do not have basic knowledge about vector. In this game, a learner uses a haptic device to control a virtual bucket to catch the falling objects such as a watermelon, an apple, a banana, and an egg. Thus, the learner can perceive the gravitational force of these objects. In terms of English perspective, after the game, our system asks students to use comparison sentences and superlatives. For example, if the system asks “What is the heaviest object?”, then a learner can answer “A watermelon is the heaviest object”. Therefore, students can learn both English conversation and the law of gravity.



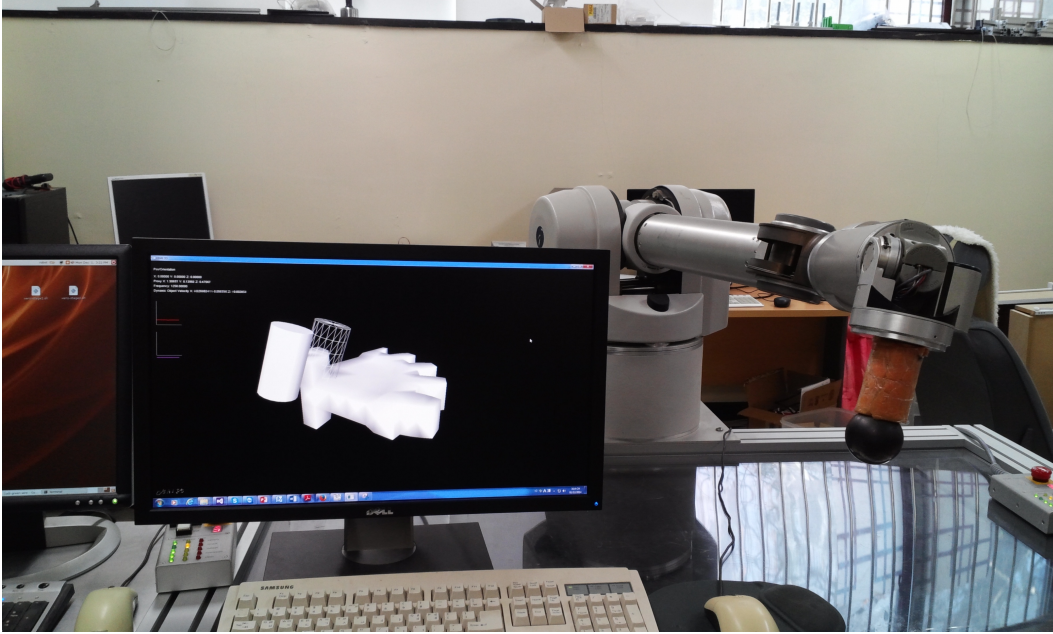


Figure 4.2: Six-DOF haptic rendering using WAM arm

#### 4.2.2 Implementation and Results

In this scenario, our system uses 3-DOF haptic rendering to convey the force to learners. A student can perceive a gravitational force:

$$\mathbf{F} = m \cdot g \quad (4.1)$$

where  $g$  is the gravitational acceleration and  $m$  is the mass of an object. For the purpose of distinguishing the weights of objects rather than realistic rendering purpose, the weights of objects are exaggerated. For example, an apple has a weight of 600 mg that means 6N in gravity force or an egg has a weight of 300 mg that means 3N in gravity force. Thus, the student would be able to discriminate the weights of these objects. Figure 4.3 illustrates the 3-DOF haptic rendering gravity game.



Figure 4.3: Haptic-enabled Gravity Game.

## 4.3 Haptic-enabled Texture Exploration Scenario

### 4.3.1 Goal and Design

The goal of this scenario is to encourage learners to study adjectives and passive voice sentences in English and to feel the texture properties of various materials. Learning English adjectives is not easy task for children in non-English speaking countries. The main reason is that most adjectives are abstract terms. Therefore, an intuitive way to learn these terms is to experience the same situations as native speakers. Our texture exploration scenario enables learners to feel the textures of various materials such as metal, wood and rock using a haptic interface. When students explores virtual objects, our system explains verbally the characteristics of the material. Thus, learners would express the perceived tactile sensation using appropriate adjectives, for example, “Metal is hard and smooth”. Lastly, various objects are made of these material, can be interacted by learners. Then, a student would understand how to make a passive voice sentences such as “A spaceship is made of metal”. Figure 4.4 illustrates

the texture exploration scenario.

