Stable haptic rendering in virtual environment

Hou, Xiyuan

2014

Hou, X. (2014). Stable haptic rendering in virtual environment. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/61783>

<https://doi.org/10.32657/10356/61783>

Stable Haptic Rendering in Virtual Environment

Hou Xiyuan

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement of the degree of
Doctor of Philosophy

2014

Abstract

Haptics refers to the science of perception and manipulation of objects in virtual environments. Its applications spread rapidly from a human-computer interface to manufacturing, scientific discovery, medical training, etc. In a complex dynamic virtual environment, it is important to have smooth and realistic haptic feedback. In this project, we focus on research and development of stable haptic rendering methods and algorithms to provide continuous force and torque feedback in dynamic virtual environments.

In haptic rendering, many algorithms and methods were proposed such as the god-object method, spring-damper method, virtual proxy method, Voxmap Point Shell (VPS) method, constraint-based method, Quasi-Static Approximation (QSA) method, etc. Currently, for six degrees-of-freedom (6-DOF) haptic rendering, the direct haptic rendering methods only support geometric rendering without physically based dynamic simulation. Virtual coupling based methods separate the haptic device from the virtual tool. It enables high stable force feedback and supports dynamic simulation of the virtual objects with physical properties. Although these algorithms have greatly improved performance of haptic rendering, there are still unsolved and challenging problems as follows. 1) Buzzing. If a virtual tool has physically based properties (for example, mass), the buzzing would appear as continuous high frequency vibrations. 2) Inaccurate manipulation. When the virtual tool has a large mass value, the displacement would become larger because of the gravity. This large displacement would introduce an inaccurate movement during the haptic manipulation that can cause accuracy problems. 3) Discontinuous force update. When there are complex models and/or deformable models, the physical simulation may produce a low update rate of force which causes discontinuous force output on the haptic device.

The aim of the research is to propose general haptic rendering algorithms to improve stability of haptic rendering systems. To improve stability of haptic rendering, we

propose new stable haptic rendering algorithms based on virtual coupling. The algorithms can be used for various static or dynamic applications to provide stable haptic force and torque feedback. First, we propose a stable dynamic algorithm based on virtual coupling for 6-DOF haptic rendering. It can overcome the “buzzing” problem when a virtual tool with small mass values is used. The novelty is that a nonlinear force/torque algorithm is proposed to calculate the haptic interaction when the collision happens between the virtual tool and virtual objects. The force/torque magnitude is automatically saturated to the maximum force/torque value of the haptic device. The algorithm is tested on the standard benchmarks and outperforms available algorithms such as spring-damper algorithm and QSA algorithm. Experimental results show that this algorithm is capable to provide stable 6-DOF haptic rendering for dynamic rigid virtual objects with physical properties. Second, we propose an adaptive haptic rendering algorithm based on virtual coupling to overcome the inaccurate manipulation problem caused by the large mass values of the virtual tool. The algorithm can automatically adjust virtual coupling parameters according to the mass values of the simulated virtual tools. In addition, the force/torque magnitude is saturated to the maximum force/torque values of the haptic device when large interaction force is generated. The algorithm is tested on the standard haptic rendering benchmarks. Compared to other algorithms, the adaptive algorithm supports more accurate haptic manipulation. Third, we propose a new prediction algorithm for smooth haptic rendering to overcome the low update rate of the force during physical simulation of complex and/or deformable models. We propose to use a prediction method combined with an interpolation method to calculate smooth haptic interaction force. An auto-regressive model is used to predict the force value from the previous haptic force calculation. We introduce a spline function to interpolate smooth force values for the haptic force output. The proposed method can provide smooth and continuous haptic force feedback in a high update rate during the virtual manipulation of complex and/or deformable objects. It outperforms other force estimation/prediction methods.

A haptic enabled molecular docking system HMolDock is developed to find the correct docking positions between ligand and receptor. Here, a stable haptic rendering

algorithm is implemented at the application level of the system to enable stable haptic manipulation of large molecules. HMolDock can help the drug designer to find the correct docking positions between molecular systems. For medical applications, we develop a haptic-based serious game "T Puzzle" and an EEG-enabled haptic-based serious game "Basket". In the game "T Puzzle", virtual blocks are assigned with small mass values, and the stable dynamic algorithm is implemented to provide stable haptic manipulation in virtual environments. This game can be used for intellectual development and post stroke rehabilitation exercises. The EEG-enabled haptic based post stroke rehabilitation serious game "Basket" is developed to help patients to perform rehabilitation activities. In the game, the haptic device is used to manipulate various virtual objects and move them into the basket. The adaptive haptic rendering algorithm is implemented to guarantee an accurate haptic manipulation of the virtual objects with different mass values. The EEG based emotion recognition algorithm is implemented to recognize emotions of the patient and automatically adjust the difficulty level of the game. The proposed haptic rendering algorithms are also integrated in CHAI 3D library.

Acknowledgements

First of all, I would like to express my gratitude to my supervisor, Dr. Olga Sourina, whose expertise and patience added considerably to my research experience. I would also like to acknowledge that she provided all required equipment in the lab from her research projects. I would like to thank Prof. Lipo Wang for his assistance and guidance with this thesis.

Furthermore, I want to express a very special thanks to my teammates Ph.D students Yisi Liu and Qiang Wang, who gave me their support and encouragement during my research and study. I must also acknowledge M.Sc. student Hangcong Guan, FYP students Heng Chee Wei and Oh Zhanyang for their ideas on haptic-based games design. I would also like to thank technical staff from Software Engineering lab in School of EEE, Institute for Media Innovation and Fraunhofer@NTU Centre for beautiful and comfortable research environment and support.

This work is supported by IDM Grant NRF2008IDM-IDM004-002 “Visual and Haptic Rendering in Co-Space” of National Research Fund of Singapore and MOE NTU grant RG10/06 “Visual and Force Feed-back Simulation in Nanoengineering and Application to Docking of Transmembrane Helices”.

Content

Abstract.....	I
Acknowledgements	IV
Content.....	V
List of Tables.....	IX
List of Figures.....	X
Abbreviations	XV
Notations	XVII
Chapter 1 Introduction.....	1
1.1 Background.....	1
1.2 Motivation.....	2
1.3 Objective.....	4
1.4 Thesis Organization	6
Chapter 2 Background of Haptic Rendering	8
2.1 Haptic Devices	8
2.1.1 Mechanical Structure	8
2.1.2 Virtual Representation.....	10
2.2 Haptic Interaction	11
2.3 Pipeline of Haptic Rendering.....	13
2.4 Applications of Haptic Technology	14
2.5 Summary.....	19
Chapter 3 Related Work on Haptic Rendering.....	20
3.1 Haptic Rendering Algorithms	20

3.2	Point-based/3-DOF Haptic Rendering.....	22
3.3	Object-based/6-DOF Haptic Rendering	23
3.3.1	Direct Rendering	23
3.3.2	Virtual Coupling based Rendering	26
3.3.3	Comparison of Algorithms	31
3.4	Prediction for Haptic Rendering.....	33
3.4.1	Force Prediction	33
3.4.2	Force Interpolation.....	34
3.5	Benchmarks	35
3.6	Summary.....	39
Chapter 4 Stable Dynamic Haptic Rendering Algorithm		40
4.1	Stable Dynamic Algorithm based on Virtual Coupling	40
4.1.1	Haptic Force	42
4.1.2	Haptic Torque.....	45
4.2	Pipeline of Stable Dynamic Rendering	46
4.3	Experiment Results on Benchmarks.....	47
4.3.1	Peg-in-hole Benchmark	47
4.3.2	Stanford Bunnies Benchmark	51
4.4	Comparison of Haptic Forces	53
4.5	Discussion.....	53
Chapter 5 Adaptive Haptic Rendering Algorithm		56
5.1	Adaptive Force/Torque Calculation.....	56
5.2	Force Saturation Algorithm	58
5.3	Experiment Results.....	63
5.3.1	Determination of Virtual Coupling Parameters	65
5.3.2	Stability Analysis	70
5.3.3	Comparison of the Displacements	74
5.4	Discussion.....	76
Chapter 6 A Prediction Algorithm for Haptic Rendering		77

6.1	Methodology.....	78
6.1.1	Force/Torque Prediction and Interpolation	78
6.1.2	Adaptive Virtual Coupling	82
6.1.3	Architecture of Haptic Rendering	83
6.2	Experiment Results.....	85
6.2.1	Implementation and Benchmarks	85
6.2.2	Error Measurement	87
6.2.3	Smooth haptic force analysis	90
6.3	Real-time AR Coefficient Update.....	92
6.4	Discussion.....	95
Chapter 7 Haptic Based 6-DOF Molecular Docking		96
7.1	Molecular Docking Background.....	96
7.2	Related Works of Molecular Docking	98
7.3	Basic Concepts and Algorithm Description.....	100
7.3.1	Lennard-Jones Potential.....	100
7.3.2	Lennard-Jones Force	102
7.3.3	Haptic Torque.....	104
7.3.4	Stable Haptic Rendering	105
7.4	System Implementation and Performance	107
7.4.1	Implementation	107
7.4.2	System Performance	109
7.4.3	Performance Analysis	111
7.5	Collaborative Molecular Docking	113
7.6	Discussion.....	115
Chapter 8 Applications of Haptic Rendering Algorithms		116
8.1	Haptic Rendering System	116
8.2	Haptic-Enabled Puzzle Game	118
8.2.1	T Puzzle Game	118
8.2.2	Force and Torque Calculation	120

8.2.3	Implementation and Performance	121
8.3	Emotion-enabled Haptic-based Serious Game for Post Stroke Rehabilitation	123
8.3.1	Stroke Rehabilitation Game Background	123
8.3.2	Methods and Materials	125
8.3.3	Game Implementation and Result.....	127
8.3.4	Discussion of Stroke Rehabilitation Game	130
8.4	Discussion.....	131
Chapter 9 Conclusion and Future Work		133
9.1	Conclusion	133
9.2	Future Work	135
References		137
Appendix A. Haptic Devices.....		147
Appendix B. Molecular Docking		151
Appendix C. Calculation of Motion and Friction		155
Author's Publications		157

List of Tables

Table 2.1:	Application areas and examples of haptic applications.	15
Table 3.1:	Comparison of object-based haptic rendering algorithms.	32
Table 4.1:	Complexity of models used in the benchmarks.	47
Table 5.1:	Complexity of models used in the benchmarks.	64
Table 5.2:	The spring stiffness for different mass values.	66
Table 5.3:	The damping coefficients for different mass values.	68
Table 5.4:	The displacements comparison with different mass values.	75
Table 6.1:	The force prediction accuracy of different algorithms.	88
Table 6.2:	Comparison of accuracy with different window sizes.	93
Table 6.3:	Accuracy comparison of algorithms.	94
Table A.1:	Specifications of Phantom Premium 1.5/6DOF, 1.5 High Force/6DOF haptic devices.	148
Table A.2:	Specifications of the Novint Falcon haptic device.	150
Table A.3:	Protein Data Bank Record Types	151
Table A.4:	Protein Data Bank format	152

List of Figures

Figure 2.1:	Two different structures of haptic devices [30]. (a) Serial structure haptic device. (b) Parallel structure haptic device.....	9
Figure 2.2:	The object representation of the virtual tool in the virtual environment. (a) The PHANToM 6-DOF haptic device is used to manipulate the mechanical element in a complex assembly system developed by Boeing Company [2]. (b) Object representation of the drill is used in a bone drilling surgery simulation [32].....	11
Figure 2.3:	Two haptic interaction methods. (a) The point-based haptic interaction, (b) The object-based haptic interaction [33].	12
Figure 2.4:	Three main parts in the pipeline of haptic rendering system.....	14
Figure 2.5:	Biomolecular Docking System HMolDock with PHANToM 1.5/6-DOF haptic device.	17
Figure 3.1:	Classification of haptic rendering algorithms.....	21
Figure 3.2:	Pipeline of direct haptic rendering algorithms.	24
Figure 3.3:	The basic virtual coupling model used in haptic rendering [69].	27
Figure 3.4:	Mass-spring-damper link for dynamic virtual coupling.	28
Figure 3.5:	A dynamic virtual tool under the influence of both penalty force and haptic force [21]. The virtual tool is constrained on the static surface, and the haptic force is calculated from the displacement between the position of the haptic device and the virtual tool.	29
Figure 3.6:	Forcegrid data structure for haptic force interpolation [78]. (a) A virtual surgical tool located in the rectangle workspace $\mathbf{W}$. The operation point of the virtual tool is $\mathbf{P}$. (b) The force vectors $\mathbf{F}_v$ in the workspace. The haptic force of the virtual tool can be interpolated from the neighboring force vectors.....	35

Figure 3.7:	The peg-in-hole haptic rendering benchmark. Left: The models used in the peg-in-a-hole benchmark. Right: A haptic interaction in the peg-in-a-hole benchmark.	36
Figure 3.8:	Haptic interaction between two Stanford Bunnies [81]. Each bunny model contains 20898 triangles.	37
Figure 3.9:	The pipeline of the haptic algorithm evaluation system [84].	38
Figure 3.10:	The duck model used in the standard haptic evaluation system. (a) The duck model (4,212 triangles) used for haptic rendering evaluation. (b) The scanned path of the haptic probe on the surface of the duck model [84].	39
Figure 4.1:	Virtual coupling model for stable dynamic haptic rendering algorithm.	42
Figure 4.2:	The Displacement-Force relationship in the stable dynamic algorithm based on virtual coupling.	43
Figure 4.3:	Haptic rendering pipeline for stable dynamic rendering algorithm.	46
Figure 4.4:	Key configurations of the peg-in-hole benchmark tested during haptic rendering simulation.	48
Figure 4.5:	The experiment result of the original Spring-damper virtual coupling algorithm.	49
Figure 4.6:	The experiment result of the quasi-static approximation algorithm.	50
Figure 4.7:	The experiment result of the stable dynamic algorithms.	50
Figure 4.8:	Stanford bunny benchmark used for haptic rendering simulation. Each Stanford bunny contains 20898 polygons.	51
Figure 4.9:	Performance of the proposed algorithm in Stanford bunny benchmark with PHANToM haptic device.	52
Figure 4.10:	The comparison of three haptic rendering algorithms during a deep contact between the virtual tool and the virtual object. (a) The changes of the force magnitude during a deep contact between the virtual tool and virtual object. (b) A zoomed view of the buzzing at the beginning of the contact between the virtual tool and virtual object.	54
Figure 5.1:	The virtual coupling force saturated to the maximum force magnitude of the haptic device from $(1 - \varepsilon)\mathbf{F}_{\max}$	62

Figure 5.2:	The contact simulation for the force saturation. The virtual tool is tested with four mass values 0.05kg, 0.1kg, 0.15kg, and 0.2kg.....	63
Figure 5.3:	Three benchmark models used in the haptic rendering system: (a) peg model, (b) box with hole model, and (c) bunny model.	65
Figure 5.4:	The spring stiffness for different mass values.	67
Figure 5.5:	The damping coefficient for different stiffness values.	69
Figure 5.6:	Haptic rendering with the peg-in-hole benchmark. The mesh model represents the haptic device, and the green model represents the virtual tool. The displacement during manipulation of the peg inside the hole is shown.	70
Figure 5.7:	The five steps in the peg-in-hole benchmarks.	71
Figure 5.8:	The haptic performance of the peg-in-hole benchmark with different mass values. (a) The mass value is 0.05kg; (b) the mass value is 0.1kg; (c) the mass value is 0.15kg; (d) the mass value is 0.2kg. The top two rows indicate the force and torque magnitudes, and the bottom two rows indicate the computation time and the number of contact points in each haptic frame. If there is no contact, the force magnitude is generated from the mass of the peg.	72
Figure 5.9:	Haptic rendering with complex models: (a) Stanford bunny benchmark used for haptic rendering simulation. Each Stanford bunny contains 20898 polygons. (b) Performance of the proposed algorithm in Stanford bunny benchmark with PHANToM haptic device.	73
Figure 5.10:	The comparison of displacements when the algorithms render the virtual tool with different mass values (0.05kg, 0.1kg, 0.15kg, 0.2kg).	75
Figure 6.1:	The FPE results of the AR model with different orders.	80
Figure 6.2:	The force prediction and interpolation for haptic display. The prediction thread is represented with circles and the haptic thread represented with rectangles.	82
Figure 6.3:	The pipeline of the haptic rendering system using prediction algorithm. The prediction algorithm is used in the prediction thread (200Hz) to predict the	

	next force value. The interpolation algorithm is used in the haptic thread (1 KHz) to interpolate smooth haptic force values.	84
Figure 6.4:	Three benchmark models used for evaluation of the proposed haptic rendering method. (a) peg model, (b) box with hole model, and (c) bunny model.	86
Figure 6.5:	Key configurations of the haptic manipulation using the peg-in-hole benchmark.	87
Figure 6.6:	Evaluation of the accuracy of the prediction algorithms. (a), (b) The predicted force and absolute errors in the linear prediction algorithm. (c), (d) The predicted force and absolute errors in the spline interpolation based prediction algorithm.	89
Figure 6.7:	Evaluating the smoothness of haptic rendering. (a) The performance of the linear prediction algorithm with vibrations. (b) The result of the spline interpolation based prediction algorithm with smooth force change.....	91
Figure 6.8:	The haptic rendering pipeline with real-time auto-regressive coefficient calculation.....	92
Figure 6.9:	The comparison of the RMS Force Error with different window size.	94
Figure 7.1:	The simulation of Lennard-Jones potential.	101
Figure 7.2:	The zero area (in red) is set when the potential energy approximates to zero and the force changes from attraction to repulsion.....	106
Figure 7.3:	HMolDock system with PHANToM 6-DOF haptic device.	107
Figure 7.4:	The haptic rendering pipeline of the HMolDock system with both force and torque feedback.....	108
Figure 7.5:	An interaction between two α -helices: the yellow and cyan arrows represent the force and torque vectors. In (a), two molecules are separated in a distance, and the force vector indicates the attractive force. In (b), two molecules contact with each other and the force vector changed to the inverse direction as a repulsive force.	110
Figure 7.6:	Force and torque magnitude in a molecular docking process.	111

Figure 7.7:	The Simulation results of a molecular docking process using same ligand and receptor. (a) The optimized interaction force feedback during a molecular docking manipulation with stable method. (b) The interaction force feedback without stable method during a docking process.....	113
Figure 7.8:	Collaborative molecular docking with two different haptic devices. Falcon haptic device is shown on the left, and PHANToM Premium 1.5/6-DOF haptic device is shown on the right of the screen. The ligand and receptor can be controlled in the same collaborative virtual environment.....	114
Figure 8.1:	The CHAI library integrated with 3-DOF/6-DOF haptic rendering and prediction algorithm.	117
Figure 8.2:	“T Puzzle” Game: (a) Four wooden pieces of the puzzle. (b) Different levels: From junior, middle to senior level of game. (c) User interface of the haptic-based game.	119
Figure 8.3:	Haptic rendering pipeline of the “T Puzzle” game.....	120
Figure 8.4:	The user interface of “T puzzle” game. The PHANToM 6-DOF haptic device is used to manipulate virtual objects in game.....	121
Figure 8.5:	Screenshot of haptic-based “T puzzle” game. It is developed based on OpenGL and Glut.	122
Figure 8.6:	The overall diagram of the EEG-enabled haptic-based serious game for post stroke rehabilitation.	126
Figure 8.7:	EEG-based and haptic-based stroke rehabilitation 3D game. (a) User with EEG and haptic devices. (b) The hard level of game with a small basket and four objects with different mass values.	129
Figure A. 1:	PHANTOM Premium 1.5 haptic device. It comes in 6-DOF models, which offer six degree-of-freedom (3 translational, 3 torque) in output capabilities.	147
Figure A. 2:	Novint Falcon haptic device. It is a consumer product for the control of haptic games.	149