### 4.3.2 Implementation and Results

In our implementation, we use an image-based haptic texturing method [13]. First, a texture field is constructed from 2-D image data including rock, wood and metal image file. Then, we map these images onto 3-D polygonal object. Then, the height value  $h$  for any point on the object surface can be obtained. The gradient of the height field  $\nabla h$  is computed using the central differences approximation for partial derivatives as follows:

$$\frac{\partial h}{\partial x} = \frac{h_{x+\varepsilon} - h_{x-\varepsilon}}{2\varepsilon} \quad (4.2)$$

$$\frac{\partial h}{\partial y} = \frac{h_{y+\varepsilon} - h_{y-\varepsilon}}{2\varepsilon} \quad (4.3)$$

$$\frac{\partial h}{\partial z} = \frac{h_{z+\varepsilon} - h_{z-\varepsilon}}{2\varepsilon} \quad (4.4)$$

$$\nabla h = \frac{\partial h}{\partial x} \hat{i} + \frac{\partial h}{\partial y} \hat{j} + \frac{\partial h}{\partial z} \hat{k} \quad (4.5)$$

Once the local gradient is known, it is employed to perturb the surface normal vector at the collision point as follows:

$$\mathbf{M} = \mathbf{N} - \nabla h + (\nabla h \mathbf{N}) \mathbf{N} \quad (4.6)$$

Then, by using both the original surface normal  $\mathbf{N}$  and the perturbed surface normal  $\mathbf{M}$ , we estimated the force feedback for image-based texture rendering as follows:

$$\mathbf{F} = \begin{cases} (d - Kh)\mathbf{N} + Kh\mathbf{M} & \text{if } d \geq Kh \\ d\mathbf{M} & \text{if } d < Kh \end{cases} \quad (4.7)$$

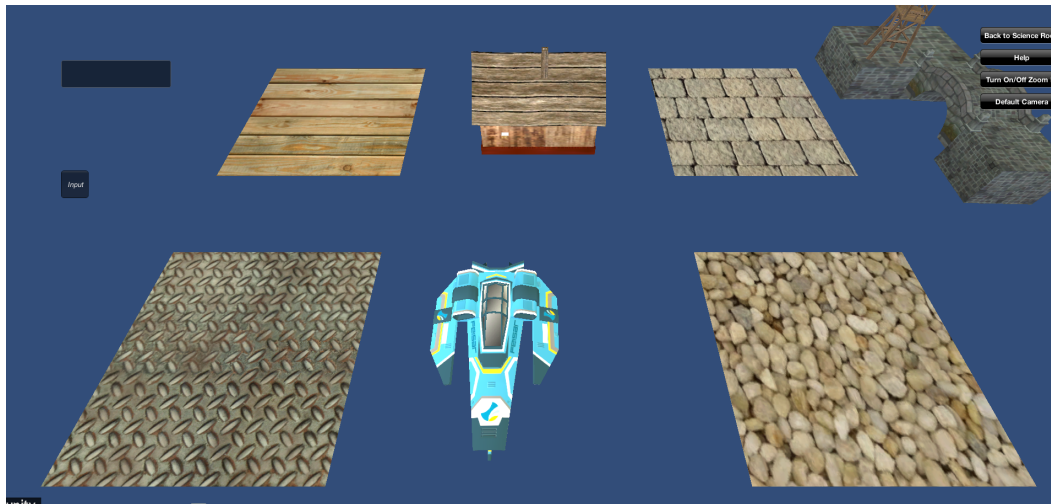


Figure 4.4: Haptic-enabled Texture Exploration Scenario.

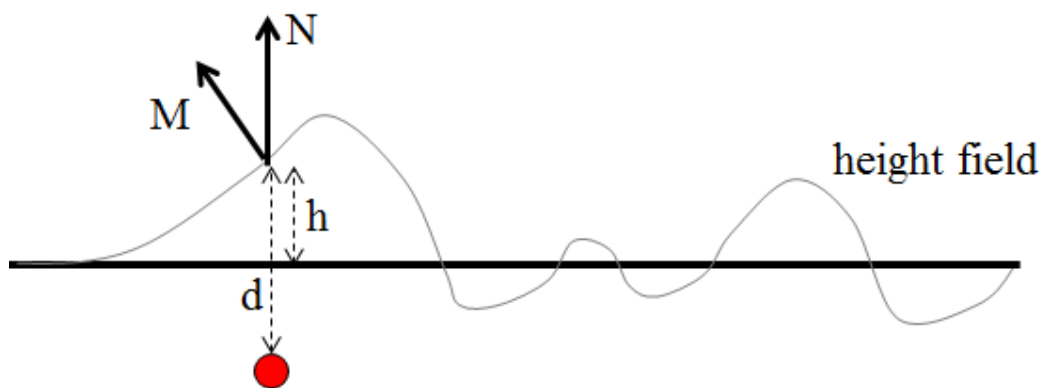


Figure 4.5: Illustration of texture rendering algorithm.

where  $F$  represents the rendering force,  $d$  represents the penetration depth,  $K$  represents a predefined-scalar that depends on the properties of texture. In practice, for metal and rock,  $K$  should be set close to 2 and 1, respectively. Figure 4.5 illustrates the texture rendering algorithm.

## 4.4 Haptic-enabled Solar System

### 4.4.1 Goal and Design

The haptic-enabled solar system introduces a basic concept of torque to elementary students. In addition, learners can gather specific information on space objects like planets and moons by verbally explanations from a virtual assistant and explore their surface using our haptic device.

### 4.4.2 Implementation and Results

Our solar system are constructed virtually with the sun and eight planets rotating around the sun and itself. Users would visit these planets in a view mode or a haptic mode. In the view mode, a close-up, constant view of the planet is presented to learners so that they can visually observe detailed information about each planet. In addition, if a student has a question about the planet, he/she would use a headphone to communicate with the virtual assistant. For example, a student would ask “How many moons does the earth have?”. Then, the virtual assistant can answer “Our earth has one moon”.

In the haptic mode, we use our voxel-based 6-DOF haptic rendering implementation to convey force and torque feedback to learners. The planet is regarded as a sphere that the user can use a virtual haptic tool to rotate it and explore its surface. The interaction force from learners causes the revolution of the planets around its axis. In addition, if a student want to know what is torque, Figure 4.6 is used to describe torque intuitively. We simply explain torque,  $\tau$  equals the radius of the planet  $R$  multiplied by tangential force  $\mathbf{F}_{\text{tangential}}$ :

$$\tau = R \cdot \mathbf{F}_{\text{tangential}} \quad (4.8)$$

Figure 4.7 illustrates the haptic-enabled solar system using the WAM arm.

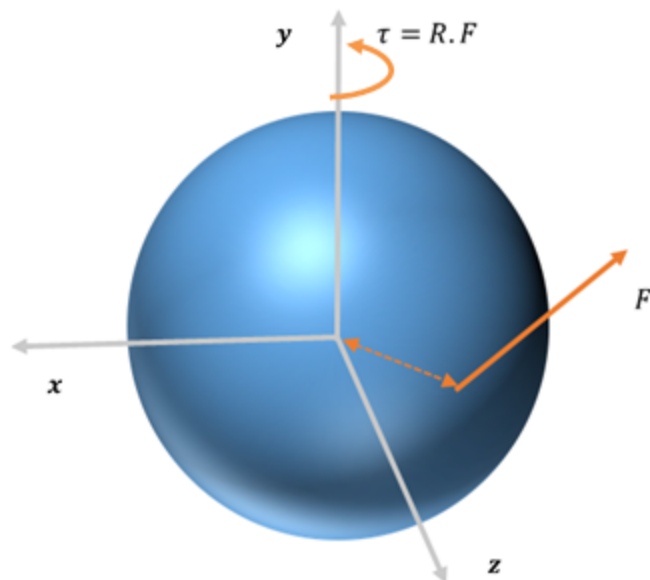


Figure 4.6: Illustration of torque on a sphere by an interaction force

In addition, students can learn about torque by Archimedes experiment sub-scenario that is a part of our haptic-enabled solar system. Inspired by Archimedes quote “Give me a lever and a place to stand and I shall move the world” [1], an Archimedes experiment scenario is designed to convey the concepts of torque to learners. By using a haptic device, users would be able to move a virtual cylinder. Then, the virtual cylinder can interact with the virtual lever to move the Earth (as shown in Figure 4.8). The equilibrium state of the lever depends on two interaction torques including a torque caused by gravitational force of the Earth and a torque caused by interaction force between the virtual cylinder and the virtual lever. We use our voxel-based 6-DOF haptic rendering implementation to render the force and torque feedback to users.