Abbreviations

2D	Two Dimension
3D	Three Dimension
3-DOF	Three Degrees-of-Freedom
6-DOF	Six Degrees-of-Freedom
AABB	Axis-aligned Bounding Box
AHW	Active Haptic Workspace
AR	Auto Regressive
BV	Bounding Volume
BVH	Bounding Volume Hierarchies
BVTT	Bounding Volume Test Tree
CAMD	Computer-Aided Molecular Docking
CHAI	Computer Haptics and Active Interfaces
CPU	Central Processing Unit
CSO	Configuration Space Obstacle
DEAP	A Dataset for Emotion Analysis using EEG, Physiological and video signals
DEEP	Dual-Space Expansion for Convex Polyhedra
EEG	Electroencephalogram
EFFD	Extended Free-form Deformation
FEM	Finite Element Method
FFD	Free-Form Deformation
FFS	Force Feedback Slider
GFT	Generalized Front Tracking
HCI	Human Computer Interface
HIP	Haptic Interface Point

HMolDock	Haptic-based Molecular Docking
IGD	Interactive Global Docking
IMD	Interactive Molecular Dynamics
LJ	Lennard-Jones
MV	Molecular Visualize
NURBS	Non-Uniform Rational B-Splines
OBB	Oriented Bounding Box
ODE	Open Dynamic Engine
PD	Penetration Depth
PDB	Protein Data Bank
PDE	Partial Differential Equations
QSA	Quasi-Static Approximation
RFFD	Rational Free-Form Deformation
RMS	Root Mean Square
SMD	Steered Molecular Dynamics
VPS	Voxmap Point Shell
VR	Virtual Reality
VRML	Virtual Reality Modeling Language

Notations

k_d	The spring translational stiffness
k_R	The spring rotational stiffness
b_d	The spring translational damping
b_R	The spring rotational damping
$\mathbf{d}_H$	The displacement between the virtual tool and the haptic device
θ_H	The equivalent-axis angle
$\mathbf{V}_{tool}$	The linear velocity of the virtual tool
$\mathbf{V}_{HIP}$	The linear velocity of the haptic device
ω_{tool}	The angular velocity of the virtual tool
ω_{HIP}	The angular velocity of the haptic device
$\mathbf{F}_{haptic}$	The haptic force
$\mathbf{T}_{haptic}$	The haptic torque
dop_{max}	The limitation of displacement
$\mathbf{p}_{HIP}$	The position of haptic device
$\mathbf{p}_{tool}$	The position of the virtual tool
$\mathbf{f}_{max}$	The maximum force of haptic device
$\mathbf{t}_{max}$	The maximum torque of haptic device
$\mathbf{A}_{HIP}$	The main axis of haptic device
$\mathbf{A}_{tool}$	The main axis of the virtual tool

$H(\mathbf{F}_{haptic})$	The saturation function for haptic force
ϕ	The coefficient of the auto-regressive model
$\mathbf{P}(u)$	The function of B-spline interpolation
m	The mass value of the virtual tool
$V(r)$	The Lennard-Jones Potential
ε	The depth of potential well
r_{ij}	The distance between the atom pair

Chapter 1

Introduction

1.1 Background

The term haptics comes from the Greek word *haptesthai* meaning “to touch”. Haptics refers to the science of providing touch interaction between computer applications and human hands. Haptic interfaces enable the user to feel physical properties of virtual objects and manipulate the objects in virtual environments. [1]. In the early 1990s, the first haptic system aimed to computer interaction between geometric objects emerged [2]. After years of development, currently, the haptic technologies have been widely used for computer simulation in virtual environment, such as medical training [3], surgery simulation [4], 3D games [5, 6], art design [7], molecular docking [8], and education[9].

The haptic rendering technology refers to the process of computing interactive forces between the virtual objects in the virtual environment [10]. There are mainly two types of representation for haptic devices: point representation and object representation. For the point representation, the algorithms calculate contact forces in three dimensions at the tip of the haptic interface point [11]. Haptic rendering algorithms that follow this point-based approach are named three degree-of-freedom (3-DOF) haptic rendering algorithms as only positions of three axes are calculated [12]. Zilles and Salisbury [13] proposed a constraint-based god-object method to improve performance of the point-based haptic rendering. Later Ruspini et al. [14] extended the god-object method and proposed a

virtual proxy method to allow more efficient 3-DOF haptic rendering.

However, for many complex applications like surgery simulation, manufacturing, and scientific exploration, it is necessary to use a haptic device to control a 3D virtual tool during the simulation. The haptic device is represented with a virtual tool that is visualized in the virtual environment. The haptic rendering methods which are used to manipulate the virtual tool and provide both force and torque feedback are called six degrees-of-freedom (6-DOF) haptic rendering methods and enable the user to have more flexibility to explore virtual environments.

1.2 Motivation

The existing haptic rendering algorithms can be classified into direct rendering and virtual coupling based rendering ones. In the direct rendering, the configuration of the haptic device is received from the device controller, and its position and orientation are directly applied to the grasped virtual tool. The final force and torque are fed back to the user directly. It could be useful if the haptic rendering and graphic rendering maintain a high update rate. In 2000, Gregory et al. [15] presented an incremental direct rendering algorithm for contact determination between convex primitives. The algorithm keeps track the pairs of closest features between convex primitives. Later, Kim et al. [16, 17] improved the direct rendering algorithm in the collision detection part and penetration depth calculation between convex polytopes. They decompose the polyhedron into the convex pieces and use the incremental algorithm to speed up the computation of penetration depth. Nelson et al [18] introduced a novel surface-surface tracing paradigm in direct haptic rendering by using a novel velocity calculation. Johnson and Willemsen [19] use specialized normal cone hierarchies for fast collision detection between the virtual objects. In addition, they implement a local gradient search to speed up the update rate of the calculation of the local minimum distances.

However, the direct rendering has several disadvantages. First, when the penetration depth is big, the objects interpenetration can be seen on the graphic display. Second, the overall stiffness of multiple point contacts may exceed the maximum stiffness value of the haptic device. Third, during multiple point contacts of complex models, the update

rate would drop to a low level, and the force/torque display would be discontinuous.

In the virtual coupling based rendering, a spring-damper link is used between the haptic handle and the virtual tool. The virtual coupling can largely improve the stability of the haptic rendering and overcome the problems that exist in the direct rendering such as force discontinuity, object interpenetration, high stiffness. It supports the modular design of haptic rendering in which the graphics and physical simulation could run at a low update rate, and the haptics rendering could run at a high update rate. McNeely et al. [20] proposed a 6-DOF Voxmap Pointshell (VPS) approach to enhance the performance of the haptic rendering system in an arbitrarily complex environment. They adopt a virtual coupler scheme, which connects the user's haptic motions with the motions of the dynamic virtual tool through a virtual spring-damper link. Wan [21] proposed a Quasi-Static Approximation (QSA) method to overcome the computational instability problem in overly stiff systems in static virtual environment. The configuration of the virtual tool is computed by the quasi-static equilibrium at each haptic frame. Later, for deformable virtual objects, Barbic et al. [22, 23] proposed a real-time time-critical method for contact between a rigid virtual tool and a reduced deformable models. A pointshell-based hierarchical representation was used for deformable objects. The force and torque are calculated from haptic rendering algorithm based on static virtual coupling. In 2004, Otaduy et al. [24] proposed a novel haptic rendering approach for virtual objects with highly combinatorial complexity and haptic textures. The contact level of multiresolution bounding volume hierarchies was used in the collision detection algorithm. Later, Otaduy and Lin [25, 26] proposed to use implicit integration to simulate the movement of rigid virtual tool and virtual objects. The linearization of the contact force, contact point clustering, and virtual coupling are used to ensure stable manipulation in haptic rendering. Furthermore, they extended their work for deformable models. Wang [27] proposed a method that can select the appropriate parameters for virtual coupling model to maintain the stability and transparency of the deformable haptic rendering.

Comparing with the direct rendering, the virtual coupling based rendering has following advantages in haptic rendering. First, it supports multi-rate haptic rendering when the graphics, physical simulation, and haptic calculation work in different update

rates to guarantee the stable force feedback. Secondly, the haptic force computed through the virtual coupling is more stable in multiple point contacts. Finally, modular software design can be realized in haptic rendering system. Since the graphic loop and haptic loop are separated, a new fast collision detection, physical simulation or force/torque calculation approaches could be easily added into the haptic rendering system.

However, in the current virtual coupling based haptic rendering algorithms and approaches, there are some unsolved and challenging problems. They can be listed as follows:

- Instability during the contact between the virtual tool and virtual objects. The instability can arise due to the large force caused by the multiple point contacts.
- Buzzing. If a virtual tool has physically based properties (for example, mass), the buzzing would appear as continuous high frequency vibrations.
- Inaccurate manipulation. When the virtual tool has a large mass value, the displacement would become larger because of the gravity. This large displacement would introduce inaccurate movement during the haptic manipulation that could cause accuracy problems.
- Discontinuous force update. When there are complex models and/or deformable models, the physical simulation may produce a low update rate of force which causes discontinuous force output on haptic device.

With the above mentioned limitations of the existing haptic rendering approaches, the motivation is to propose stable haptic rendering algorithms based on virtual coupling. The algorithms could be used in various static or dynamic applications in virtual environments to provide stable and accurate force and torque feedback. For example, stable haptic rendering and accurate haptic manipulation are the key requirements in surgery simulation, mechanical assembling, training, etc.

1.3 Objective

In a complex dynamic virtual environment, it is very important to have a stable haptic feedback. In this work, the main objective is to study stability problems in haptic

rendering systems and propose novel methods and algorithms of haptic rendering to improve stability of the systems.

First, we need to study haptic related technologies and review the existing haptic rendering approaches and algorithms as follows:

- To study efficient collision detection algorithms.
- To study virtual coupling and other haptic rendering algorithms.
- To review on 3-DOF and 6-DOF haptic rendering algorithms.

Secondly, we need to propose and develop stable haptic rendering algorithms to solve existing problems. The proposed algorithms are implemented and evaluated in the haptic rendering system with standard benchmarks.

To propose and develop stable haptic rendering algorithms the work should be done as follows:

- To build haptic rendering system with collision detection and physical simulation.
- To propose adaptive haptic rendering algorithms that can provide accurate and stable haptic feeling with different mass values.
- To propose new predictive algorithm that can improve performance of haptic rendering with complex and/or deformable models.

Thirdly, we compare the proposed algorithms with existing algorithms using the standard benchmarks. The details are listed as follows:

- To choose appropriate benchmarks in haptic rendering systems for evaluation.
- To evaluate the stability and accuracy of the proposed haptic rendering algorithms with different mass values.
- To evaluate performance of the force prediction algorithms using standard haptic force data.

Finally, as the work is supported by IDM Grant NRF2008IDM-IDM004-002 “Visual and Haptic Rendering in Co-Space” of National Research Fund of Singapore and MOE NTU grant RG10/06 “Visual and Force Feed-back Simulation in Nanoengineering and Application to Docking of Transmembrane Helices”, various haptic-based applications are developed with the proposed algorithms. The details are listed as follows:

- To propose and develop haptic-based molecular docking system and collaborative haptic-based molecular docking system.

- To design and develop haptic-based puzzle game for intellectual development and rehabilitation exercises.
- To design and develop emotion-enabled haptic-based games for post stroke rehabilitation exercise.

1.4 Thesis Organization

The rest of the thesis is organized as follows:

In Chapter 2, the hardware and software of haptic technology is introduced, and some existing haptic applications are reviewed. Then, basic haptic rendering concepts are introduced following with the haptic rendering algorithms and related works.

In Chapter 3, the direct and simulated-based 6-DOF haptic rendering algorithms are discussed. Collision detection, virtual coupling algorithms and prediction algorithms are given. Moreover, the benchmarks used in haptic rendering are listed and described.

In Chapter 4, a stable dynamic haptic rendering algorithm based on virtual coupling is proposed. It can overcome the “buzzing” problem appeared in the haptic rendering process. We test the proposed algorithm using two benchmarks in haptic rendering system and compare with other existing algorithms.

In Chapter 5, an adaptive haptic rendering algorithm based on virtual coupling is proposed. The algorithm can automatically adjust virtual coupling parameters according to the mass values of the simulated virtual tools. In addition, the force/torque magnitude is saturated to the maximum force/torque value of the haptic device when large interaction force is generated. The algorithm is tested on the standard haptic rendering benchmarks and compared to other algorithms.

In Chapter 6, a new prediction method for smooth haptic rendering is proposed to overcome the low update rate of the force during physical simulation of complex and/or deformable models. The auto-regressive model is used to predict the force value from the previous haptic force calculation. We introduce a spline function to interpolate force values for the smooth haptic force output. In addition, a real-time coefficient calculation is proposed to update the auto-regressive model during haptic rendering. The algorithms

are tested with benchmarks of peg-in-hole, bunny and data set of duck model.

In Chapter 7, a molecular docking system HMolDock is developed to find the correct docking positions. We propose a haptic rendering interface for biomolecular docking with both force and torque feedback. The stable dynamic algorithm is implemented to improve stability of the haptic rendering during docking molecular systems. Furthermore, collaborative docking with two devices is implemented.

In Chapter 8, the architecture of our haptic rendering system is presented. Then, various applications implemented with the proposed algorithms are described. First, a haptic-based serious game "T Puzzle" implemented with the stable dynamic haptic rendering algorithm is developed to provide stable and accurate haptic manipulation of virtual blocks. Second, an EEG-enabled haptic based post stroke rehabilitation serious game implemented with the adaptive haptic rendering algorithm is developed to help patients to perform rehabilitation activities.

In Chapter 9, conclusion and future works are discussed.

Chapter 2

Background of Haptic Rendering

In this chapter, I introduce some basic hardware and software technologies in haptic rendering. It starts with a review of haptic devices and their virtual representations. Then, we introduce the concepts of haptic rendering pipelines, interaction methods, and fundamental researches in haptic rendering. Finally, various applications related to our work are discussed.

2.1 Haptic Devices

2.1.1 Mechanical Structure

Haptic device can be regarded as a mechanical device that provides communication between computer and human by the energy exchange [28]. As a haptic device is a bidirectional communication device, both input and output functions can be realized through the haptic device simultaneously. As an input device, a haptic device is a tool to manipulate 3D objects in virtual environment or in tele-operation. As an output device, force feedback is exerted to the hand of the user, and he/she can have a touch feeling [1]. Examples of haptic devices include common commercialized 3-DOF or 6-DOF devices equipped with special motors and sensors (e.g., force feedback joysticks and steering wheels) and some sophisticated devices designed for scientific simulation, medical training, and industrial applications [29].

Figure 2.1: Two different structures of haptic devices [30]. (a) Serial structure haptic device. (b) Parallel structure haptic device.

- **Serial structure**

For haptic devices with serial mechanisms, all actuators are in serial order within one single kinematic chain, shown in Fig. 2.1(a). It doesn't include any passive joint. For instance, the SensAble PHANToM and Omni haptic devices have the serial structure design and are composed with a set of actuators and robotic arms. In our work, we use the PHANToM Premium 1.5 with serial structure to test proposed and implemented 6-DOF haptic rendering algorithms. The PHANToM Premium 1.5 device description is given in Appendix A.1.

- **Parallel structure**

The haptic devices with parallel structure place all actuators at the frame which is non-moving body. It can minimize the moving masses of the haptic handle, as shown in Fig. 2.1(b). For example, Novint Falcon [31] is this kind of commercial haptic device. Parallel design is successfully used in Falcon haptic devices which can provide users a truly virtual touch in game applications. When the user holds a weapon in a game, the gravity can be simulated to make the user feel the weight of the weapon. When users are fighting in the game, the vibration and collision will be generated so that the users can

feel the shooting of gun or fighting with sword. The Falcon haptic device description is given in Appendix A.2.

In our works, the serial structure haptic device (PHANToM Premium 1.5 device) is used for HMolDock molecular docking and Puzzle game. The parallel structure haptic device (Novint Falcon device) is used for EEG-enabled Haptic-based Stroke Rehabilitation Game.

2.1.2 Virtual Representation

Haptic devices can be classified by their virtual representation in virtual environment during haptic rendering. Depending on the application, haptic devices can be represented as a single point (for 3-DOF haptic device) or a complex virtual tool (for 6-DOF haptic device) in the virtual environment.

- **Point representation**

The 3-DOF haptic device is commonly represented as a virtual point. In virtual environment, the position of the haptic interface point is calculated in the coordinate system along three axes, and the haptic force is represented as a three dimensional vector. The direction and magnitude of the vector correspond to the directions and magnitudes of the haptic force along three axes. For point representation applications, the interactive forces are calculated based on the position of a haptic interface point, then, the force is applied to the user through the haptic device. The PHANToM Omni and Novint Falcon are typical 3-DOF haptic devices which can be used in such applications.

- **Object representation**

Recently, virtual environments became more and more complex. The point representation of 3-DOF haptic device cannot satisfy the requirement of sophisticated simulations in complex virtual environments. For this purpose, the object representations of 6-DOF haptic devices are used to manipulate complex virtual tools in virtual environments.

6-DOF haptic device can provide not only movement along three axes but also rotation around three axes. Such devices have been widely used in medical training

systems, virtual assembly, scientific applications, etc. For example, SensAble PHANToM 6-DOF devices can provide force feedback in three translational degree-of-freedom and torque feedback in three rotational degree-of-freedom of the yaw, pitch and roll directions [29]. As shown in Fig. 2.2(a), the assembling simulation makes it possible to feel the reaction force and torque produced by multiple point contacts between machine parts. In the medical training application shown in Fig. 2.2(b), the 6-DOF haptic device can be used as a drilling tool in the bone surgery simulation. The haptic rendering processes the collision between the virtual drill and bones to calculate and display force feedback. The combination of the force and torque feedback makes the training process more realistic for medical students.

Figure 2.2: The object representation of the virtual tool in the virtual environment. (a) The PHANToM 6-DOF haptic device is used to manipulate the mechanical element in a complex assembly system developed by Boeing Company [2]. (b) Object representation of the drill is used in a bone drilling surgery simulation [32].

2.2 Haptic Interaction

The existing interaction methods for haptic rendering can be distinguished by virtual representation of the haptic devices (described in Section 2.1.2). There are mainly two kinds of interaction methods to calculate the interaction force/torque in the virtual environment: point-based (3-DOF) haptic interaction and object-based (6-DOF) haptic interaction, shown in Fig. 2.3.

Figure 2.3: Two haptic interaction methods. (a) The point-based haptic interaction, (b) The object-based haptic interaction [33].

- **Point-based Haptic Interaction**

In point-based haptic interactions, the interaction and collision are only happened between the Haptic Interface Point (HIP) and the surface of the virtual object, shown in Fig. 2.3(a). The HIP is represented as a minimize point in the virtual environment. When the user moves the probe of haptic device, the collision detection algorithm will check is the position of the HIP inside or outside the virtual object in each update [33].