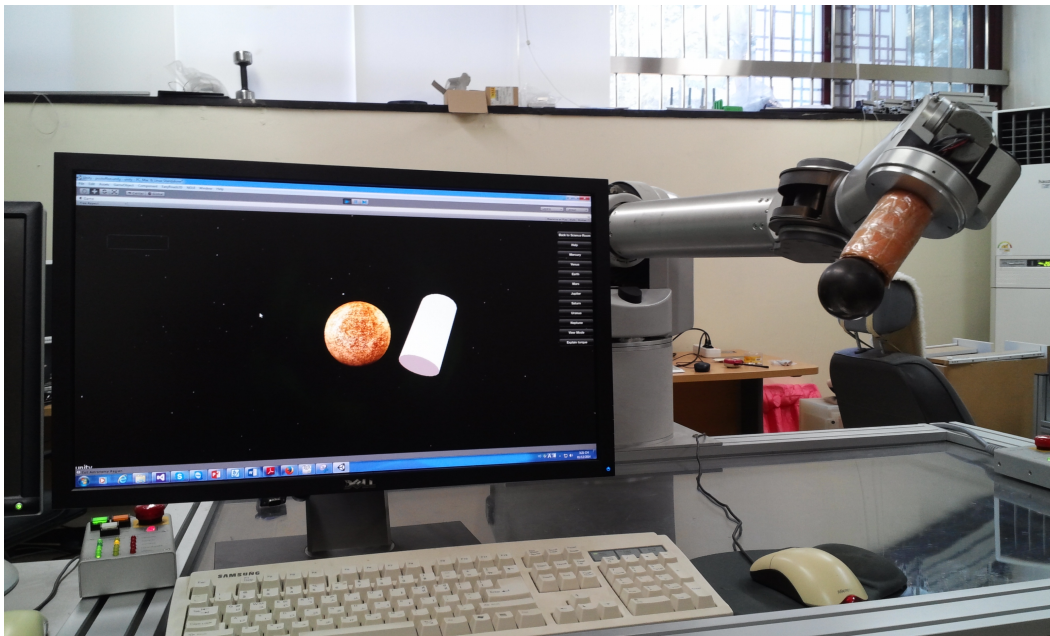


Figure 4.7: Haptic-enabled Solar System.

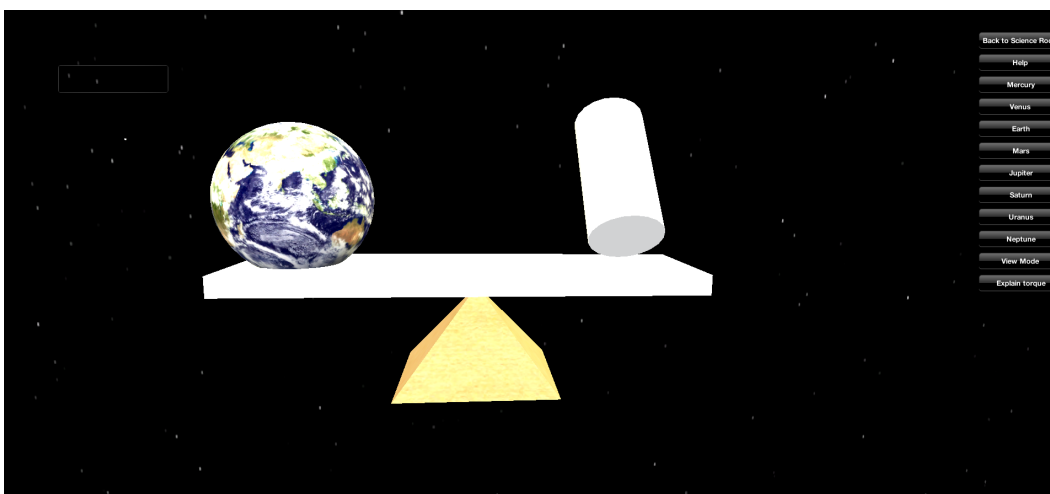


Figure 4.8: Archimedes experiment scenario.

## 4.5 Haptic-enabled Kite Simulation

### 4.5.1 Goal and Design

In this scenario, we aim to teach students about forces exerting on a object. As part of such effort, here, our scenario is designed to simulate a flying kite on an

ideal environment. Students are able to use a haptic device to control the flying kite. As learners can perceive the force and torque feedback in controlling the kite, our system would explain to them how many forces act on a kite and how a kite would fly.

#### 4.5.2 Implementation and Results

NASA proposed a 2-D flying kite simulation known as KiteModeler for educational purposes [4]. The program allows students to change the shape, size and materials of kites. Then, learners would know the stability of their designs. Our flying kite scenario extends the theory of flying kites to 3-D simulation where elementary students would control their kites using a haptic device. In addition, the students would perceive the tension force and torque in the control line between the kite and a kite holder. First, a flying kite simulation is presented. In the ideal environment, a flying kite is simulated as a dynamic object. The forces acting on a kite include a gravitational force of the kite  $\mathbf{G}$ , the aerodynamic force from wind  $\mathbf{A}$ , and the tension force  $\mathbf{T}$  in the control line (see Figure 4.9). A unique feature of our flying kite simulation is that a kite is rotating around its pivot point instead of its center of mass. Therefore, the orientation of a flying kite is independent to the tension force from users. Derivation of kite geometry, forces and torques on a kite, balance of a kite are described in [5]. In our implementation, we simply implemented a flying kite as a rigid body. The Verlet integration algorithm was used to estimate the new position and orientation of the kite:

$$S(\mathbf{G}, \mathbf{A}, \mathbf{T}, \tau, \mathbf{Y}(t)) \rightarrow \mathbf{Y}(t + \Delta t) \quad (4.9)$$

where  $\mathbf{Y}(t)$  represents the state vector of the kite,  $\tau$  represents the total torque caused by the aerodynamic force and the gravitational force on a kite. Secondly, the control line between the kite holder and a kite is simulated as a virtual linear spring–damper system (as illustrated in Figure 4.10). The main



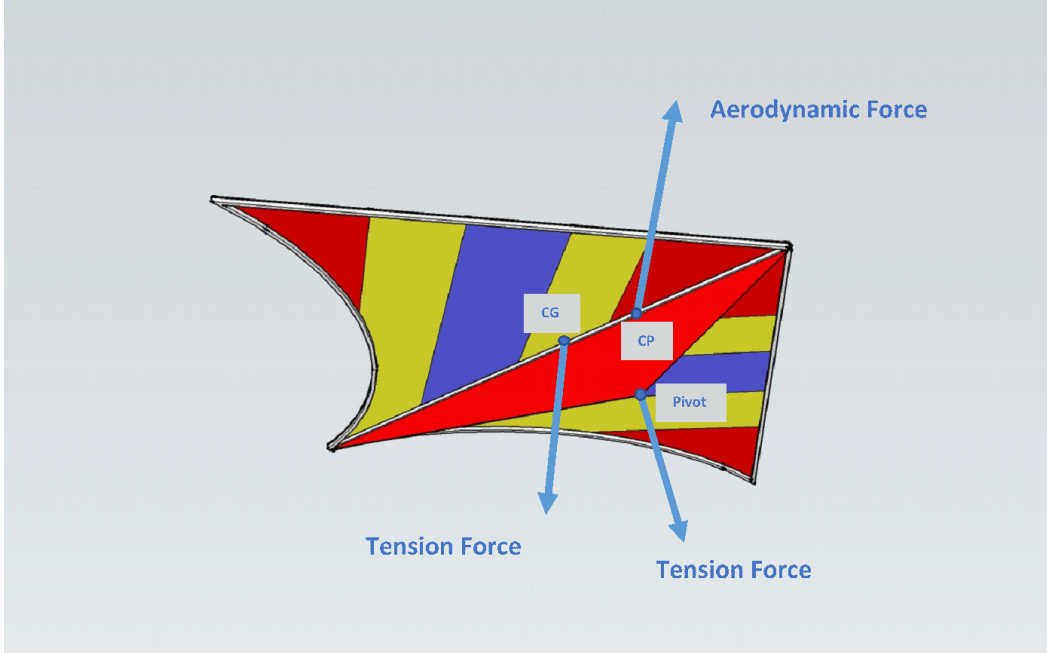


Figure 4.9: Illustration of forces on a kite. The center of pressure is abbreviated as CP. The center of gravity is abbreviated as CG.

reason of our implementation is to simply demonstrate the tension force and torque to elementary students instead of realistic rendering. Thus, the force  $\mathbf{F}$  and torque  $\tau$  exerting on users are defined as follows:

$$\begin{cases} \mathbf{F} = k_l \Delta x - b_l v \\ \tau = (p - COM) \times \mathbf{F} \end{cases} \quad (4.10)$$

where  $k_l$  and  $b_l$  represents the stiffness and damping constants of the linear spring and damper;  $\Delta x$  and  $v$  represents the linear displacement and the relative linear velocity between the kite holder and the kite;  $p$  represents the position where the the force exerting on the kite holder and  $COM$  represents the center of mass of the kite holder. Figure 4.11 shows our kite simulation with the WAM arm device.