When collision happens, there is a proxy of HIP (also known as god-object) on the surface of the object [34], shown in the Fig. 2.3(a). The real HIP is shown as empty circle and the proxy of HIP is shown as filled circle. Then, the algorithm calculates the depth of indentation as the distance between two real and proxy HIP, and the force $\mathbf{F}_p$ can be calculated based on the following equations:

$$\mathbf{F}_p = K_s \mathbf{d}, \quad (2.1)$$

where K_s is the stiffness parameter, and $\mathbf{d}$ is the distance between the real and virtual HIP.

For exploring the shape and surface properties of objects in virtual environments, point-based methods can provide the users with similar force feedback as what they

would feel when exploring the objects in real environments with the tip of a stick. However, it is not capable of simulating more general object-based interactions which have multiple point contacts between virtual objects at any location of the virtual environment. In such simulation, both force and torque need to be computed separately and be sent to the user.

- **Object-based Haptic Interaction**

In object-based haptic interactions, the handle of the haptic device is represented by a virtual tool which can be manipulated in six degree-of-freedom and provide force and torque feedback, shown in Fig. 2.3(b). The simulation of haptic interactions between the virtual tool and virtual objects is desirable for many applications, but it is computationally more expensive than the point-based haptic interactions. As shown in Fig. 2.3(b), a single point is not sufficient for computing the force $\mathbf{F}_0$ and torque $\mathbf{T}_0$ between the virtual tool and virtual object. The force and torque are calculated from the multiple point contacts $\mathbf{p}_i$ which are distributed over the surface of the virtual object. The force and torque calculation when multiple point contacts collision detection happened are explained in section 3.3.

2.3 Pipeline of Haptic Rendering

Haptic rendering is a bidirectional interactive activity. Generally, to complete haptic rendering the following three parts are needed: graphic rendering, haptic rendering, and physical simulation. The relationship between software and haptic device is shown in Fig. 2.4.

When the user manipulates the haptic device, the configurations of orientation and position are acquired at 1 KHz and sent to the haptic rendering system. Then, based on the configurations of the haptic device, contacts between the HIP or the virtual tool and virtual object are detected by the collision detection algorithm. From the contact results, the movement of the virtual tool and other objects are calculated. The force/torque algorithm calculates the interactive force/torque which is applied to the haptic device.

The graphic engine renders the models based on the positions calculated from the contact simulation and visualizes them on the display.

Figure 2.4: Three main parts in the pipeline of haptic rendering system.

2.4 Applications of Haptic Technology

The haptic technology is used in virtual environments, education, scientific simulation, medical training, rehabilitation, etc. Compared with traditional human-computer interface devices such as keyboard and mouse, the advantages of the haptic device is that it provides more flexible operation in 3D environment and real-time force/torque feedback to the user. Therefore, with haptic technology, many applications can be more advanced with real world simulation. In this Section, we follow the classification of haptic applications proposed in [1]. Table 2.1 lists the classification of application areas and corresponding examples.

Table 2.1: Application areas and examples of haptic applications.

Application Areas	Examples
Medicine	Rehabilitation system, training, and surgical simulation.
Risky and Specialized Areas	Astronauts, mechanics, and military.
Education	E-learning, chemistry, and molecular biology.
Creative 3D Work	Modeling, product design, and 3D painting.
Interaction in VR Environments	3D games and science simulation.

- **Haptics in Medical Simulation**

The haptic device has a great potential for medical applications, such as surgical simulation, tele-surgery, and rehabilitation exercise. Many challenges of surgical simulation can be solved with haptic technology. In the past decade, more and more haptic based training systems were used in medical simulation. These systems include surgery simulation, rehabilitation and tele-surgery. For haptic-based medical simulation, first, it could be generated complex virtual environment like the operating room. Second, the surgeon can practice and explore the surgical procedures in a virtual environment on computer. Third, the haptic manipulation of the hand movement could be recorded for students. The students can repeat the surgery procedures for practice [1].

Depending on the complexity of operation, the surgical simulation can be classified into needle surgery, minimally invasive surgery, and open surgery [35]. For rehabilitation applications, the haptic device can apply force to the injured organs to help patient to regain its strength and motion. The haptic technology has advantages in motion and

force/torque control of the rehabilitation therapy [1]. The stability of haptic rendering is also crucial in medical simulations which require an accurate manipulation and continuous force feedback. For example, stable haptic manipulation is needed for surgeon's training. Now, there are many researches focus on the haptic based medical applications [36, 37].

- **Haptics in Biomolecular Simulation**

Haptic enabled biomolecular simulation is another popular area for haptic applications. For instance, haptic-based technology can be used to enhance molecular visualization systems and molecular docking systems with force feedback in such way that the user could “feel” force fields of molecule and molecular interaction. There are haptic-based systems that enable users to feel the electrostatic force between a probe molecule and the explored biomolecule. Lai-Yuen and Lee [38] developed computer-aided design system for molecular docking and nano-scale assembly. During this docking process of ligand to protein, the force feedback is calculated according to van der Waals forces. Stocks and Hayward developed a haptic system HaptiMol ISAS [39]. In another approach for the rigid body molecular docking, proposed by Subasi and Basdogan [40], the user can inserts a rigid ligand molecule into a protein molecule to explore the binding cavity. Similarly to the cube approach, an Active Haptic Workspace (AHW) was implemented for the haptic-based exploration of large protein-protein docking. In the Interactive Molecular Dynamics system, the molecules are simulated with dynamic movement, force feedback and graphic display [41].

We developed the system of biomolecular docking HMolDock (**H**aptic-based **M**olecular **D**ocking) (described in Chapter 7) that can be used in molecular docking application [42]. The system is shown in Fig. 2.5. The format of molecular structure file from the Protein Data Bank is chosen as an input source. Two molecules or one molecule are visualized on the screen. As it shown in Fig. 2.5, the user can manipulate the ligand using a 6-DOF haptic device and move the molecule towards/around the receptor. The interaction force and torque are calculated at each position, and the resulting attraction/repulsion force between two molecules is applied to the user through the haptic device.

Figure 2.5: Biomolecular Docking System HMolDock with PHANToM 1.5/6-DOF haptic device.

- **Haptics in Data Visualization**

Haptic based data visualization provides the users with a more flexible interface to analyze scientific problems. The haptic device could reflect the magnitude and direction of vectors more directly and accurately. For example, SCIRun is a problem-solving system designed for scientific computing [43]. The haptic based data visualization system can interact with vector fields, such as magnetic force, gravity, current, pressure and so on. The user can move the HIP of haptic device through the data volume of vectors. During the movement, the graphic displays the position of the HIP, and the vector magnitude and orientation could be realized by the user through haptic feedback.

- **Haptics in E-Commerce**

In the electronic commerce, the haptic interface enables the user to feel physical properties of a product [1]. Human hands can feel the temperature, hardness, and weight properties of the product. Consumers like to touch or try a product (such as cloth or bed sheet) before they buy [44].

- **Haptics in Education**

With the development of the virtual reality technology, people can learn information more efficiently through the haptic interface in the virtual environment. For example, in the geometry study, the student can feel the force/torque feedback with haptic manipulation to understand a shape [1]. A force feedback slider (FFS) application is developed to help users to understand the laws of physics [45]. The user manipulates the haptic device to grab the slider which moves in one degree-of-freedom. The haptic force feedback is calculated from the physical law according to the movement of user's hand.

- **Haptics in Entertainment**

Recently, the haptic-based computer games have got fast development as the haptic device became cheaper. There are four important aspects in game experience: physical, social, emotional, and mental [1]. Haptic technology can provide the real-world physical feeling to improve game experience of users.

For example, in a game of haptic battle pong, the PHANTOM haptic device is used to control the movement of a pong through a paddle. When a contact happens, the force feedback is calculated and applied to the user through the haptic device [11]. In another game Haptic Airkanoid, the user manipulates the paddle to hit the ball against a wall. The haptic device provides vibration feedback to enhance the entertainment in the game [46]. In the haptic-based game HaptiCast [5], the user controls wands with haptic force feedback in the virtual world.

In our work, we develop a serious game "T puzzle" (described in Section 8.2) with the stable dynamic haptic rendering algorithm (described in Chapter 4).

- **Haptics in Arts and Designs**

Haptic interface improves realism of virtual sculpting, 3D modeling, and painting [1]. For example, a novel painting system DAB is proposed with haptic interface [7]. The haptic rendering improves the sense of realism of the control feeling of the paint brush. In this simulation, the haptic device is used as a virtual paint brush which is modelled as mass-spring system skeleton with a subdivision surface. The deformation of the brush is calculated according to the contact with canvas. According to the tests, the painting system provides a high-quality simulation of the real-world painting environment and the user can create art work in few minutes.

In this work, we propose and implement haptic rendering algorithms that can be used in the above applications. The extended 3-DOF/6-DOF haptic rendering system based on CHAI library is developed with the proposed algorithms. The extended system is described in Section 8.1. In addition, two games T Puzzle (described in Section 8.2) and EEG-enabled Haptic-based Stroke Rehabilitation Game Basket (described in Section 8.3) are implemented based on the stable haptic rendering algorithms. Moreover, HMolDock (**H**aptic-based **M**olecular **D**ocking) system for real-time haptic-based visual molecular docking is developed. The stable dynamic algorithm is proposed and implemented to improve the haptic manipulation of molecular systems.

2.5 Summary

In this chapter, the related technologies of haptic rendering which include hardware design, virtual representation of haptic devices, and a pipeline of haptic rendering were reviewed. There are mainly two kinds of designs for bidirectional haptic devices, which are a serial structure and parallel structure. In virtual environments, the virtual tool can be represented with a point or a virtual object. The point representation is corresponded to the 3-DOF haptic rendering that only calculates force feedback in three axes. The object representation is corresponded to the 6-DOF haptic rendering that calculates both force feedback along three axes and torque feedback around three axes. Furthermore, a basic haptic rendering pipeline was described. It consists with three basic modules: graphic rendering, haptic rendering, and physical simulation. At last, various application areas using haptic rendering technologies and the corresponding related works and examples were discussed.

Chapter 3

Related Work on Haptic Rendering

In this chapter, related works on haptic rendering algorithms, prediction algorithms, and haptic evaluation benchmarks are introduced. The classification of haptic rendering algorithms is described in Section 3.1. In Section 3.2 and Section 3.3, the 3-DOF and 6-DOF haptic rendering algorithms are reviewed. In Section 3.4, the force prediction algorithms used in haptic rendering are introduced. Finally, in Section 3.5, the benchmarks for haptic rendering evaluation are described.

3.1 Haptic Rendering Algorithms

The haptic rendering algorithms are used to calculate interactive force feedback between the virtual tool and virtual objects in the virtual environment [10]. The classification of existing haptic rendering algorithms is shown in Fig. 3.1. According to representation of the virtual tool gripped by the haptic device, there are 3-DOF haptic rendering and 6-DOF haptic rendering algorithms. In the 3-DOF haptic rendering, the haptic device is represented with a point probe in the virtual environment. The 3-DOF haptic rendering algorithms compute the force components of three directions at the probe's tip [47]. The 6-DOF haptic rendering is proposed to calculate the interactive force and torque in six degree-of-freedom based on the contact between a virtual tool and a virtual object. It can be used for more complex simulations which require the virtual tool with rotation manipulation and torque feedback, for example in mechanical assembling and medical

training. According to the force/torque calculation methods, the 6-DOF haptic rendering could be classified into direct rendering and virtual coupling based rendering. For direct 6-DOF haptic rendering, the force/torque is calculated directly from the pure geometry contacts. It involves collision detection, contact manifold computation, penetration depth estimation and forces computation [15].

Figure 3.1: Classification of haptic rendering algorithms.

Although the direct rendering has an apparent simplicity, the computation of contact and penetration depth may lead to problems of object interpenetration and force discontinuity because of the expensive computation of contact points. To improve the stability of 6-DOF haptic rendering, haptic rendering algorithms based on virtual coupling are proposed for the haptic manipulation of both rigid and deformable models [23, 26, 48, 49]. The haptic rendering algorithms based on virtual coupling can be classified as static virtual coupling and dynamic virtual coupling based on the type of the virtual coupling spring [50]. In the static virtual coupling algorithms, the virtual tool has zero mass value. In the dynamic virtual coupling algorithms, the virtual tool can have different mass values, and dynamic simulations are done in virtual environments.

3.2 Point-based/3-DOF Haptic Rendering

There are many researches focused on point-based haptic rendering algorithms with 3-DOF haptic devices [34, 51, 52]. The approaches are based on the point representation haptic rendering in which the haptic device controls a probe point to touch the virtual objects in virtual environment.

In the vector field method [53, 54], the contact point is set on the polygon surface closest to the position of the haptic interface point. However, in the vector field method, the force is calculated from the penetration depth between the current position of the haptic interface point and the closest surface polygon. The previous position information of the haptic interface point is not used in the force calculation. Thus, the vector field method has the problem of force discontinuity when a virtual tool is touching the edges of virtual object or thin virtual objects [28]. When the HIP is on the edge of virtual object, the closest point may jump from one surface to another discontinuous surface. Furthermore, when the HIP passes through a thin virtual object, the contact point will penetrate to the opposite side of the object. These methods can lead to the force discontinuities and “pop-through” problems.

To solve these problems the proxy-based methods are proposed. Zilles and Salisbury [13] proposed the god-object algorithm for 3-DOF haptic rendering using a virtual HIP. The position calculation of the virtual HIP is subject to contact constraints of the virtual object. This algorithm allows the user to intuitively control the point probing the virtual objects without penetration. The god-object haptic rendering algorithm will check the position of haptic probe in the current haptic frame and previous haptic frame. So, it avoids the discontinuity problems appeared in closest-point algorithms.

Ruspini et al. [14] proposed a virtual proxy algorithm to avoid gaps between triangles on model surface, the contact point is modelled to be a sphere with a radius, and the position of the sphere is calculated from contact constraints. Ruspini also added some other effects like force shading for rounding of corners (modifying the normals of constraint planes) and friction on the contact surface.

3.3 Object-based/6-DOF Haptic Rendering

The point-based interface has limitations in complex haptic-based simulations like, for example, surgery training simulation which requires both translational and rotational manipulation and both force and torque feedback. On one hand, the grasped virtual tool requires six degree-of-freedom control. The haptic feedback is not only the force but also rotational torque on the user's hand. On another hand, the collision between the virtual tool and the virtual object in the virtual environment causes multiple point contacts.

The 6-DOF haptic rendering algorithms are proposed to calculate both force and torque feedback for objects-based haptic interaction. [17, 48, 49, 55-58]. Comparing with 3-DOF point-based haptic rendering, the 6-DOF haptic rendering has the following features:

- Multiple point contacts between the virtual tool and the virtual objects instead of the point-based contact.
- Both force and torque feedback is applied to haptic device.
- Stable and accurate haptic rendering in dynamic virtual environment for physically-based manipulation of virtual tool.

As shown in Section 3.1, the existing methods of 6-DOF haptic rendering can be classified as direct rendering and virtual coupling based rendering based on their overall computation pipelines.

3.3.1 Direct Rendering

The overall pipeline of direct rendering methods is shown in Fig. 3.2. Direct rendering relies on an impedance-type control strategy. First, the configuration of the haptic device is received from the controller. Its position and orientation are directly applied to the grasped virtual tool (virtual representation of the haptic device). The collision detection is performed between the virtual tool and other virtual objects in virtual environment. The penetration depth calculation uses a function to computer the depth of interpenetration between the virtual tool and other virtual object. Finally, the force and torque results are directly fed back to the user through the haptic device.

Figure 3.2: Pipeline of direct haptic rendering algorithms.

The popularity of the direct rendering methods comes from its simple calculation for tool's configuration. This comes on the account that the position and orientation of the haptic device is passes directly to the virtual tool. However, there are some inevitable disadvantages for penetration-depth based force calculation. If the haptic update rate drops below the range of stable rendering or the penetration depth is large, the haptic rendering system tends to be instable and discontinues.

Gregory et al. [15] proposed a direct 6-DOF haptic rending composed with collision detection, computing of the contact manifold, penetration depth estimation, and force/torque response. The algorithms use incremental techniques for contact calculation between the virtual tool and the virtual objects and predictive algorithms for penetration depth estimation. Kim et al. [59] developed a DEEP haptic rendering using Minkowski sums to calculate the locally optimal position on the surface and an incremental algorithm to compute the penetration depth during the contact between the virtual tool and the virtual object. Johnson and Willemsen [19] proposed to use localized contact computations for polygonal models. The spatialized normal cone hierarchies are implemented to realize fast collision calculation. Nelson et al. [18] use a surface-surface algorithm to track contact points with minimum penetration depth. The traced points can be used for penalty-based force feedback.

Wang et al. [60] proposed an efficient a configuration-based constrained optimization method for fine haptic manipulation in narrow space. The quasi-static motion of the virtual tool is calculated from a configuration-based constrained optimization. The sphere-tree based constraint identification method is used to calculate contacts between

virtual objects. During haptic rendering, both collision detection and collision response are updated at 1 kHz haptic rate. The calculated haptic forces and torques are applied to the haptic device directly. Although it could be applied in complex applications like surgical simulations, it cannot be used in fast moving scenarios like fighting games.

3.3.1.1 Collision Detection

The collision detection approaches have been well studied in computer graphics and computer animation. In haptics, the basic ideas of collision detection are similar with computer graphics. Bounding volume hierarchies (BVH) is widely used for collision detection. According to the bounding volume, the collision detection algorithms include sphere trees [61], AABB trees [62], OBB trees [63], convex hull-based trees [64], and swept sphere volumes [65].

3.3.1.2 Penetration Depth Computation

A few efficient algorithms are proposed to compute the penetration depth (PD) between virtual objects. Dobkin et al. computed the directional PD using Dobkin and Kirkpatrick polyhedral hierarchy [66]. Agarwal et al. used a randomized approach to compute the PD [67]. They proposed some approximation approaches to estimate the efficient of PD computation. This algorithm could compute the upper and lower bounds for PD of virtual objects. This method also could be improved by expanding a polyhedral approximation of the Minkowski sum of two polytopes [68].

3.3.1.3 Contact Force Computation

In the direct haptic rendering, the contact force can be calculated with volumetric approaches or prediction algorithms[4]. The contact force can be calculated with volumetric approaches or prediction algorithms. One way to calculate the object-based haptic force is using all the surface points of the virtual objects to check whether the virtual tool is inside or outside the objects. The contact points are used to trace the force direction and magnitude, moreover, the sum vector is the force vector of the haptic device. The disadvantage is that, the forces calculated from the sum vector are discontinuous when the number of the contact points changes quickly.

The direct rendering algorithms still have several limitations as follows.

- 1) Object interpenetration. The contact force is calculated from the penetration depth of the virtual tool and virtual object. If the contact force is large, the penetration depth is correspondingly large, which can be seen on the graphic display.
- 2) High stiffness. When the multiple point contacts happen, stiffness of each contact point could be accumulated together. The final stiffness of the system can exceed the maximum stiffness value of haptic device.
- 3) Low update rate. During multiple point contacts of complex models, the update rate would drop to a lower level, and the force/torque display would be discontinuous.

3.3.2 Virtual Coupling based Rendering

Despite the apparent simplicity of direct rendering, the stability may be a problem because of the computation of contact and haptic force/torque. The stability of haptic rendering can largely be improved by separating configurations of the device and virtual tool, and connect them with a virtual coupling link [69]. The connection of passive subsystems through virtual coupling can improve the overall stability of the system. The most common form of virtual coupling is a viscoelastic spring-damper link. This concept is used in 6-DOF haptic rendering by considering translational and rotational springs [48]. The use of virtual coupling separates the simulation of the virtual tool from the manipulation of the haptic device.

3.3.2.1 Virtual Coupling

The general structure of virtual coupling is shown in Fig. 3.3. The virtual coupling connects the haptic handle and the virtual tool and consists of a virtual spring and virtual damper in mechanical parallels. There are mainly two kind of Virtual Coupling: dynamic virtual coupling and static virtual coupling. In the dynamic virtual coupling, the virtual tool is assigned a non-zero mass value and connected with haptic device using a mass-spring-damper. Static virtual coupling simplifies the mass-spring damper model with a quasi-static spring and assigns a zero mass value to the virtual tool.

Figure 3.3: The basic virtual coupling model used in haptic rendering [69].

3.3.2.2 Dynamic Virtual Coupling

The original Voxmap-Pointshell (VPS) haptic rendering of McNeely uses a dynamic virtual coupling [20] to generate stable force and torque. For this dynamic model, an impedance approach is used to sense the motion of haptic device and generate a force/torque feedback. They use a “virtual coupler” scheme to connect the user’s haptic motions with the dynamic virtual tool through a spring-damper link. This is a well-known method to enhance the stability of haptic system.

In Fig. 3.4, the virtual haptic device is placed in the virtual scene and is coupled to the virtual tool through a mass-spring-damper connection. The real haptic device controls the position and orientation of its virtual counterpart. The spring’s displacement and rotation generate force and torque on the dynamic virtual tool and an opposite force on the real haptic device. Mass properties are assigned to the virtual tool which can be felt through the user’s hand.

The force and torque equations are as follows:

$$\mathbf{F}_{haptic} = k_d \mathbf{d}_H - b_d (\mathbf{V}_{HIP} - \mathbf{V}_{tool}), \quad (3.4)$$

$$\mathbf{T}_{haptic} = k_R \boldsymbol{\theta}_H - b_R (\boldsymbol{\omega}_{HIP} - \boldsymbol{\omega}_{tool}), \quad (3.5)$$

where k_d, b_d are spring translational stiffness and damping, k_R, b_R are spring rotational stiffness and rotational damping, $\mathbf{d}_H$ is the displacement between the virtual tool and the

haptic device. θ_H is equivalent-axis angle (including axis direction) between the virtual tool and the haptic device. $\mathbf{V}_{tool}, \omega_{tool}$ are the dynamic virtual tool's linear and angular velocity. $\mathbf{V}_{HIP}, \omega_{HIP}$ are the haptic device's linear and angular velocity.

Figure 3.4: Mass-spring-damper link for dynamic virtual coupling.

3.3.2.3 Static Virtual Coupling

As it is described in Section 3.3.2.2, the dynamic virtual coupling uses a linear displacement-force relationship. But, in the static virtual coupling, the spring-damper is replaced with a quasi-static spring which is composed with one scalar parameter for translational virtual coupling stiffness and one scalar parameter for rotational stiffness.

The static virtual coupling algorithm is first proposed in [21]. A quasi-static approximation (QSA) approach is proposed to simulate the movement of the virtual tool with zero-mass value in 6-DOF haptic rendering. The advantage of the static virtual coupling is that it overcomes the computational instability problem in overly stiff systems described in VPS algorithm. As described in [20], the high performance of the voxel-based approach comes dynamic virtual coupling and a memory-efficient voxel tree.

Although it supports real-time force feedback from multiple point contacts, the dynamic simulation still have some problems of stability. For example, when multiple

point contacts occur between the virtual tool and the virtual object, the accumulated stiffness may become so large that it can provoke haptic instabilities.

The QSA algorithm modifies the virtual coupling displacement-force relationship to make sure large spring force does not exceed the maximum force of haptic device. They use an initial linear region when the penetration is small, but then exponentially saturates the force to a maximum force value.

In the QSA algorithm, static equilibrium is established by two force/torque pairs (see Fig. 3.5, only force is drawn) such as a penalty force pair from multiple point contacts and a haptic force pair from static virtual coupling. They are created from the user motion of the virtual haptic device.

Figure 3.5: A dynamic virtual tool under the influence of both penalty force and haptic force [21]. The virtual tool is constrained on the static surface, and the haptic force is calculated from the displacement between the position of the haptic device and the virtual tool.

- **Penalty Force and Torque**

From the depth of penetration at equilibrium, the penalty force $\mathbf{F}_i$ can be obtained based on the tangent-plane force model. Then, the net penalty force $\mathbf{F}_{coll}$ is calculated as

the sum of the penalty forces of all contact points on the virtual tool:

$$F_{coll}(\mathbf{d}, \mathbf{r}) = \sum \mathbf{F}_i(\mathbf{d}, \mathbf{r}) = \sum k_i d o p_i(\mathbf{d}, \mathbf{r}) N_i, \quad (3.6)$$

where k_i is the penalty-force stiffness at each point. $\mathbf{d}$ is translational movement and $\mathbf{r}$ is rotational movement. N_i is osculation normal.

Similarly, we can assume R_i is the moment arm pointing from the dynamic object's center of mass. The symbolic representation of the net penalty torque $\mathbf{T}_{coll}$ on the virtual tool at equilibrium can be calculated as:

$$\mathbf{T}_{coll}(\mathbf{d}, \mathbf{r}) = \sum k_i d o p_i(\mathbf{d}, \mathbf{r}) (R_i \times N_i). \quad (3.7)$$

- **Haptic Force and Torque**

In the static virtual coupling model, the haptic force/torque is calculated from the displacement and the equivalent angle between the virtual tool and the haptic device [21].

First, assume that $\mathbf{d}_H$ is the displacement of the spring, θ_H and K_H are the equivalent angle and the equivalent axis between the virtual tool and the virtual haptic device. Then, at equilibrium, the displacement $\mathbf{d}_e$ of the static spring can be represented as $(\mathbf{d}_H - \mathbf{d})$, where $\mathbf{d}$ is the translational movement of the virtual tool during the current time step. The haptic force at equilibrium is:

$$\mathbf{F}_{haptic}(\mathbf{d}) = k_d \mathbf{d}_e = k_d (\mathbf{d}_H - \mathbf{d}), \quad (3.8)$$

where k_d is the translational stiffness.

According to the static virtual coupling model, the haptic torque $\mathbf{T}_{haptic}$ at equilibrium from θ_e and K_e as follows:

$$\mathbf{T}_{haptic}(\mathbf{r}) = k_\theta (\theta_e K_e) = k_\theta (\mathbf{r}_H - \mathbf{r}), \quad (3.9)$$

- **Force Saturation**

In QSA, the 6-DOF spring force should be smaller than the maximum force value of the haptic device. Otherwise, penetration appears between the virtual tool and virtual object. The spring stiffness can be considered as a constant during each haptic frame [21]. The spring stiffness (3.11) could be derived from the function of the spring displacement (3.10).

Assuming $dop_{\max}$ is the limitation of displacement. The stiffness parameter k_d can be calculated by the following functions:

$$dop_{EQ}(\mathbf{d}_H) = dop_{\max} (1 - e^{-\|\mathbf{d}_H\|/dop_{\max}}), \quad (3.10)$$

$$k_d(\mathbf{d}_H) = dop_{EQ}(\mathbf{d}_H) / \|\mathbf{d}_H\|. \quad (3.11)$$

In order to avoid surface sticky effect, the function should be applied only when the user pushes the virtual tool toward the virtual object.

Similarly, the rotational stiffness k_θ as a function of the angle θ_H between the virtual tool and the virtual haptic handle, by using the following equations:

$$\theta_{\max} = dop_{\max} / R_{\min}, \quad (3.12)$$

$$\theta_{EQ}(\theta_H) = \theta_{\max} (1 - e^{-|\theta_H|/\theta_{\max}}), \quad (3.13)$$

$$k_\theta(\theta_H) = (2R_{\min})\theta_{EQ}(\theta_H) / |\theta_H|, \quad (3.14)$$

where $R_{\min}$ represents the moment arm and $\theta_{\max}$ represents the maximum rotation offset.

3.3.3 Comparison of Algorithms

A comparison of the existing object-based haptic rendering algorithms is shown in Table 3.1. We compare the haptic rendering abilities and performance of the direct rendering

algorithms [17, 55], dynamic virtual coupling algorithms [48, 70], and static virtual coupling algorithms [49, 58].

Table 3.1: Comparison of object-based haptic rendering algorithms.

		Direct Rendering	Virtual Coupling based Rendering	
			Dynamic Virtual Coupling	Static Virtual Coupling
Rigid Objects		Yes	Yes	Yes
Complex Environment		No	Yes	Yes
Dynamic Simulation		No	Yes	No
Collaborative Environment		No	Yes	Yes
Rendering Speed		Medium	Fast	Fast
Force Feedback	Stability	Medium	Medium	High
	Stability with Mass	No	Low	No
	Accuracy	Medium	Low	Low

In the object-based haptic rendering algorithms, although the virtual coupling based algorithms have obvious improvement on the stability, rendering speed, and dynamic simulation comparing with the direct rendering algorithms as it is seen from Table 3.1, there are still some unsolved problems. For example, 1) Buzzing. If a virtual tool has physically based properties (for example, mass), the buzzing would appear as continuous high frequency vibrations. 2) Inaccurate manipulation. When the virtual tool has a large mass value, the displacement would become larger because of the gravity. This large displacement would introduce inaccurate movement during the haptic manipulation that can cause accuracy problems. 3) Discontinuous force update. When there are complex models and/or deformable models, the physical simulation may produce a low update rate of force which causes discontinuous force output on haptic device.

3.4 Prediction for Haptic Rendering

3.4.1 Force Prediction

Smooth haptic force feedback is an important task for multirate haptic rendering for both rigid and deformable objects. The update rate of the haptic force may be too low and may be changed during the simulation as the high computation time is required for complex physical simulation or for deformation model simulation. Therefore, to implement a stable and smooth haptic rendering, some prediction algorithms are proposed to calculate smooth haptic interaction force in a high update rate.

Picinbono [71] proposed a linear extrapolation method to reconcile the update rate of the physically based deformable simulation and the haptic rendering. The linear extrapolation algorithms are used to predict the force in haptic thread. In [72-74], the real-time linear force extrapolation algorithm is implemented in the minimally invasive haptic surgery simulator. As the deformable biomechanical model is simulated at a low rate of about 30 Hz, they use the proposed linear force prediction algorithm to achieve high frequency update of haptic force feedback. Kim [68] implemented a real-time haptic rendering system for deformable objects based on visual information such as images obtained from the camera. They use the force extrapolation based on the positions of the manipulator's tip captured from the images. Hu [75] developed a magnetic haptic feedback system in which the position of the tool is estimated by using a video-based surgical-tool tracking algorithm. The position of the surgical-tool is estimated using linear extrapolation to solve the problem of different update rates. But if the force changes its direction, the extrapolated force may be over-shoot or under-shoot. Although the linear extrapolation over position could provide more accurate results, this implementation is for point-based 3-DOF haptic rendering [71]. For an object-based 6-DOF haptic rendering that calculates the interaction force from all contact points between the virtual tool and virtual objects, the extrapolation based on the position of haptic device is not the accurate one.

To improve the stability of the haptic force prediction, some auto-regressive methods are proposed for haptic rendering. A time series based prediction method is proposed by

Wu [76], where AR model is used to extrapolate the force for haptic display. When the deformable model simulation is being calculated, the predicted force value is used for haptic force update. For physically-based medical simulation of the organs and tissues, Lee [77] proposed a multi-rate output-estimation using the ARMAX model to improve the computational speed and accuracy. Although the AR model could improve the prediction accuracy, the problem of smooth haptic rendering between two successive haptic frames still is not solved.

3.4.2 Force Interpolation

To provide smooth haptic force feedback, Mazzella[78] proposed a force grid data structure for the haptic force interpolation and extrapolation, shown in Fig. 3.6. In the forcegrid structure, the virtual workspace is divided into regular grids, and the force values are interpolated in each vertex. The force interpolation algorithm is independent from the modeling method and the simulation rate. So, the haptic device can get force feedback at a higher update rate, regardless of the complexity of the models and virtual environment. The drawback is that the haptic device is three degree-of-freedom and only one haptic device is supported. Fousek [79] also presented a state-space haptic force pre-computation and approximation method based on the radial-basis function (RBF). The RBF interpolation could improve the accuracy of the approximation during the haptic interaction. But the limitation is that the RBF does not support dynamic virtual environment and deformable model. Liu [80] proposed a proxy position prediction method using the geometric patch to provide smooth haptic force. The patch is calculated from the real-time contact region prediction method. The continuous force feedback could be generated on the boundaries of the rigid model. However, this method also only works for static rigid model with 3-DOF haptic device. It cannot support force and torque interpolation of object-based 6-DOF haptic rendering.

Figure 3.6: Forcegrid data structure for haptic force interpolation [78]. (a) A virtual surgical tool located in the rectangle workspace $\mathbf{W}$. The operation point of the virtual tool is $\mathbf{P}$. (b) The force vectors $\mathbf{F}_v$ in the workspace. The haptic force of the virtual tool can be interpolated from the neighboring force vectors.

3.5 Benchmarks

To test the performance of haptic rendering algorithms, we use some standard benchmarks for evaluation. The peg-in-hole benchmark is a classic assembly problem, it has been implemented for haptic rendering in [81]. The Stanford bunny benchmark developed by [82] for graphic rendering also has been used for evaluation of haptic rendering in [81, 83]. A standard haptic data set is introduced in [84]. It collected the haptic force and position data used for haptic rendering evaluation. In this section, we introduce these benchmarks and how they are used in haptic rendering experiments.

- **Peg-in-hole Benchmark**

The stability of the haptic interaction can be evaluated in a classical case using the peg-in-a-hole benchmark (180 triangles for peg and 176 triangles for hole), shown in Fig. 3.7. Although it is composed with simple geometries, it is a challenging task to provide a stable haptic feedback during the insertion of the peg into the hole [15]. Fig. 3.7 shows several frames during a haptic interaction, they are as follows:

- 1) Sliding the tip of the peg on the top side of the box.

- 2) Laying the peg on the top side of the box and sliding it on the box.
- 3) Pushing on the front side of the box.
- 4) Inserting the peg into the hole.
- 5) Leaving the hole model.

In our experiments, the proposed algorithms are run using the peg-in-hole benchmark to calculate the interaction force and torque. The force performance is evaluated from the touch of the surface (1-2 steps). The torque performance is evaluated from the step 3 where one end of the peg is touching the wall of the box. In step 4, we evaluate both force and torque performance when the peg is inserted into the hole.

Figure 3.7: The peg-in-hole haptic rendering benchmark. Left: The models used in the peg-in-a-hole benchmark. Right: A haptic interaction in the peg-in-a-hole benchmark.

- **Stanford bunny benchmark**

The Stanford bunny benchmark involves two Stanford bunnies (20898 triangles, shown in Fig. 3.8.) The Stanford bunny model is a famous benchmark developed by Greg Turk and Marc Levoy [82], it has been widely used in graphic rendering and evaluation. In haptic rendering, we use the Stanford bunny model to test the haptic performance for large number of contact points.

During haptic manipulation, one bunny is set to be static, and the user controls another bunny as a virtual tool to touch the static one. There are a large number of contact

points generated during the haptic interaction. We evaluate the stability of the haptic force and torque feedback during the contact of the body parts and ear parts.

Figure 3.8: Haptic interaction between two Stanford Bunnies [81]. Each bunny model contains 20898 triangles.

- **Duck Benchmark**

Ruffaldi et al. [84] developed a haptic rendering evaluation system which includes a standard haptic force and position data set. It solves two problems in haptic evaluation. On one hand, it collected standard position data sets of the haptic device to solve the problem of inconsistency of hand manipulation. On the other hand, the experiment force information is recorded from the sensor to provide standard force data. This data set can be used for the accuracy evaluation.

The pipeline of the haptic algorithm evaluation system is shown in Fig. 3.9. The duck model is scanned and loaded into the system as a 3D model. The trajectory of the haptic device is recorded as Out-trajectory. Finally, the calculated forces and trajectory can be used to compare with the physically-scanned forces and the original trajectory.

Figure 3.9: The pipeline of the haptic algorithm evaluation system [84].

In this evaluation, the user controls the haptic device to touch a duck model (with 4,212 triangles, shown in Fig. 3.10). The data set of the trajectories and haptic forces can be downloaded from internet. We use this benchmark for evaluation of the performance of the prediction algorithms in Chapter 6. During the evaluation, we read the data set of the rendered forces. At the same time, the prediction algorithm predicts the next haptic force and applies the interpolation algorithm to calculate the predicted forces in haptic thread for haptic device. Finally, we compare the calculated forces with the data set of the rendered forces to get the accuracy of the algorithms.

Figure 3.10: The duck model used in the standard haptic evaluation system. (a) The duck model (4,212 triangles) used for haptic rendering evaluation. (b) The scanned path of the haptic probe on the surface of the duck model [84].

3.6 Summary

In this chapter, the related works on 3-DOF and 6-DOF haptic rendering algorithms, prediction algorithms, and existing benchmarks used for haptic evaluation are reviewed. The classification of the existing haptic rendering algorithms is presented. Then, based on the classification, point-based/3-DOF haptic rendering algorithms and object-based/6-DOF haptic rendering algorithms were reviewed. The related technologies that include collision detection, penetration depth computation and contact force computation were introduced. In addition, for virtual coupling based haptic rendering, the dynamic virtual coupling and static virtual coupling were discussed. Besides, the comparison of different haptic rendering algorithms was given. Furthermore, the prediction algorithms used in haptic rendering were reviewed. Finally, the different benchmarks (peg-in-hole, Stanford bunny, and duck benchmark) that are used for haptic rendering evaluation were described.

Chapter 4

Stable Dynamic Haptic Rendering Algorithm

In this chapter, we introduce the proposed stable dynamic haptic rendering algorithm based on virtual coupling. The main contribution of the stable dynamic algorithm is that it can overcome the “buzzing” problem appeared in the haptic manipulation when a virtual tool has small mass values. In the haptic rendering process, we consider the dynamic property such as rotation inertia in each haptic frame. A nonlinear force/torque algorithm is proposed to calculate the haptic interaction when the collision happens between the virtual tool and virtual objects. The force/torque magnitude is automatically saturated to the maximum force/torque value of the haptic device. In Section 4.1, we describe the haptic force and torque calculation. Section 4.2 presents the pipeline of the stable dynamic rendering system. The experiments implemented with standard benchmarks are described in Section 4.3. Section 4.4 lists the comparison results of haptic rendering algorithms.

4.1 Stable Dynamic Algorithm based on Virtual Coupling

In the virtual coupling based haptic rendering [21], the QSA approach was used to

improve the stability and speed of force display. However, the physical property such as inertia of the virtual tool was not considered. There would be a “buzzing” problem in force display, when the virtual tool has the mass physical property. To eliminate this instability problem, we propose a stable dynamic algorithm based on virtual coupling using the asymptotic increment of force magnitude to maintain stable haptic rendering for virtual tools with small mass values. It can be used for applications when a virtual tool has small mass value and high stability is needed, for example, in surgery simulation.