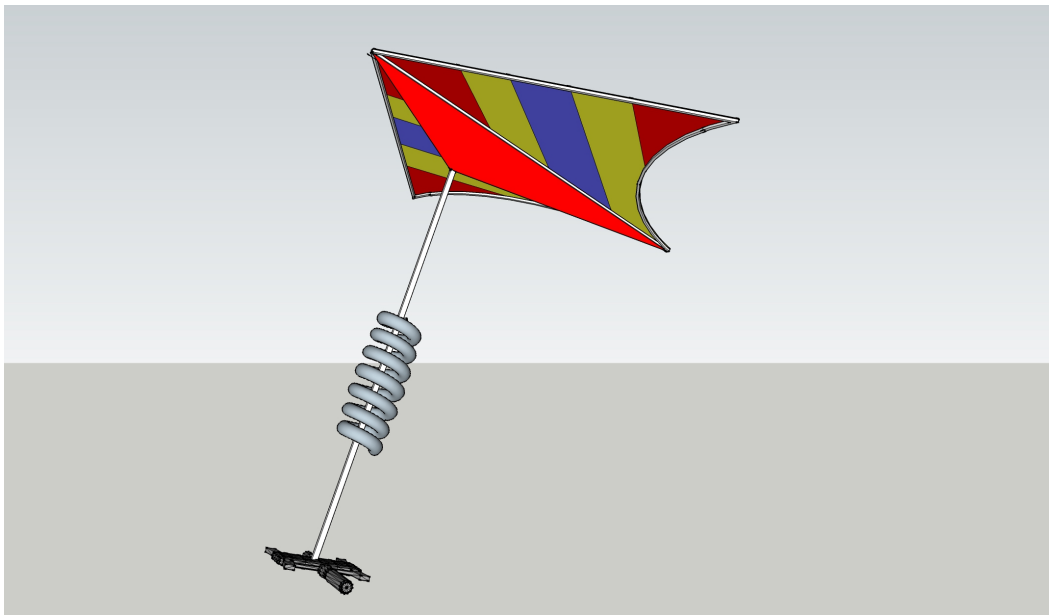


Figure 4.10: Modelling a control line as a spring damper system.

## 4.6 Maze Park Scenario

The main goal of this scenario is to enable students to revise their understanding of the above concepts in an entertaining way. A maze park is designed with a virtual character. Users should use a keyboard to navigate the virtual character to get out the maze. Then, if the virtual character encounters a dead-end region, our system asks a randomly selected question related to the knowledge learned. For example, the system would ask “How many planets in our solar system?”. Then, by using a microphone, the student would answer to the question. The answer is recognized to determine whether it is true or false. If the answer is wrong, the learner is not able to move their virtual characters to anywhere. A variety of questions are used to clarify the understanding and the abilities of using English to answer. Figure 4.12 shows the maze park.



Figure 4.11: Haptic-enabled kite simulation using WAM arm device.



Figure 4.12: Illustration of maze park for a quiz game.

# Chapter 5

## Discussion

The performance of our system depends on haptic rendering algorithms. The implementation of the 6-DOF voxel-based method showed that the first issue is the haptic update frequency which is inversely proportional to the number of points in the pointshells. The reason for this is that at every haptic cycle, our algorithm traverses the pointshell linearly (point by point) to determine the contact between a voxel tree and the pointshell. The maximal number of points that fits the computational power of our computer is approximately 8,000. The haptic update frequency can be enhanced in many ways. First, McNeely et al. [16] proposed to rely on temporal coherence to track and predict the status of points in the pointshell to speed up collision detection. An alternative approach is to use GPU to speed up collision detection.

In addition, another problem of our implementation is the stability of the dynamic simulation for the virtual haptic tool. The problem can be improved by utilizing a multi-rate approach [22]. By decoupling the collision detection from haptic thread, the stability of the force and torque rendering can be greatly improved.