In the virtual coupling based haptic rendering algorithms, the haptic handle that connected to the virtual tool through a spring-damper connection controls the position and orientation of the virtual tool, as shown in Fig. 4.1. If any displacement happens to the linear/rotational virtual spring, the system will generate the force/torque on the virtual tool and the opposite force/torque on the haptic handle. The torque and rotation motions are generated by the displacement of a spiral spring at the center of the virtual tool. In virtual coupling methods, a spring force is proportional to the displacement of the linear virtual spring, and a spring torque is proportional to the angle of rotation of the rotational virtual spring. In practice, if large displacement is generated in contact between the virtual tool and the virtual object, the interaction force would increase over the maximum force limit of the haptic device. In such case, the performance of the haptic feedback would be instable, and the haptic device could be damaged. Therefore, to improve the efficiency of virtual coupling, the stiffness parameter of the virtual coupling needs to be updated at the frame-by-frame basis depending on the spring displacements at each time step. Furthermore, as the displacement and stiffness increase, the force/torque output values would be limited by saturation values.

For the spring-damper connection model shown in Fig. 4.1, assume that the haptic handle and the virtual tool are connected in the following manner. The center of the haptic handle $\mathbf{p}_{HIP}$ is connected with the center of the virtual tool $\mathbf{p}_{tool}$ through a spring-damper link, and the haptic handle main axis $\mathbf{A}_{HIP}$ is connected with one of the principal axes of the virtual tool $\mathbf{A}_{tool}$ using spiral spring link. Although there is no collision detection directly happened between the haptic handle and the virtual environment, the interaction force could be calculated from this spring-damper

connection represented by parameters k_{trans} and b_{trans} . The force $\mathbf{F}_{Haptic}$ would be applied to the haptic handle, and the opposite force would be applied to the virtual tool at the same time.

Figure 4.1: Virtual coupling model for stable dynamic haptic rendering algorithm.

4.1.1 Haptic Force

At the beginning of haptic rendering process, the system reads the position of the haptic handle $\mathbf{p}_{HIP}$, the velocity of haptic handle $\mathbf{v}_{HIP}$, the position of virtual tool $\mathbf{p}_{tool}$ and the velocity of virtual tool $\mathbf{v}_{tool}$. The haptic interaction force $\mathbf{F}_{Haptic}$ can be described as following equation:

$$\mathbf{F}_{Haptic} = -k_{trans}(\mathbf{p}_{HIP} - \mathbf{p}_{tool}) + b_{trans}(\mathbf{v}_{HIP} - \mathbf{v}_{tool}), \quad (4.1)$$

where k_{trans} is spring stiffness for translation and b_{trans} is damping stiffness for translation.

Figure 4.2: The Displacement-Force relationship in the stable dynamic algorithm based on virtual coupling.

In the algorithm, we propose to use an asymptotic increment in initial region from 0 to d_{H1} , as shown in Fig. 4.2. When the displacement is small, the algorithm uses an asymptotic increment of the force magnitude to eliminate the “buzzing” problem caused by the mass property assigned to the virtual tool. As the displacement increases, the force grows exponentially in the region from d_{H1} to d_{H2} . When the force magnitude is large enough, it begins to saturate to its maximum force magnitude $f_{\max}$ in region from d_{H2} to d_{H3} .

At the beginning of each haptic rendering frame, we assume that the displacement of haptic spring is $\mathbf{d}_H$

$$\mathbf{d}_H = \mathbf{p}_{HIP} - \mathbf{p}_{tool}. \quad (4.2)$$

The maximum force magnitude of a haptic device is defined as $f_{\max}$ which could be read from devices configurations. As the manipulation of the haptic handle is done in a slow movement, the difference of velocity between the virtual tool $\mathbf{v}_{tool}$ and the haptic handle $\mathbf{v}_{HIP}$ is very small in (4.1). Then, the haptic force can be simulated with the

following equation:

$$f(\mathbf{d}_H) = \mathbf{f}_{\max} \cdot (1 - e^{-\|\mathbf{d}_H\| \cdot \mathbf{f}_{\max}^2 \|\mathbf{d}_H\|}). \quad (4.3)$$

At first, the proposed function provides the asymptotic increment on the haptic force calculation when the magnitude of displacement d_H is between $0 \leq d_H < d_{H1}$, shown in Fig. 4.2. This force calculation is used to filter the buzzing vibrations. When the displacement d_H between the virtual tool and the virtual object increases to the range $d_{H1} \leq d_H < d_{H2}$, the haptic force magnitude would grow exponentially according to the equation (4.3). In this situation, the user could clearly feel the interaction force between the virtual tool and the virtual object. Therefore, the transparency of haptic feedback could be improved comparing with the linear virtual coupling algorithms. Finally, when the user keeps pushing the virtual tool against the virtual objects, the haptic interaction force would approach the maximum force value. If we use a linear threshold on the maximum force value, the user would feel no difference when he/she pushes further. To avoid this, when the displacement d_H keeps growing to the range $d_{H2} \leq d_H < d_{H3}$, a non-linear force calculation would make the haptic force saturate to the maximum force value.

Function $k_d(\mathbf{d}_H)$ is used to calculate the change of haptic spring stiffness k_d according to the variation of the displacement as follows:

$$k_d(\mathbf{d}_H) = f(\mathbf{d}_H) / \|\mathbf{d}_H\|. \quad (4.4)$$

In (4.4), the stiffness parameter k_d should be always positive to generate an attractive haptic force. The exponential growth of the magnitude of the haptic force is applied when the displacement between the haptic handle and the virtual tool becomes larger [21]. The function $k_d(\mathbf{d}_H)$ is applied only when the user pushes the virtual tool into the contact with other virtual objects. When the virtual tool recedes from the contact, the constant stiffness parameter k_{trans} would be used for k_d . To determine whether the virtual tool is approaching virtual object or receding from the prior contact, we use the orientation

vector of the haptic device $\mathbf{v}_{HIP}$ and displacement vector $\mathbf{d}_H$ to identify status of the displacement. Thus, the virtual spring stiffness k_d can be determined as follows:

$$\begin{cases} k_d = k_{trans}, & \text{if } \mathbf{d}_H \cdot \mathbf{v}_{HIP} < 0 \\ k_d = k_d(\mathbf{d}_H), & \text{if } \mathbf{d}_H \cdot \mathbf{v}_{HIP} \geq 0 \end{cases} \quad (4.5)$$

Assuming that the damping stiffness parameter $b_d = b_{trans}$ and k_d is defined by (4.5). The final function for the haptic force calculation following (4.1), (4.2), and (4.3) is expressed as follows:

$$\mathbf{F}_{haptic} = -k_d \mathbf{d}_H + b_d (\mathbf{v}_{HIP} - \mathbf{v}_{tool}). \quad (4.6)$$

4.1.2 Haptic Torque

In the calculation of rotation torque, the rotational stiffness k_{rot} is represented as a function of offset rotation vector $\boldsymbol{\theta}_H$ (including rotation magnitude and rotation axis). The equation is described as follows:

$$k_{rot}(\boldsymbol{\theta}_H) = \frac{\mathbf{t}_{max} \cdot (1 - e^{-\|\boldsymbol{\theta}_H\| \cdot \mathbf{t}_{max}^2 \|\boldsymbol{\theta}_H\|})}{\|\boldsymbol{\theta}_H\|}, \quad (4.7)$$

where $\mathbf{t}_{max}$ is the maximum torque of haptic device.

The equation of the haptic torque calculation $\mathbf{T}_{haptic}$ could be described as follows:

$$\mathbf{T}_{haptic} = k_{rot}(\boldsymbol{\theta}_H) \boldsymbol{\theta}_H + b_{rot} (\boldsymbol{\omega}_{HIP} - \boldsymbol{\omega}_{tool}), \quad (4.8)$$

where $\boldsymbol{\omega}_{HIP}$ and $\boldsymbol{\omega}_{tool}$ are the angular velocity of haptic handle and the angular velocity of virtual tool. We also use an equation similar to (4.5) in torque calculation to calculate k_{rot} .

4.2 Pipeline of Stable Dynamic Rendering

In this haptic rendering system, the virtual objects and virtual environment are loaded in CHAI library that can provide both graphic rendering and haptic interface. CHAI library integrated an Open Dynamic Engine (ODE) which can efficiently compute collision detection between multiple moving objects [85]. However, the CHAI library only provides the 3-DOF haptic rendering algorithm for the point-based haptic rendering. Therefore, we integrated our 6-DOF stable dynamic algorithm based on virtual coupling in the CHAI library to provide both force and torque feedback, and details are introduced in Section 8.1.

The haptic rendering pipeline is shown in Fig. 4.3. In each haptic rendering frame, the algorithm reads the position and orientation parameters of the haptic handle and the virtual tool. The ODE detects collisions and calculates dynamic movement when there are contacts between the virtual tool and virtual objects. In the virtual coupling part, the haptic handle and the virtual tool are connected through the stable dynamic virtual coupling link. The algorithm calculates the haptic force/torque of virtual coupling from the linear and rotational displacement between the haptic handle and the virtual tool. After force/torque saturation, the final force/torque is applied to the user through haptic device.

Figure 4.3: Haptic rendering pipeline for stable dynamic rendering algorithm.

4.3 Experiment Results on Benchmarks

In this section, the proposed stable dynamic haptic rendering algorithm is implemented and compared with other algorithms in our real-time haptic rendering system. Two benchmarks were implemented for testing and analysis. For each benchmark, we use the same haptic manipulation path to test different virtual coupling algorithms. The comparison of the algorithms includes force magnitude, torque magnitude, computation time, and number of contact points. From the graphs of force/torque magnitude, the stability of haptic feedback can be estimated. Similarly, number of contact points and computation time of each haptic frame can be used to evaluate the rendering speed of algorithm.

The experiments were performed on a Windows XP Professional PC with Intel® Core™2 Quad Q9400 2.66 GHz CPU and 3.25GB memory. A PHANToM Premium 1.5/6DOF haptic device from SensAble Technologies was used in the experiment to provide 6-DOF force and torque feedback. Table 4.1 shows complexity of the benchmark models which were used to evaluate the performance of our stable dynamic algorithm implemented in the real-time haptic rendering system.

Table 4.1: Complexity of models used in the benchmarks.

Model		Polygons
Peg-in-hole Benchmark	Peg	180
	Box	176
Stanford Bunny Benchmark		20898

4.3.1 Peg-in-hole Benchmark

In the experiment, we use the peg-in-hole benchmark (the peg model with 180 triangles as a virtual tool and the box-with-hole model with 176 triangles as a virtual object) to evaluate the stability of these haptic rendering algorithms. The virtual tool is assigned

with mass value 0.02 kg in the experiment.

The experiment process is similar with the tasks in [56]. Fig. 4.4 shows six key configurations of the peg-in-hole model used during the haptic interaction.

Figure 4.4: Key configurations of the peg-in-hole benchmark tested during haptic rendering simulation.

From Fig. 4.5 to Fig. 4.7, the results of testing three algorithms such as original spring-damper virtual coupling, quasi-static approximation algorithm and the proposed stable dynamic algorithm with peg-in-hole benchmark are shown. In these three figures, the first two rows report changes of the force and torque magnitudes during the six key peg-in-hole haptic interactions. The third row shows computational time of collision detection and force/torque calculation in haptic rendering process. The forth row shows the number of contact points in each haptic frame.

The original spring-damper algorithm [48] can provide relatively stable performance for virtual objects with low mass. However, two problems could affect the haptic rendering performance. First, there would be vibration when the force changes quickly from large force magnitude to small force magnitude as shown in area (e) of the force magnitude graph in Fig. 4.5. Second, the speed of the force growth is not fast enough to generate immediately the force feedback when the collision happens, as it is shown in Fig. 4.6, area (b).

Comparing the force magnitudes in Fig. 4.5, area (b) and in Fig. 4.6, area (c), we could see that the QSA method has a faster speed of the force growth than the original spring-damper algorithm. In area (c) of Fig. 4.6, the vibration problem in force magnitude has been solved when the force magnitude has a sudden change from large value to small value. However, for the virtual tool with mass property, the QSA method would suffer a

“buzzing” problem when the user manipulates the virtual tool. The “buzzing” problem could be seen as small continuous changes of the force magnitude in Fig. 4.6, area (e). It shows a low-magnitude vibration with a high frequency.

The experiment implemented with our stable dynamic algorithm based on virtual coupling is shown in Fig. 4.7. From the force magnitude graph of Fig. 4.7, area (b), we could see that our algorithm could provide stable performance for sudden large change in force magnitude. In addition, when the user is manipulating the virtual tool with mass property, the algorithm could avoid the “buzzing” problem as shown in force magnitude graph of Fig. 4.7. Furthermore, when collision happens between virtual tool and virtual objects, our algorithm could provide fast force growth as shown in the force magnitude graph in Fig. 4.7, area (b).

Figure 4.5: The experiment result of the original Spring-damper virtual coupling algorithm.

Figure 4.6: The experiment result of the quasi-static approximation algorithm.

Figure 4.7: The experiment result of the stable dynamic algorithms.

4.3.2 Stanford Bunnies Benchmark

In the second benchmark experiment, two Stanford bunny models were used for haptic rendering as shown in Fig. 4.8. The left bunny is manipulated by the user to have contacts with the white bunny which was set static in the center of the virtual environment. Each model contains 20,898 polygons and is considered to be a complex benchmark for haptic rendering experiment.

Figure 4.8: Stanford bunny benchmark used for haptic rendering simulation. Each Stanford bunny contains 20898 polygons.

Fig. 4.9 reports the haptic rendering performance of Stanford bunny benchmark with our stable dynamic algorithm. Although the bunny model is much more complex than peg-in-hole model, the haptic rendering system also could provide continuous force and torque feedback to the user. The vibration and buzzing problems were eliminated with the proposed stable dynamic algorithm.

Figure 4.9: Performance of the proposed algorithm in Stanford bunny benchmark with PHANToM haptic device.

Because of the complexity of the models, the computation time and the number of contact points have increased in the Stanford bunny benchmark compared with the peg-in-hole benchmark as shown in Fig. 4.9. At the beginning part of haptic manipulation process, the number of contact points is limited. When two bunnies have close contact, especially with the ear part, the number of contact points would increase rapidly as well as computation time for the current haptic frame. To save computation time and generate a stable haptic rendering, we set a limitation for the number of contact points (150 points) used for collision calculation in each haptic rendering frame. In the Fig. 4.9, it is shown that when the number of contact points increased to 150 the computation time increased from 0.002s to 0.01s correspondingly.

4.4 Comparison of Haptic Forces

The analysis results of three algorithms are shown in Fig. 4.10. All algorithms were tested on the peg-in-hole benchmark in the same interaction process when the user lays the peg on the top side of the box, as shown in Fig. 4.4(b). Comparison of three haptic rendering algorithms during a deep contact between the virtual tool and the virtual object is shown in Fig. 4.10(a), and the zoomed view of the beginning of the contact between the virtual tool and virtual object with the buzzing is shown in Fig. 4.10(b).

Fig. 4.10(b) shows clear buzzing in the force magnitude of the QSA algorithm (in green) before 150th haptic frame. At the beginning of the contact, around the 200th haptic frame, the force magnitude of the proposed stable dynamic algorithm (in red) grows slower in comparison with other two algorithms. This slow speed of the force growth can eliminate the “buzzing” vibrations in the haptic feedback. In the following haptic frames, the speed of the force growth of the proposed algorithm increases faster than of other two algorithms and saturates to the maximum force magnitude (4N). From Fig. 4.10(a), it is seen that the force magnitude of the spring-damper algorithm has sudden changes when the force magnitude increases to the maximum force value at 700th haptic frame and then decreases from the maximum force value at 2100th haptic frame.

4.5 Discussion

In this chapter, a stable dynamic haptic rendering algorithm based on virtual coupling for 6-DOF haptic rendering is proposed. The purpose of the algorithm is to solve the “buzzing” problem appeared during the haptic rendering with the virtual tool that has small mass values. When the displacement is small, the stable dynamic algorithm uses a non-linear function to calculate haptic force/torque to make sure stable haptic feedback implemented without “buzzing”. On the other hand, in a deep contact, the stable dynamic algorithm saturates the force/torque magnitude to the maximum force/torque value of the haptic device.

(a)

(b)

Figure 4.10: The comparison of three haptic rendering algorithms during a deep contact between the virtual tool and the virtual object. (a) The changes of the force magnitude during a deep contact between the virtual tool and virtual object. (b) A zoomed view of the buzzing at the beginning of the contact between the virtual tool and virtual object.

The stable dynamic haptic rendering algorithm was tested on the standard benchmarks (peg-in-hole benchmark with 180/176 polygons, and Stanford bunny benchmark with 20898 polygons), and compared with other algorithms. The experiment result shows that the proposed algorithm can solve the “buzzing” caused by the mass physical property assigned to the virtual tool, and the force/torque can saturate to their maximum values during the contact.

Chapter 5

Adaptive Haptic Rendering Algorithm

In this chapter, we propose an adaptive algorithm based on virtual coupling for haptic rendering. The algorithm can provide accurate haptic rendering when virtual tools with different mass values including large mass values are needed. The proposed adaptive algorithm solves the displacement problem of the haptic manipulation caused by mass values of the virtual tools. It can adjust parameters of the virtual coupling to overcome the problem of displacement and keeps an accurate haptic manipulation in dynamic virtual environments. In addition, when the large contact force is generated, our algorithm saturates the force/torque values asymptotically to the maximum force/torques magnitude of the haptic device. In Section 5.1, the force/torque calculation is presented. Section 5.2 describes a saturation algorithm to make sure the generated force magnitude does not exceed the limitation. The experiment results and analysis are described in Section 5.3 and Section 5.4 is discussion.

5.1 Adaptive Force/Torque Calculation

In this section, we describe an adaptive algorithm for 6-DOF haptic rendering. The proposed algorithm can be used when simulated virtual tools have different mass values. Most of the existing haptic rendering algorithms can be used only when virtual tools have small mass value [26, 48] or no mass value [49, 58]. But for realistic simulation in medicine or other applications, the accurate mass simulation is essential for haptic

rendering. For virtual tools with different mass values, our adaptive algorithm can automatically adjust parameters of the virtual coupling to maintain an accurate and stable haptic simulation.

In virtual coupling algorithms, first, the system reads the position of the haptic handle $\mathbf{p}_{HIP}$, velocity of haptic handle $\mathbf{v}_{HIP}$, position of virtual tool $\mathbf{p}_{tool}$ and velocity of virtual tool $\mathbf{v}_{tool}$. The haptic interaction force $\mathbf{F}_{hapitc}$ using the spring-damper model is defined as follows:

$$\mathbf{F}_{hapitc} = -k_{trans}(\mathbf{p}_{HIP} - \mathbf{p}_{tool}) + b_{trans}(\mathbf{v}_{HIP} - \mathbf{v}_{tool}), \quad (5.1)$$

where k_{trans} is spring stiffness for translation and b_{trans} is damping stiffness for translation. These parameters can be empirically set as constant values, and they can provide stable haptic rendering only for one mass value of the virtual tool [48].

To find the appropriate parameters of virtual coupling for a haptic device, when virtual tools can have different mass values, at first, we do a series of experiments for different mass values m . For each mass value, we find a best stiffness parameter for the virtual coupling. Second, we use a least square method to get the polynomial from the data set of the haptic simulation. The details of using the least square method to calculate the coefficients of the virtual coupling model will be explained in Section 5.3.1. So we have got a function of k_t from mass value m for the particular haptic device. Because the inertia and configuration of haptic devices are not the same, the function may be different for different haptic devices. Third, based on the stiffness function of k_t and mass value m , we do a series tests on the parameter of damping b_t . Finally, the polynomial function for damping coefficient b_t is calculated by using the least square method

$$k_t = f(m), \quad (5.2)$$

$$b_t = g(k_t, m). \quad (5.3)$$

In each haptic frame, the system reads the position of the haptic handle $\mathbf{p}_{HIP}$, velocity of haptic handle $\mathbf{v}_{HIP}$, position of virtual tool $\mathbf{p}_{tool}$ and velocity of virtual tool $\mathbf{v}_{tool}$. From

the configurations of the haptic device and the virtual tool, the displacement $\mathbf{d}$ and the difference of velocity $\mathbf{v}$ is calculated and used as the input of the virtual coupling in each haptic rendering frame

$$\mathbf{d} = \mathbf{p}_{HIP} - \mathbf{p}_{tool}, \quad (5.4)$$

$$\mathbf{v} = \mathbf{v}_{HIP} - \mathbf{v}_{tool}. \quad (5.5)$$

The haptic interaction force $\mathbf{F}_{haptic}$ based on virtual coupling and mass value is described in the following equation:

$$\mathbf{F}_{haptic}(\mathbf{d}, \mathbf{v}) = -k_t \mathbf{d} + b_t \mathbf{v}. \quad (5.6)$$

In the calculation of rotation torque, we use the similar process to get the function of the spiral spring stiffness k_{rot} and damping b_{rot} from the inertia moment I

$$k_{rot} = d(m), b_{rot} = h(k_{rot}, I). \quad (5.7)$$

The rotational vector $\mathbf{u}$ (including rotation magnitude and rotation axis) represents the offset between the orientation of the haptic device $\mathbf{u}_{HIP}$ and orientation of the virtual tool $\mathbf{u}_{tool}$

$$\mathbf{u} = \mathbf{u}_{HIP} - \mathbf{u}_{tool}, \mathbf{w} = \mathbf{w}_{HIP} - \mathbf{w}_{tool}, \quad (5.8)$$

where $\mathbf{w}$ represents the difference between angular velocity of the haptic device $\mathbf{w}_{HIP}$ and the virtual tool $\mathbf{w}_{tool}$. The haptic $\mathbf{T}_{haptic}$ torque is calculated from the following equation:

$$\mathbf{T}_{haptic}(\mathbf{u}, \mathbf{w}) = -k_{rot} \mathbf{u} + b_{rot} \mathbf{w}. \quad (5.9)$$

5.2 Force Saturation Algorithm

The hardware limitation of the haptic device should be considered in the force display of

haptic rendering. To improve stability of the haptic rendering system, the virtual coupling force should always be smaller than the maximum force magnitude of the haptic device. For example, during the deep contact between the virtual tool and the virtual object the calculated force magnitude can exceed the haptic device maximum force value. A simple solution is to set a threshold value smaller than the maximum force value of the haptic device. When the force magnitude exceeds the threshold value, the magnitude of the coupling force is kept in a constant value to make sure that the force feedback is controlled within the limitation of the haptic device. However, in this approach, if the user keeps exerting force or pushing further virtual tool against the virtual object, he/she could not feel the change of the force from the force feedback.

To control the coupling force within the maximum of haptic force of the haptic device and to display a continuous force growth, during the interaction of the virtual tool and the virtual object, the non-linear saturation algorithms need to be implemented to make sure that the haptic force approaches to the maximum force value asymptotically.

Wan and McNeely [49] proposed a saturation method using exponential function to saturate haptic force according to the displacement between the virtual tool and the haptic device. As the displacement increases, the force asymptotically approaches the maximum force value of the haptic device. In this saturation method, the stiffness value decays exponentially with the increment of the displacement. Otaduy and Lin [26] proposed a non-linear virtual coupling algorithm in which the saturation method uses a spline function for force saturation. The spring stiffness is considered as a nonlinear function of displacement between the virtual tool and the haptic device. For small displacement, the stiffness is a constant value, and haptic force increases linearly. For large displacement, the change of haptic force follows a cubic interpolation function. When the haptic force exceeds the maximum force value of the haptic device, the haptic force is kept at the maximum force value.

However, there are some limitations of these saturation methods. First, these saturation methods calculate the force only based on the displacement between the virtual tool and the haptic device. The damping force which is calculated from the difference of the velocities of virtual tool and haptic device is neglected. However, if the virtual coupling is used in a dynamic virtual environment where the virtual tool has physical

properties like mass, the final coupling force is also affected by damping part. When the user moves the virtual tool fast or the virtual tool is sliding along the contact surface quickly, the difference of velocities can be large enough to generate a significant damping force in the calculation of the virtual coupling. Therefore, if the saturation methods are only calculated based on the displacement, it is possible that the haptic force exceeds the limitation of the haptic device. Second, it is hard to control the saturation process. For the saturation methods based on exponential function, if the stiffness parameter changes, the start point and speed of saturation would change. The user cannot control the saturation easily. For the saturation methods based on interpolation, if the stiffness changes according to the mass of virtual tool, the interpolation function needs to be recalculated each time.

The force saturation method of the stable dynamic haptic rendering algorithm proposed and described in Chapter 4 is similar with the method of Wan and McNeely [49]. The advantage of the proposed saturation method is a fast calculation speed as the calculation of the saturated force is integrated into the force calculation of the virtual coupling. In addition, the proposed force saturation method in Chapter 4 supports haptic rendering of the virtual tool with mass value. However, if there are different virtual tools with different physical properties, the coefficients of the virtual coupling model need to be adjusted to generate stable haptic rendering. It takes more time to calculate new coefficients that meet requirements of both virtual coupling and force saturation.

To overcome such limitations, in the Chapter 5, we proposed and implemented a new force saturation method, in which the calculation of the force saturation is separated from the virtual coupling. If the user needs to manipulate different virtual tools with different mass values, the spring and damping coefficients of the virtual coupling will be adjusted automatically by the adaptive haptic rendering algorithm. However, unlike the force saturation method proposed in Chapter 4, the new method proposed in Chapter 5 needs no recalculation, and its function is not affected by changes of the coefficients of the virtual coupling.

The saturation algorithm includes both a linear part and a non-linear part. The linear part is used when the magnitude of the coupling force is not large and under the best working state of the haptic device. The non-linear part is used when the coupling force is

approaching to the limitation of the haptic device. To realize smooth force saturation, the non-linear function must be continuous in the first order derivative with the linear function. Therefore, the user can feel a gradual growth of coupling force during the deep contact of virtual tool and the object or during the fast movement of the virtual tool. The Hermite interpolation also can be used for saturation. But, when the start point of the force saturation changes, it requires recalculation of the parameters of the cubic function at each time. In the proposed force saturation method, ε is used to control the start point of the force saturation. If the start point changes, there are no additional calculations of parameters. The saturation method of the final virtual coupling force is as follows:

$$H(\mathbf{F}_{haptic}) = \begin{cases} \mathbf{F}_{haptic} & \text{if } 0 < \mathbf{F}_{haptic} \leq (1 - \varepsilon)\mathbf{F}_{\max} \\ \mathbf{F}_{haptic} \left(1 - \frac{\varepsilon\alpha}{\mathbf{F}_{haptic} - \beta}\right) & \text{if } (1 - \varepsilon)\mathbf{F}_{\max} < \mathbf{F}_{haptic} < \mathbf{F}_{\max} \end{cases}, \quad (5.10)$$

$$\alpha = \varepsilon\mathbf{F}_{\max}, \quad (5.11)$$

$$\beta = (1 - \varepsilon)\mathbf{F}_{\max}, \quad (5.12)$$

where ε is a parameter which controls the start point of the force saturation $(1 - \varepsilon)\mathbf{F}_{\max}$, as shown in Fig. 5.1. The value of ε can be set from 0.0 to 1.0 by the user according to the requirement of the applications. Before the magnitude of the virtual coupling force reaches the start point, the linear part keeps the haptic force the same as the virtual coupling force. As the force grows, when the force magnitude is greater than the start point $(1 - \varepsilon)\mathbf{F}_{\max}$, the non-linear equation in (5.10) is used to saturate the coupling force to the maximum force value of haptic device $\mathbf{F}_{\max}$. The final function $H(\mathbf{F}_{haptic})$ is continuous in the first order derivative to provide smooth force feedback.

Figure 5.1: The virtual coupling force saturated to the maximum force magnitude of the haptic device from $(1 - \varepsilon)\mathbf{F}_{\max}$.

Fig. 5.2 shows the simulation result of the virtual coupling force with different mass values. The virtual coupling force comes mainly from the spring force, because the difference of velocities is tiny during the close contact and the damping force is nearly zero. In this simulation, the $\mathbf{F}_{\max}$ is set to be 6.5N and the ε is set to be 0.2 which means the coupling force begins to saturate to the maximum force limitation when it reaches 80% of the maximum force magnitude of the haptic device. From Fig. 5.2, we can see that the virtual coupling force begins to saturate when the force magnitude reaches 5.2N. Moreover, four different mass values of the virtual tool (0.05kg, 0.1kg, 0.15kg and 0.2kg) are simulated to show how the force grows in close contacts. From the gradient of the coupling force, we can see that the stiffness parameter of the virtual coupling is adjusted

automatically according to the mass values. This adjustment can properly reduce the displacement during haptic manipulation. As the mass increases, the stiffness value becomes larger to avoid an obvious misalignment between the virtual tool and the haptic device.

Figure 5.2: The contact simulation for the force saturation. The virtual tool is tested with four mass values 0.05kg, 0.1kg, 0.15kg, and 0.2kg.

5.3 Experiment Results

In this section, the proposed stable dynamic haptic rendering algorithm is analyzed and compared with other algorithms using our real-time haptic rendering system. Two benchmarks were implemented for testing and analysis. For each benchmark, we use the same haptic manipulation path to test different virtual coupling algorithms. The following

parameters of the algorithms are analyzed: force magnitude, torque magnitude, computation time, and number of contact points. From graphs of the force magnitude changes, stability of haptic feedback can be estimated. The number of contact points and computation time of each haptic frame are used to evaluate the rendering speed of the algorithms.

The experiments were performed on a Windows PC with Intel Core2 Quad Q9400 2.66 GHz CPU. A PHANToM Premium 1.5/6DOF haptic device from SensAble Technologies was used in the experiment to provide 6-DOF force and torque feedback. Table 5.1 lists the complexity of the benchmark models used to evaluate performance of the stable dynamic algorithm. The benchmark models such as peg-in-hole and bunny models implemented in the real-time haptic rendering system are shown in Fig. 5.3.

In our system, virtual objects are implemented using CHAI library that provides both graphic rendering and haptic interface. Different haptic devices can be used in a virtual environment through the haptic interface. CHAI library can be integrated with Open Dynamic Engine (ODE) where collision detection between multiple moving objects could be efficiently computed [85]. A multi-rate rendering approach is used to improve the rendering speed. We integrated our 6-DOF haptic rendering algorithm based on virtual coupling in the CHAI library to provide both force and torque feedback.

Table 5.1: Complexity of models used in the benchmarks.

Model	Polygons
Peg	180
Box with hole	176
Stanford Bunny	20898

Figure 5.3: Three benchmark models used in the haptic rendering system: (a) peg model, (b) box with hole model, and (c) bunny model.

In the haptic rendering pipeline, the CHAI library reads the position and orientation parameters of the haptic handle and the virtual tool. The ODE detects collisions between objects and calculates dynamic movement when there are contacts between the virtual tool and virtual objects. In the virtual coupling part of the haptic rendering pipeline, the haptic handle and the virtual tool are connected through the virtual coupling link.

5.3.1 Determination of Virtual Coupling Parameters

In virtual environments, different virtual tools can have different mass values changing from light to heavy. For each mass value of the virtual tools, the virtual coupling parameters must be changed to realize stable and accurate haptic rendering. For each kind of haptic devices, we need to find the relationship between the mass values and the virtual coupling parameters k_t and b_t .

Table 5.2: The spring stiffness for different mass values.

Exp. No	Mass(kg)	Stiffness(N/m)
1	0.01	19
2	0.02	18.5
3	0.03	20
4	0.04	20
5	0.05	21
6	0.06	23
7	0.07	24
8	0.08	25
9	0.09	26.5
10	0.1	29
11	0.12	29.5
12	0.14	33
13	0.16	35.5
14	0.18	37
15	0.2	39.5

Figure 5.4: The spring stiffness for different mass values.

At first, the peg-in-hole experiments are implemented with different mass values. For each mass value of the virtual tool, the best fitting spring parameter is found, as shown in Table 5.2. The least square method is used to get the polynomial function of the data. In Fig. 5.4, a linear function (in red) and a quadratic function (in green) are calculated to fit the experiment data of Table 5.2. The degree of such polynomial is determined for each new type of device. According to the experiment with our haptic device, the linear function could provide a good fitting performance. So we use the linear function instead of the quadratic function for the calculation of the spring parameter. According to the data of Table 5.2, the parameters a and b of the linear function are calculated using the least square method as follows:

$$a=116.53, \quad b=16.21. \quad (5.13)$$

The linear function of spring stiffness k_t from mass values of virtual tool m is defined as follows:

$$k_t = am + b = 116.53m + 16.21. \quad (5.14)$$

Table 5.3: The damping coefficients for different mass values.

Exp. No.	Stiffness(N/m)	Damping(Ns/m)
1	14	0.185
2	16	0.18
3	18	0.2
4	19	0.1975
5	20	0.205
6	21	0.2025
7	22	0.21
8	23	0.2225
9	24	0.235
10	25	0.23
11	27	0.195
12	29	0.2
13	30	0.2
14	31	0.205
15	33	0.205
16	35.5	0.185
17	37	0.18
18	39.5	0.19

Figure 5.5: The damping coefficient for different stiffness values.

If the spring parameter is identified, then we implement the experiments to find the relation between the mass value and the damping parameter. The best fitting damping parameters we received from the experiment are presented in Table 5.3. In Fig. 5.5, a linear function is used to fit the data of the damping parameters.

In the virtual coupling, the haptic force is calculated from the spring part and the damping part. During a contact between the virtual tool and virtual object, the magnitude of the damping part is very small comparing with the magnitude of the spring part. So the damping coefficient of the virtual coupling in both cases when it is assigned to a constant or calculated with a polynomial function does not influence on stability of the haptic rendering. To realize the faster calculation, we can use the mean of damping coefficient values to approximate the damping coefficient b_t as follows:

$$b_t = c = 0.2. \quad (5.15)$$

From the stiffness and damping coefficients, we get the virtual coupling function as follows:

$$\begin{aligned}\mathbf{F}_{haptic} &= -k_t \mathbf{d} + b_t \mathbf{v} = -(am + b) \mathbf{d} + c \mathbf{v} \\ &= -(116.53m + 16.21) \mathbf{d} + 0.2 \mathbf{v},\end{aligned}\quad (5.16)$$

where $\mathbf{d}$ is the displacement between the positions of the virtual tool and the haptic device. $\mathbf{v}$ is velocity difference between the virtual tool and the haptic device. $\mathbf{F}_{haptic}$ is the final haptic force used for haptic display.

5.3.2 Stability Analysis

Figure 5.6: Haptic rendering with the peg-in-hole benchmark. The mesh model represents the haptic device, and the green model represents the virtual tool. The displacement during manipulation of the peg inside the hole is shown.

In the first experiment, as shown in Fig. 5.6, we use the peg-in-hole benchmark to evaluate stability of the proposed adaptive algorithm. The evaluation method is similar to experiments in [81]. Fig. 5.7 shows the positions and orientations of peg in different steps. In the step (a) and step (b), we focus on the evaluation of the force feedback. In the step (c), we focus on the evaluation of the torque force feedback. In the step (d) and step (e), we focus on stability problems when the multiple contacts occur.

Figure 5.7: The five steps in the peg-in-hole benchmarks.

In this physical simulation, the mass values of the virtual tool changes from 0.05kg to 0.2kg as shown in Fig. 5.8. In each graph, the first two rows on the top show the force and torque feedback during the haptic manipulation, and the bottom two rows show the corresponding computation time and the number of contact points in each haptic frame.

When the user holds the haptic device without any contact, the peg is only subjected to gravity. So, in the step (a) of the peg-in-hole benchmark, the force magnitude values show the gravity forces which can be felt by the user. From Fig. 5.8, we can see the force magnitude increases when there is no contact because of the mass increase. When the virtual tool has contact with the box, the step (b) in Fig. 5.8 shows the stable force change and the step (c) shows the smooth torque change. It is the challenging problem for the algorithm to provide stable force/torque feedback while the user is inserting the peg into the hole. In step (d) of Fig. 5.8, although the stability is not as good as in the previous steps, the force magnitude does not display any obvious vibration or buzzing problems. The adaptive algorithm can automatically adjust the virtual coupling parameters to keep a stable haptic rendering for different mass values.

(a) Mass=0.05kg

(b) Mass=0.1kg

(c) Mass=0.15kg

(d) Mass=0.2kg

Figure 5.8: The haptic performance of the peg-in-hole benchmark with different mass values. (a) The mass value is 0.05kg; (b) the mass value is 0.1kg; (c) the mass value is 0.15kg; (d) the mass value is 0.2kg. The top two rows indicate the force and torque magnitudes, and the bottom two rows indicate the computation time and the number of contact points in each haptic frame. If there is no contact, the force magnitude is generated from the mass of the peg.

In the second benchmark experiment, two Stanford bunny models are used for haptic rendering as shown in Figure 5.9(a). The bunny benchmark is implemented to evaluate the stability when the haptic system renders complex models and has large number of contact points. In this benchmark, each model contains 20,898 polygons that are considered to be the complex benchmark for haptic rendering experiments. The right bunny (green) is manipulated by the user to touch the left bunny (blue) which is set to be static.

Figure 5.9: Haptic rendering with complex models: (a) Stanford bunny benchmark used for haptic rendering simulation. Each Stanford bunny contains 20898 polygons. (b) Performance of the proposed algorithm in Stanford bunny benchmark with PHANToM haptic device.