# Chapter 6

## Conclusion

In this thesis, a scientific English education system was developed by utilizing multimodal sensory feedback to contribute to better learning of physics and English for elementary students. The concept of physics such as gravity law, texture of objects, force and torque on a moving object, and the solar system can be conveyed to students intuitively with haptic feedback. In English education perspective, comparison sentences and superlatives; passive voices, and adjectival descriptions can be taught intuitively in the gravity game and the texture scenarios. To test the understanding of students, a quiz game was designed and implemented in the maze park.

In all scenarios, learners would be able to communicate with our system with a microphone to get more information about physics phenomena and English. A student would ask a question related to these concepts. Then, all subsequent explanations are given with visual and auditory feedback.

In addition, we exploited haptic force feedback rendering algorithm to encourage learners to be more involved in the learning process. In 3-DOF haptic rendering, an image-based texture rendering method is used to render texture objects. In addition, the force and torque feedback in 6-DOF haptic rendering are utilized to demonstrate the concept of force and torque acting on a moving

object through the solar sytem and the kite simulation scenario. The 6-DOF voxel-based method is described in detail for all computation steps of including: collision detection, collision response, rigid body simulation, and virtual coupling.

In the future, we plan to help students perceive haptic feedback in different subjects such as biology and chemistry. In addition, we plan to collect comments about the system and accommodate them in order to upgrade the learning scenarios. Furthermore, we plan to conduct a formal human-subjects experiment to validate the effectiveness of our system.

# Bibliography

- [1] Archimedes quotes. <http://en.wikiquote.org/wiki/Archimedes>. Accessed: 2014-12-04.
- [2] Chai 3d. <http://www.chai3d.org/>. Accessed: 2014-12-04.
- [3] Named pipes. <http://msdn.microsoft.com/en-us/library/windows/desktop/aa365590> Accessed: 2014-12-04.
- [4] Nasa's kitemodeler. <http://www.grc.nasa.gov/WWW/k-12/airplane/kiteprog.html>. Accessed: 2014-12-04.
- [5] Theory of flying kites. <http://www.grc.nasa.gov/WWW/k-12/airplane/kite1.html>. Access: 2014-12-04.
- [6] Wam arm specification. <http://www.barrett.com/robot/products-arm-specifications.htm>. Accessed: 2014-12-04.
- [7] D. Baraff. Physically based modeling: Rigid body simulation. *SIGGRAPH Course Notes, ACM SIGGRAPH*, 2001.
- [8] J. Barbic and D. James. Six-dof haptic rendering of contact between geometrically complex reduced deformable models. *Haptics, IEEE Transactions on*, 2008.

- 
- [9] K. Erleben, J. Sporring, K. Henriksen, and K. Dohlman. *Physics-based Animation (Graphics Series)*. Charles River Media, Inc., Rockland, MA, USA, 2005.
- [10] A. Gregory, A. Mascarenhas, S. Ehmann, M. Lin, and D. Manocha. Six degree-of-freedom haptic display of polygonal models. In *Visualization 2000. Proceedings*, 2000.
- [11] D. I. Grow, L. N. Verner, and A. M. Okamura. Educational haptics. In *AAAI Spring Symposium: Semantic Scientific Knowledge Integration*, pages 53–58, 2007.
- [12] F. G. Hamza-Lup and W. H. Baird. Feel the static and kinetic friction. In *Lecture Notes on Computer Science (Eurohaptics 2012, Part I)*, volume LNCS 7282, pages 181–192, 2012.
- [13] C. Ho, C. Basdogan, and M. Srinivasan. Efficient point-based rendering techniques for haptic display of virtual objects. *Presence: Teleoperators and Virtual Environments*, 8(5):477–491, 1999.
- [14] W. L. Johnson, N. Wang, and S. Wu. Experience with serious games for learning foreign languages and cultures. In *SimTecT Conference*, 2007.
- [15] W. A. McNeely, K. D. Puterbaugh, and J. J. Troy. Six degree-of-freedom haptic rendering using voxel sampling. *Proc. ACM SIGGRAPH '99*, pages 401–408, 1999.
- [16] W. A. McNeely, K. D. Puterbaugh, and J. J. Troy. Voxel-based 6-dof haptic rendering improvements. In *Haptics-e*, volume 3, 2006.
- [17] H. Morton and M. A. Jack. Scenario-based spoken interaction with virtual agents. *Computer Assisted Language Learning*, 18(3):171–191, 2005.