Fig. 5.9(b) reports the haptic rendering performance of our adaptive haptic rendering algorithm with the Stanford bunny model with mass value assigned to 0.1kg. Although the bunny model is more complex than the peg-in-hole model, the haptic rendering system still can provide continuous and stable force and torque feedback. At the beginning of the haptic manipulation process, the number of contact points is low. But when two bunnies have close contact, the number of contact points increases rapidly as well as the computation time. In the third and fourth rows of Fig. 5.9(b), from the 1000th haptic frame, the contact points number increases to almost 150, and the computation

time is more than 0.03s. The long computation time could make lag and jitter in force display. So, to make a continuous haptic feedback, we set a limit for the number of contact points. In our system, we can support maximum 150 contact points in each haptic rendering frame. In the Stanford bunny benchmark, the 150 contact points are enough for the accurate collision computation and smooth haptic feedback. If the contact points increase to more than 150, we only calculate the first 150 contact points for collision detection and collision response. As shown in the Fig. 5.9(b), the number of the contact points stop increase after 1000th haptic frame until the two bunnies leaving each other

5.3.3 Comparison of the Displacements

The main advantage of the proposed adaptive algorithm is that it allows keeping a small displacement of the virtual link for different mass values. We compare our algorithm with original spring-damper algorithm and QSA algorithm for four cases when the virtual tool is assigned mass values as 0.05kg, 0.1kg, 0.15kg and 0.2kg. For each haptic rendering algorithm, we implemented experiments with peg-in-hole model for the mass value range from 0.05kg to 0.2 kg. As the displacement problem is caused by mass value assigned to the virtual tool, and the displacement values do not depend on the complexity of the model, we use the results of experiments with peg-in-hole benchmark. The average displacement calculated during the whole haptic simulation process for each algorithm is shown in Table 5.4 and the corresponding graph is shown in Fig. 5.10. From the Fig. 5.10, first, we can see that the adaptive algorithm can keep a small displacement during the haptic simulations with peg-in-hole benchmark. When the mass value is 0.05kg, the difference in the displacement is small for all three algorithms. But, when the mass value is 0.2kg, the displacement of the spring-damper algorithm is 271% of the displacement of the adaptive algorithm. The displacement of the QSA algorithm is also 216% of the displacement of the adaptive algorithm. When the displacement of the virtual link has large value, the accuracy and stability of the haptic rendering is affected greatly. In addition, when the mass value of the virtual tool increases, the displacement of the adaptive algorithm grows very slowly comparing to other algorithms. When the mass value increases from 0.05kg to 0.2kg, the displacement of the QSA algorithm increases

dramatically, and the displacement of the spring-damper algorithm increases even more.

Table 5.4: The displacements comparison with different mass values.

Algorithms	Displacement (mass=0.05kg)	Displacement (mass=0.1kg)	Displacement (mass=0.15kg)	Displacement (mass=0.2kg)	Displacement (Average)
Spring-damper Algorithm	0.0596	0.0803	0.1084	0.1469	0.0988
QSA Algorithm	0.043	0.0674	0.0995	0.1171	0.08175
Adaptive Algorithm	0.0373	0.0462	0.0503	0.0542	0.047

Figure 5.10: The comparison of displacements when the algorithms render the virtual tool with different mass values (0.05kg, 0.1kg, 0.15kg, 0.2kg).

The Adaptive algorithm improves the haptic rendering ability for wide range mass values of virtual tools. It provides accurate haptic manipulation not only for tools with small mass value but also for various virtual tools with large mass values (10g - 1kg). For the different mass values, the adaptive algorithm would adaptively change the stiffness and damping parameters to calculate the virtual coupling force and torque.

The disadvantage of the adaptive algorithm is that, for a new type of haptic device, a series of calibration tests are required before the haptic rendering. The stable dynamic algorithm and the adaptive algorithm could be used for different applications according to the types of tools implemented. The stable dynamic algorithm could be implemented for simulations including the virtual tools with small mass values, like the surgery simulation with the knife and the art painting system with painting pens. The adaptive algorithm could be used for physical simulations which have virtual objects with different mass values, for example, the rehabilitation game with different objects to pick up or the assembling simulation with different mechanical parts.

5.4 Discussion

In this chapter, an adaptive 6-DOF haptic rendering algorithm based on virtual coupling is proposed. The objective is to automatically adjust parameters of the virtual coupling according to the mass values of virtual tools. For a virtual tool with large mass values, the adaptive algorithm can minimize the displacement of the virtual coupling to make the haptic manipulation more accurate. In addition, a new saturation method is proposed to saturate the force/torque values asymptotically to the maximum magnitude of the haptic device.

The algorithm was tested in our 6-DOF haptic rendering system with the standard benchmarks. The results of the experiments confirm that the proposed adaptive algorithm has a better performance than spring-damper algorithm and QSA algorithm if virtual tool is assigned with different mass values.

Chapter 6

A Prediction Algorithm for Haptic Rendering

The haptic force applied to the haptic device must be updated in a high rate about 1kHz to maintain a realistic haptic feedback. But, the computational process of the collision detection with multiple contact points, deformable models simulation, and the physically-based simulation in virtual environment are time consuming that needs the update rate of haptic force to be guaranteed. Although the improvement of the computational ability of the hardware can speed up the haptic rendering, there is always a contradiction between high requirement of advanced physical simulation and the current computation speed of the computer.

Some multi-thread haptic rendering is proposed to separate the haptic thread with the physical thread [25, 56, 86]. Although, these methods could improve the performance of the haptic rendering, it still has limitations and the update rate of 1 kHz is not guaranteed [56]. When there are complex models or deformable models, the physical simulation has a low update rate which causes discontinuous force output on the haptic device.

In this chapter, we propose a new prediction method for smooth haptic rendering to overcome the low update rate of the physical simulation of complex or deformable models. In Section 6.1, a prediction method combined with an interpolation method is proposed to calculate smooth haptic interaction force in the haptic thread which is independent of the physical simulation. The auto-regressive model is used to predict the

force value from the previous haptic force calculation. In addition, a spline function is proposed to interpolate force values for the smooth haptic force output. Section 6.2 describes the experiment and analysis of the algorithm accuracy and stability. In Section 6.3, a real-time coefficient calculation algorithm is used to update the parameters of auto-regressive model.

6.1 Methodology

6.1.1 Force/Torque Prediction and Interpolation

Different from previous force prediction methods, we propose to use a prediction method combined with an interpolation method to calculate smooth haptic interaction force. In our method, the interaction force is calculated from the virtual coupling model in the physical simulation thread. Then, the auto-regressive model uses the pre-calculated force values to predict the next force value in a fixed update rate. Finally, the interpolation algorithm uses the calculated interaction forces and the predicted force to interpolate smooth force values to output to the haptic device at 1 kHz rate in the haptic loop.

6.1.1.1 Prediction Algorithms

The auto-regressive (AR) model is used as an estimator to realize the force prediction. During the real time haptic rendering, the autoregressive model calculates the next force value by using a linear combination of the past force values and a white noise. To build the autoregressive model, we have to measure and record the forces during the haptic interaction and identify the parameters of the autoregressive model.

The virtual coupling model calculates the current interaction force $\mathbf{F}_t$ between the virtual tool and the virtual objects at current time t . So, $\mathbf{F}_t, \dots, \mathbf{F}_{t-p+1}$ (p is the order of AR model) are known and recorded during the haptic rendering process. These force values are used to predict the next haptic interaction force using the autoregressive model. The predicted interaction force can be forecasted with the following equation of order p at time t :

$$\mathbf{F}_{t+1|t}^p = \phi_1 \mathbf{F}_t + \phi_2 \mathbf{F}_{t-1} + \cdots + \phi_p \mathbf{F}_{t-p+1} + \varepsilon_{t+1|t}, \quad (6.1)$$

where $\phi_1, \dots, \phi_p$ are coefficients of AR model, $\varepsilon_{t+1|t}$ is a white noise series with zero mean. The details of the coefficients calculation of the regression model will be given later in section 6.3. $\mathbf{F}_{t+1|t}^p$ is the predicted interaction force value in the next physical simulation loop at the current time t .

Similarly, the corresponding torque values are calculated as follows:

$$\mathbf{T}_{t+1|t}^p = \varphi_1 \mathbf{T}_t + \varphi_2 \mathbf{T}_{t-1} + \cdots + \varphi_p \mathbf{T}_{t-p+1} + \varepsilon_{t+1|t}, \quad (6.2)$$

where $\varphi_1, \dots, \varphi_p$ are coefficients for torque predictions. $\mathbf{T}_t, \dots, \mathbf{T}_{t-p+1}$ and $\mathbf{T}_{t+1|t}^p$ are known torque values and the predicted next torque value.

The order of the auto-regressive model is determined according to the Final Prediction Error (FPE) criterion. The FPE criterion can measure the quality of the models with different orders based on the standard haptic data set [84] described in Chapter 3.5 “Benchmarks”. The FPE function is formed as follows:

$$FPE = V \left(\frac{1 + d / N}{1 - d / N} \right), \quad (6.3)$$

where d represents the number of parameters, N is a number of values in the estimated haptic force data set and V is a loss function.

For the AR prediction model, the higher order means more accurate prediction and smaller FPE value. But, if a high order AR model is used for prediction, the computation will be time consuming. For the real-time haptic rendering which requires a high update rate, we need to consider both the prediction accuracy and computation time.

We calculate FPE values for AR models with different orders based on the standard haptic data set. Fig. 6.1 shows the FPE results of the AR model with different orders. We can observe that the FPE decreases quickly when the order increases from 1 to order 5 and does not change much after order 5. So, we choose the order equal to 5 to be used in our AR model to predict force values during haptic rendering.

Figure 6.1: The FPE results of the AR model with different orders.

6.1.1.2 Interpolation Algorithm

After we predicted new haptic force values from the AR model, the latest calculated force values and the predicted new force value are used together for the force interpolation.

Based on the existing physical simulation forces $\mathbf{F}_t, \dots, \mathbf{F}_{t-i}$ and the predicted force $\mathbf{F}_{t+1|t}$ in the next physical time step, the interpolation algorithm calculates the haptic forces during the time interval $t \leq t < t + 1$. Meanwhile, these interpolated forces keep curvature continuity with the previous haptic force and are applied to the haptic device at a high update rate (1 kHz).

For the interpolation algorithm, first, the interpolated force values should be at least C1 curvature continuity. Second, the predicted force values are used as the control points in the interpolation force calculation. Because the predicted force values have errors comparing with the real force values, the interpolated force values may not include the control points to reduce the error. In our algorithm, we propose to use the B-Spline function to calculation the interpolated force values. It is a spline curve parameterized by spline functions which has C2 continuity. The B-spline function is widely used in computer-aided design and 3D modeling.

The cubic interpolation equation can be written as follows:

$$\mathbf{P}(u) = \mathbf{a}u^3 + \mathbf{b}u^2 + \mathbf{c}u + \mathbf{d}, \quad (6.4)$$

where $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ are parameters. u changes from 0.0 to 1.0 to control the position on the interpolated curve $\mathbf{P}(u)$. According to the B-spline conditions, the parameters are calculated as

$$\mathbf{P}(u) = \frac{1}{6} \begin{vmatrix} u^3 & u^2 & u & 1 \end{vmatrix} \begin{vmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{vmatrix} \begin{vmatrix} \mathbf{P}_i \\ \mathbf{P}_{i+1} \\ \mathbf{P}_{i+2} \\ \mathbf{P}_{i+3} \end{vmatrix}, \quad (6.5)$$

$$\mathbf{P}(u) = \mathbf{U}^T \mathbf{M}_s \mathbf{P}, \quad (6.6)$$

where $\mathbf{P}_i, \dots, \mathbf{P}_{i+3}$ are control points, $\mathbf{M}_s$ is the parameter matrix, $\mathbf{U}^T$ is the transpose of u vector.

Based on the known force/torque values and the predicted force/torque value, the force and torque interpolation equations are

$$\mathbf{F}^h(u) = \mathbf{U}^T \mathbf{M}_s \mathbf{F}^p, \quad (6.7)$$

$$\mathbf{T}^h(u) = \mathbf{U}^T \mathbf{M}_s \mathbf{T}^p. \quad (6.8)$$

We use n to represent the rate of the haptic rendering frequency and the frequency of calculation of prediction. So, for each prediction period, there will be n interpolated force values applied to the haptic device. The value of u is calculated as

$$u = \frac{i}{n}, \quad 0 \leq i < n. \quad (6.9)$$

In Fig 6.2, the relation between the predicted forces and interpolated forces is shown. The prediction algorithm predicts the next force values $\mathbf{F}_{t-2}^p, \dots, \mathbf{F}_{t+1|t}^p$ in the prediction thread represented with circles as shown in Fig. 6.2. The yellow circles represent previously predicted force values and the red circle represents the next force value predicted at the current time t . The interpolation algorithm calculates the interpolated force values (represented with rectangles) in the haptic thread. The $\mathbf{F}_u^h$ represents the interpolated force value applied to the haptic device at the current haptic frame during the

time interval of the physical simulation. During each time interval of the prediction thread, there is a certain number of force values interpolated in the haptic thread.

Figure 6.2: The force prediction and interpolation for haptic display. The prediction thread is represented with circles and the haptic thread represented with rectangles.

6.1.2 Adaptive Virtual Coupling

In haptic rendering, the proposed adaptive virtual coupling algorithm [57] described in Chapter 5 is used to calculate the interaction force in physical simulation. The haptic interaction force $\mathbf{F}_{haptic}$ is calculated from the following equation:

$$\mathbf{F}_{haptic} = -k_t(\mathbf{P}_{HIP} - \mathbf{P}_{tool}) + b_t(\mathbf{V}_{HIP} - \mathbf{V}_{tool}), \quad (6.10)$$

where $\mathbf{P}_{HIP}$ is the position of the haptic handle, $\mathbf{V}_{HIP}$ is velocity of haptic handle, $\mathbf{P}_{tool}$ is position of virtual tool and $\mathbf{V}_{tool}$ is velocity of virtual tool. We do a series of experiments to get function $f(m)$, which is used to calculate the appropriate parameter of the stiffness k_t of the virtual coupling for different mass values m . The parameter of damping b_t is calculated from the function $g(k_t, m)$ as follows:

$$k_t = f(m), \quad (6.11)$$

$$b_t = g(k_t, m). \quad (6.12)$$

In the calculation of rotation torque, we use the similar process to calculate the haptic torque, rotational stiffness k_{rot} , and rotational damping b_{rot} . The haptic torque $\mathbf{T}_{haptic}$ is calculated from the following equation:

$$\mathbf{T}_{haptic} = -k_{rot}(\mathbf{U}_{HIP} - \mathbf{U}_{tool}) + b_{rot}(\mathbf{W}_{HIP} - \mathbf{W}_{tool}), \quad (6.13)$$

where $\mathbf{U}_{HIP}$ is the orientation of the haptic device and $\mathbf{U}_{tool}$ is the orientation of the virtual tool. $\mathbf{W}_{HIP}$ and $\mathbf{W}_{tool}$ represent the angular velocity of the haptic device and the virtual tool. The rotational stiffness and rotational damping parameters of the virtual coupling are calculated from the inertia moment of the virtual tool I . We use function $d(I)$ to calculate the rotational stiffness k_{rot} and function $h(k_{rot}, I)$ to calculate the rotational damping coefficient b_{rot} as follows:

$$k_{rot} = d(I), \quad (6.14)$$

$$b_{rot} = h(k_{rot}, I). \quad (6.15)$$

6.1.3 Architecture of Haptic Rendering

The pipeline of the haptic rendering system is shown in Fig. 6.3. The system consists of physical thread, haptic thread, and prediction thread. The physical thread mainly calculates the collision detection between the virtual tool and other virtual objects, contact force and dynamic movement. The dynamic calculation results and the position information of the haptic device are used as inputs to the virtual coupling algorithm. Depending on the different contact positions, the computation for collision detection and the dynamic simulation could be time consuming when the number of contact points increases. So, the update rate of the physical thread has variable frequency instead of a constant frequency during the haptic rendering process. If force values are calculated in low rate in physical thread and are directly passes to haptic device, the user may feel obvious lag and discontinuous force during the haptic manipulation.

Figure 6.3: The pipeline of the haptic rendering system using prediction algorithm. The prediction algorithm is used in the prediction thread (200Hz) to predict the next force value. The interpolation algorithm is used in the haptic thread (1 KHz) to interpolate smooth haptic force values.

So, in haptic thread, we implement the virtual coupling algorithm and interpolation algorithm to implement smooth haptic rendering by providing constant high frequency force values updates. The adaptive virtual coupling algorithm can adjust the stiffness and damping parameters for different mass values of virtual tools. The interpolation algorithm will interpolate smooth force values from the predicted force and virtual coupling force.

To guarantee that the prediction algorithm has enough time to complete a calculation of force prediction, we set the update rate of the prediction thread to 200Hz. In our system, the update rate of the haptic thread is 1 KHz. Thus, there are five interpolated force values in the haptic thread during each period of the prediction thread. The interpolation algorithm interpolates continuous force values using the B-spline algorithm. The interpolated forces are directly sent to the haptic device in high update rate of 1 kHz.

6.2 Experiment Results

6.2.1 Implementation and Benchmarks

In this section, the proposed prediction method using interpolation is implemented in a real-time haptic rendering system. Two benchmarks are implemented for testing and analysis. For each benchmark, we use the same haptic manipulation path to test different haptic rendering algorithms.

The experiments were performed on a Windows PC with Intel Core2 Quad Q9400 2.66 GHz CPU. A PHANToM Premium 1.5/6DOF haptic device from SensAble Technologies is used in the experiment to provide 6-DOF force and torque feedback. Fig. 6.4 shows the peg-in-hole benchmark and Stanford bunny benchmark which is implemented in the real-time haptic rendering system.

Figure 6.4: Three benchmark models used for evaluation of the proposed haptic rendering method. (a) peg model, (b) box with hole model, and (c) bunny model.

In our system, virtual objects are implemented using CHAI library that provides both graphic rendering and haptic interface. Different haptic devices can be used in a virtual environment through the haptic interface. CHAI library can be integrated with Open Dynamic Engine (ODE) where collision detection between multiple moving objects can be efficiently computed [85]. However, the CHAI library only provides a 3-DOF haptic rendering algorithm for the point-based haptic rendering. Therefore, we integrated our 6-DOF stable adaptive algorithm based on virtual coupling in the CHAI library to provide both force and torque haptic feedback.

6.2.2 Error Measurement

Figure 6.5: Key configurations of the haptic manipulation using the peg-in-hole benchmark.

In our experiment, we compare the proposed prediction algorithm with the linear prediction algorithm [71]. The linear prediction algorithm is commonly used for force extrapolation based on the calculated force values. In the linear prediction algorithm, $\mathbf{F}_t, \mathbf{F}_{t-1}$ are calculated from the physical simulation loop to predict the force values during the next haptic frames. The predicted force $\mathbf{F}_h$ is calculated as follows:

$$\mathbf{F}_h = \frac{T - T_t}{T_t - T_{t-1}}(\mathbf{F}_t - \mathbf{F}_{t-1}) + \mathbf{F}_t, \quad T_t \leq T < T_{t+1}, \quad (6.16)$$

where $[T_t, T_{t+1}]$ is the time interval between the two physical frames. T is the current time

used to linearly interpolate the force values.

The peg-in-hole benchmark is implemented in this experiment to evaluate the prediction accuracy during the haptic manipulation. The key configurations of the haptic manipulation are shown in Fig. 6.5. In the experiment, the haptic force values from the configuration 1 to 3 are used for comparison.

The peg-in-hole benchmark is implemented in this experiment to evaluate the prediction accuracy during the haptic manipulation. Fig. 6.6 shows the experiment results for different prediction algorithms. The comparison of the accuracy (between the predicted force values and true physical force values) is presented as the absolute errors shown in Fig. 6.6 (b) and Fig. 6.6 (d). In Table 6.1, it is shown the error mean of the force magnitude. Comparing to the error of the linear prediction algorithm (0.031 N), the error of the proposed spline interpolation based prediction algorithm (0.0289 N) is reduced by 6.77%. In addition, the error variance is reduced. The proposed spline interpolation based prediction algorithm is more stable than the linear prediction algorithm.