- [18] D. D. Nelson, D. E. Johnson, and E. Cohen. Haptic rendering of surface to-surface sculpted model interaction. In *Proc. of ASME Dynamic Systems and Control Division*, 1999.
- [19] H. Noh, K. Lee, S. Lee, and G. G. Lee. Pomy: A conversational virtual environment for language learning in postech. In *Proceedings of the SIGDIAL 2011 Conference*, 2011.
- [20] D. Nunan. The impact of English as a global language on educational policies and practices in the Asia-Pacific region. *TESOL Quarterly*, 37(4):589–613, 2003.
- [21] M. Ortega, S. Redon, and S. Coquillart. A six degree-of-freedom god-object method for haptic display of rigid bodies. In *Proc. of IEEE Virtual Reality Conf.*, 2006.
- [22] M. Otaduy and M. Lin. Stable and responsive six-degree-of-freedom haptic manipulation using implicit integration. In *Eurohaptics Conference, 2005 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2005. World Haptics 2005. First Joint*, 2005.
- [23] R. Reis and P. Escudeiro. Educational software to enhance English language teaching in primary school. In *2011 International Conference on Information Technology Based Higher Education and Training (ITHET)*, pages 1–4, 2011.
- [24] D. Ruspini and O. Khatib. A framework for multi-contact multi-body dynamic simulation and haptic display. In *Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE / RSJ International Conference on*, 2000.
- [25] D. C. Ruspini, K. Kolarov, and O. Khatib. The haptic display of complex graphical environments. *SIGGRAPH '97*, pages 345–352, 1997.

- 
- [26] M. Sato, X. Liu, J. Murayama, K. Akahane, and M. Isshiki. A haptic virtual environment for molecular chemistry education. In *Lecture Notes on Computer Science (Transactions on Edutainment I)*, volume LNCS 5080, pages 28–39, 2008.
- [27] L. Verlet. Computer "experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Phys. Rev*, 159:98–103, 1967.
- [28] C. Zilles and J. Salisbury. A constraint-based god-object method for haptic display. In *Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, volume 3, pages 146–151, 1995.

# 감 사 의 글

First of all, I would like express my sincere gratitude to my official advisor *Prof. Seungmoon Choi* for his wonderful guidance and support throughout the two years. His clarity of thought and excellent intuition about different kinds of research topics have been a great source of inspiration to me. The calm and persistent attitude towards research, and the ability to come up with the right questions, are qualities I have admired greatly. One simply could not wish for a better advisor.

Additionally, I also would like express my sincere gratitude to Prof. Seungyong Lee and Prof. Gary Geunbae Lee for their enthusiasm support. Their heartfelt advise in the research

I wish to thank the seniors, labmates and alumni in Haptics and Virtual Reality Lab, POSTECH for their stimulating discussions and the wonderful time we had together. In addition, I would like to express my gratitude to Intelligent Software Laboratory for their support in my thesis. To Mr. Kyusong Lee from Intelligent Software Lab, thank you for your help and your patience with all of my bugs. To my Vietnamese friends, I want to say thank you a lot, from the joyful to the hopeless moments when I was living in POSTECH. I wish you guys the best.

Last, and most important, I would like to thank my family for their affection and support throughout my stay at Postech. This thesis is to my father who sacrificed his life to encourage me to study more and more.

# Publications

1. **Hoang Minh Phuong**, Jaebong Lee, Hojin Lee, Kyusong Lee, Gary Geunbae Lee, and Seungmoon Choi, "Haptic-enabled English Education System," In *Proceedings of the Asia Haptics*, 2014.
2. Jaebong Lee, Kyusong Lee, **Hoang Minh Phuong**, Hojin Lee, Gary Geunbae Lee, and Seungmoon Choi, "POMY: POSTECH Immersive English Study with Haptic Feedback," *Journal of Institute of Control, Robotics and Systems*, Vol. 20, No. 8, pp. 815-821, 2014.
3. Kyusong Lee, Jaebong Lee, Nohkyung Lee, Chiyoung Lee, **Hoang Minh Phuong**, Sangdo Han and Hojin Lee, "POMY (POSTECH Immersive English Study)," Demonstrated in *the HCI Korea, 2014 (Winner of the best HCI KIDS award)*.