From the results of the experiments, we can see that the large force feedback errors appear when the force begins to fall off after the peak of the force magnitude. These large errors appeared in the linear prediction algorithm also causes instability in the haptic device. For example, in the linear prediction algorithm, sharp changes on the force magnitude are detected at 800ms in Fig. 6.6 (a). In the proposed prediction algorithm, the errors also increases when the force changes sharply, but the interpolated final force applied to the haptic device is kept stable and continuous, shown in Fig. 6.6 (d).

Table 6.1: The force prediction accuracy of different algorithms.

Algorithms	Error Mean (N)	Error Variance
Linear prediction	0.031	0.0013
Spline interpolation based prediction	0.0289	0.00086

Figure 6.6: Evaluation of the accuracy of the prediction algorithms. (a), (b) The predicted force and absolute errors in the linear prediction algorithm. (c), (d) The predicted force and absolute errors in the spline interpolation based prediction algorithm.

6.2.3 Smooth haptic force analysis

The purpose of the spline interpolation based prediction algorithm is to achieve smooth haptic rendering in dynamic virtual environment when complex objects are manipulated. The Stanford bunny benchmark is used in the haptic rendering for testing. Because of the high complexity of the bunny model and multiple point contacts in the objects collision detection, the update rate of the physical simulation loop drops to less than 100 Hz while the haptic rendering is needed to be kept at 1000 Hz update rate. In Fig. 6.7, we compare the performance of the algorithm during the multiple contacts for complex models. We chose the force data during the same interaction when the forces change from increase to decrease. The green asterisk points represent the measured physical simulation force and the blue circle points represent predicted forces.

Fig. 6.7(a) shows obvious vibration in the force result in the linear prediction algorithm. If two successive haptic force values have large difference, the linearly predicted force would change immediately without the continuity with the previous forces. This sudden change can cause a force kicking or vibration in the following haptic frames.

The performance of proposed spline interpolation based prediction algorithm is shown in the Fig. 6.7(b). When the force values have large difference, the interpolation algorithm can slowly change the current force value to the next force value through a smooth curve. Comparing with the linear force prediction, both the accuracy of the prediction and the continuity of the haptic force values are improved. Even when the haptic force has a sharp change, the interpolation algorithm can calculate continuous force values in each haptic frame.

For the linear prediction algorithm, it can provide stable and accurate force prediction only when the force is changing linearly. The prediction algorithm that uses only the auto-regressive model can provide more accurate prediction, but it cannot guarantee continuous force values [76]. So, the proposed spline interpolation based prediction algorithm overcomes these problems to generate an accurate and smooth haptic force feedback during the whole haptic rendering process.

(a)

(b)

Figure 6.7: Evaluating the smoothness of haptic rendering. (a) The performance of the linear prediction algorithm with vibrations. (b) The result of the spline interpolation based prediction algorithm with smooth force change.

6.3 Real-time AR Coefficient Update

One disadvantage of the proposed prediction algorithm (in Section 6.1) is that the coefficients of the auto-regressive model need to be preset for the particular haptic rendering process. According to different virtual tools or different physical properties, we need to choose different coefficients for haptic rendering. Otherwise, the accuracy of the haptic force prediction could be reduced.

To overcome this drawback and improve the accuracy of the force prediction, we propose to update the coefficients of the auto-regressive model during the haptic rendering. First, we use default AR coefficients for the haptic prediction. When the haptic system has recorded certain number of force data, the coefficients calculation algorithm (shown in Fig. 6.8) calculates new coefficients based on the latest force data in real-time. In this way, the user doesn't need to evaluate the coefficients before the haptic rendering, and the accuracy of the prediction algorithm is suitable for different haptic rendering processes.

Figure 6.8: The haptic rendering pipeline with real-time auto-regressive coefficient calculation.

To solve the coefficients of the auto-regressive model, the equation 6.1 need to be multiplied by $\mathbf{F}_{t-d}$. Then, the matrix of Yule-Walker equations can be calculated through the expectation and normalizing [87]

$$\begin{bmatrix} 1 & r_1 & r_2 & \cdots & r_{p-2} & r_{p-1} \\ r_1 & 1 & r_1 & \cdots & r_{p-3} & r_{p-2} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ r_{p-2} & r_{p-3} & r_{p-4} & \cdots & 1 & r_1 \\ r_{p-1} & r_{p-2} & r_{p-3} & \cdots & r_1 & 1 \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{p-1} \\ \phi_p \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_{p-1} \\ r_p \end{bmatrix}, \quad (6.17)$$

where r_i is the autocorrelation coefficient at delay i . It also can be written as:

$$\mathbf{R}\Phi = \mathbf{r}, \quad (6.18)$$

$$\Phi = \mathbf{R}^{-1}\mathbf{r}, \quad (6.19)$$

where $\mathbf{R}$ and $\mathbf{r}$ are matrix and vector of r_i , Φ is the vector of the auto-regressive coefficients which can be solved with least-squares method.

The time complexity of the prediction algorithm with real-time coefficient calculation using the least-squares methods is $O(n^3)$. During the coefficient calculation, the size of data window will greatly affect the computation time. The experiment is implemented with the duck benchmark. Table 6.2 and Fig. 6.9 show the relation between the data window size and the prediction accuracy. The accuracy of the prediction algorithm with coefficients update is related to the size of the data window.

Table 6.2: Comparison of accuracy with different window sizes.

Window Size	RMS Force Error (N)
100	0.0563
200	0.0388
300	0.0306
400	0.0306
500	0.0302

Figure 6.9: The comparison of the RMS Force Error with different window size.

From Fig. 6.9 and Table 6.2, we can see that the change of errors is minor after 300 window size. Therefore, in our work, we choose to use 300 as the data window to recalculate the coefficients of auto-regressive model in real-time.

Table 6.3: Accuracy comparison of algorithms.

Algorithms	RMS Force Error without Coefficients Update (N)	RMS Force Error with Coefficients Update (N)
AR Prediction Algorithm Without Spline Interpolation	0.0445	0.039
AR Prediction Algorithm with Spline Interpolation	0.0345	0.0306

Table 6.3 shows the comparison of the RMS force errors with/without real-time coefficient update for AR prediction algorithm with/without spline interpolation. The real force value is read from the haptic force data set of the duck benchmark with the data window size 300. For the auto-regressive prediction algorithm, the error is 0.039 N which

is less 12.36% comparing with the error in the algorithm using constant coefficients. For auto-regressive prediction algorithm with spline interpolation, the error is 0.0306 N which is less 11.3% comparing with using constant coefficients.

In this chapter, we developed a prediction algorithm using the spline interpolation to provide the smooth haptic interaction force in a high update rate. The auto-regressive model is used to predict the force value from the previous haptic force results. A spline function was introduced to interpolate smooth force values for the haptic force output. In addition, a real-time AR coefficient calculation algorithm is used to update the AR model during the haptic rendering. Therefore, there is no need to calculate the coefficients before the haptic rendering and the accuracy of the prediction can be improved. The proposed algorithm is compared with other algorithms with standard benchmarks. It can overcome the force discontinuity caused by the heavy calculation of physical simulation or multiple point contacts with complex models.

6.4 Discussion

In this chapter, a prediction method with real-time coefficients updating for haptic rendering is proposed. The purpose of the proposed prediction method is to make sure a smooth and accurate haptic rendering is implemented in complex virtual environments.

An autoregressive model is used to predict force/torque from the previous haptic force/torque calculations. In addition, a real-time coefficient calculation algorithm is proposed to update the AR model during the haptic rendering. To generate smooth haptic feedback in successive haptic frames, the spline interpolation function is used to calculate the haptic forces/torques applied to haptic device. The number of the interpolated force/torque values is calculated from the ratio of the update rate of the haptic thread and the update rate of the physical thread.

Finally, the proposed algorithm is compared with other algorithms using the standard benchmarks. The experiment results show that it can overcome the force discontinuity caused by the heavy calculation of physical simulation and multiple point contacts with complex models.

Chapter 7

Haptic Based 6-DOF Molecular Docking

A haptic device allows the user to manipulate the molecules and feel interactions during the docking process in virtual environment. Implementation of force-torque feedback allows the user to have more realistic experience during molecular docking simulation and find the optimum docking positions faster. In this chapter, first, we introduce research background of molecular docking in Section 7.1 and review visual and haptic based molecular docking systems in Section 7.2. In Section 7.3, we introduce basic concepts and forces of molecular interaction. The stable dynamic algorithm is integrated in biomolecular docking system to improve the stability of force and torque feedback. The implementation and performance of the molecular docking system are discussed in Section 7.4. In Section 7.5, we develop a multi-user molecular docking system for collaboration using different haptic devices. Collaborative docking is implemented with two different haptic devices.

7.1 Molecular Docking Background

Virtual environment can integrate visual, audio and haptic tools for both research and e-learning applications. Biomolecular docking is a new research area which includes development of software systems with both visual and haptic interfaces for rational drug

design. In works [42, 88], a visual haptic-based biomolecular docking system for helix-helix docking research and application of this system in e-learning was proposed and implemented. In our work, we propose an improved stable haptic rendering algorithm for biomolecular docking. The user can experience 6-DOF haptic force-torque feedback during the process of molecular docking. Moreover, we develop an application for collaborative molecular docking with different haptic devices.

The molecular docking process of drug design can be simulated in a 3D space where a ligand can be docked on to a receptor. By using computer-aided design systems, the manipulation of molecules can be realized with real-time interactive visualization in virtual environment. Especially, it has been proved to be very helpful for users to understand the interactions between molecules in e-learning applications. As realized in [89], Cooper developed a multiplayer online molecular docking game Foldit for non-scientists to engage in hard protein prediction problems solving.

Beside the basic visual technology for molecules, the haptic interface appears to be another effective tool to improve the immersion and interaction during the molecular docking process in virtual environment. Haptic technology provides interactivity between the real and virtual environment through the force and torque feedbacks transmitted by the haptic devices. This makes it possible to manipulate molecules and transform biomolecular interactions into the sensory experience during the virtual experiment. Therefore, haptic-based visual biomolecular docking allows developing more interactive systems that could be used in rational drug design and molecular medicine.

Biomolecular docking is an assembling process for molecular structures to predict the preferred complimentary molecular shapes that can bind molecules to form a stable complex. Since there is an exponential increase in conformations as the number of atoms increases, the simulation of docking task with an automatic conformation search algorithms could be difficult. By using visual haptic-based molecular docking system, the user can manually explore the conformational molecular space to find an optimal conformation within the minimum time.

Some previous studies have proved that the force display can provide a better understanding of the molecular docking process compared with traditional visual display methods [90]. Because haptic device provides realistic force feedbacks to the users, in

recent years, more and more researchers tend to explore and analyze the molecular docking process with haptic interface [91-93].

In earlier work [94], Ouh-young proposed a real-time system for interactive molecular docking that allows the user to manipulate the position of ligand and feel the interactive forces between molecules. A force smoothing method has been presented in [95] where forces from the Lennard-Jones (LJ) force field were calculated and the instability was eliminated when two atoms are in contact.

Besides traditional 3-DOF haptic force feedback, the torque force also plays an important role in molecular docking process. Persson and Cooper [96] developed a force-torque haptic molecular interaction system. One evaluation was also conducted to test the importance of the force feedback in learning and understanding the interactions between molecules. Their results proved that both force and torque forces play important roles in helping students to understand the concepts of molecular interaction. In [97], a haptic device and computational engine were developed for computer-aided molecular docking (CAMD) which provided both force and torque feedback.

7.2 Related Works of Molecular Docking

Modern molecular visualization systems such as RasMol[98], PyMol[99], Jmol[100], MDVQS [101], etc. allow visualizing and analyzing complex molecular structures. On the other hand, haptic-based technology allows molecular docking with force feedback in such a way that the user could “feel” force field of bimolecular interactions. There are haptic-based systems that also allow the users to feel an electrostatic force of the explored molecule.

Lai-Yuen and Lee [102] developed a computer-aided design system with the lab-built 5DOF haptic device for molecular docking and nano-scale assembly. Nagata and Mizushima[103] developed a prototype for protein-ligand docking simulation with the total potential energy (Van der Waals potential energy, electrostatic potential energy and hydrogen bond potential energy) calculation between atoms of ligand and protein. In [104], a grid map was used to generate the electrostatic field data around the molecular structure. The haptic forces at any position were calculated using tri-linear interpolation

of the potential energy. Stocks and Hayward developed a haptic system HaptiMol ISAS [39]. It allows the user to interact with the biomolecular solvent accessible surface through the haptic device. A navigation cube is used to visualize the explored surface region, and the cube can be automatically scaled to fit the workspace of the haptic device. The cube approach allows choosing a limited interaction area of very large molecules. In another approach for the rigid body molecular docking proposed by Subasi and Basdogan [40], the user can insert a rigid ligand molecule into the cavities of protein to search for binding cavity. Similarly to the cube approach, an Active Haptic Workspace (AHW) was implemented for the efficient haptic-based exploration of large protein-protein docking in high resolution. In the system, the user could feel a tunneling effect when the ligand molecule is pulled towards the binding cavity. In [92], an Interactive Global Docking (IGD) approach was presented. An immersive environment for docking interface with both visual and haptic rendering was implemented. The commercial Falcon haptic device from Novint Technologies was used to provide three degree-of-freedom force feedback. In [93], an Interactive Molecular Dynamics (IMD) system was implemented. A real-time force feedback based virtual reality system Steered Molecular Dynamics (SMD) [105] was developed for dynamic simulations of bimolecular interaction.

For the six degree-of-freedom haptic-based molecular docking, Daunay and Micaelli [106] developed a 6-DOF haptic-based molecular docking system which enables the feeling of both force and torque. The system provides haptic feedback for a flexible ligand-protein docking. However, the system has limitations on the size of protein molecules. The work of CoRSAIRe [91] is an example of a multisensory virtual reality system with 6-DOF haptic device provided by the Virtuous haptic interface of Haption Company. It was designed for a study of protein-protein docking. To create an immersive virtual environment, the visual, audio and haptic feedbacks were combined to enhance the process of exploration.

Haptic-enabled web-based molecular docking is a new research direction in development of molecular docking simulation. In [107], Davies implemented a prototype of Molecular Visualiser (MV) system with Web3D standards adding an extension to support haptic interaction. MV provides the following features: visualizations of molecular systems, visualization of potential energy surfaces, and implementation of

wave packet dynamics. The system can run in a web browser using VRML, or be delivered to a virtual environment in which haptic properties are assigned based on the molecular dynamics of the system. The authors mainly focused on the visualization of the molecular models and did not study molecular docking problem. A new multiplayer online game Foldit has been developed by Cooper [89]. It is a novel approach for solving the protein prediction problems. The Foldit has proved that the accurate protein structure model can be produced through game play. As predicted by the Foldit system, human search process has advantages in complexity, variation and creativity. Liu and Sourin [108] proposed a functional approach for modeling of objects geometry. The function-based objects were added into VRML. The user is able to define function of any shape with implicit functions. Later, Wei et al [109-111] extended the system proposed in [108] by incorporating haptic based features to new FVRML nodes. A new density node was proposed for haptic implementation. It allows exploring an electrostatic force of molecule with a probe in VRML.

7.3 Basic Concepts and Algorithm Description

We developed the prototype of biomolecular docking system HMolDock using the haptic device PHANTOM 1.5/6DOF. A Protein Data Bank [112] format file of molecular structure is used as an input of the molecular model. Atom coordinate is got from the input PDB files, the radius, position and the correspondent color are also determined based on the atom type and its belonging residue which is extracted from PDB data. An interaction force including torque force between ligand and receptor are calculated and displayed in real time, so that the user could feel an attractive/repulsive force and rotation torque through the 6-DOF haptic device simultaneously.

7.3.1 Lennard-Jones Potential

The simulation is based on the Lennard-Jones Potential which is usually used to describe the interaction between a pair of neutral atoms or molecules.

The common Lennard-Jones Potential is expressed as

$$V(r) = 4\varepsilon \left[\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^6 \right], \quad (7.1)$$

where r is the distance between the atom pair. ε is the depth of potential well, σ represents the specific distance at where the inter-particle potential is zero. The values of these two parameters will be different according to the interacting particles.

The Lennard-Jones potential is an efficient mathematical model used to approximate the interaction force according to the distance between two atoms or molecules. As shown in Fig. 7.1, the potential is expressed as mildly attractive when one molecule is approaching to another from a far distance. However, when two molecules are close enough, the potential will be strongly repulsive.

Figure 7.1: The simulation of Lennard-Jones potential.

The aim of bimolecular docking is to achieve an optimized conformation and orientation between two molecules such that the potential overall combination energy is minimized. Through the calculation of Lennard-Jones potential and 6-DOF haptic device, the user can find the optimal docking positions efficiently.

7.3.2 Lennard-Jones Force

In the transmembrane α -helix interaction [113], the Lennard-Jones potential is considered as the most important factor between molecules. The interaction forces between molecules are well approximated by Lennard-Jones potential which changes according to the distance between molecules. We use the Lennard-Jones force which is a simple type of Van der Waals force to simulate the interaction between ligand and receptor.

Assume that there are M atoms in receptor and N atoms in ligand, the Lennard-Jones potential between these two molecules is

$$V = \sum_{i=1}^M \sum_{j=1}^N 4\varepsilon_{ij} \left[\left(\sigma_{ij} / r_{ij} \right)^{12} - \left(\sigma_{ij} / r_{ij} \right)^6 \right], \quad (7.2)$$

where ε_{ij} and σ_{ij} are LJ parameters for atom i in receptor and atom j in ligand, r_{ij} is the distance between the atom pair. Furthermore, the Lennard-Jones force function is derived as

$$\begin{aligned} F &= \nabla V(r) = d(V(r)\tilde{r}) / dr \\ &= \sum_{i=1}^M \sum_{j=1}^N 24\varepsilon_{ij} [2(\sigma_{ij}^{12} / r_{ij}^{13}) - (\sigma_{ij}^6 / r_{ij}^7)]\tilde{r}, \end{aligned} \quad (7.3)$$

where $\tilde{r}$ is the distance unit vector. The Lennard-Jones force is one simple type of Van der Waals force.

The algorithm described in (7.3) is sensitive to the distance r_{ij} . Depending on the distance between two molecules, the interaction force can be classified into two opposite force sections short-range repulsion force and long-range attraction force.

In the equation (7.3), the term $(1 / r_{ij})^{13}$ describes the short-range repulsive force. When the distance between a pair of atoms or molecules is very close, the denominator distance r is relatively small. Therefore, the magnitude of repulsion force is strongly large. The term $(1 / r_{ij})^7$ describes the long-range attractive force. As the separation

between a pair of atoms or molecules increases, this term plays the dominate role. Correspondingly, the interaction between two particles will change form intensively repulsive to mildly attractive. Finally, the interaction force tends to the zero as distance tends to infinity. In the practice of molecular docking process, these two opposite force factors play the dominate role alternatively according to the distance of atom or molecule pairs.

Algorithm 1 Force Calculation

```

1: Input: Postition of ligand  $P_l$ , Position of receptor  $P_r$ 
2: Output: Interactive forece  $F_{device}$ 
3: {The ligand is grasped by the haptic device}
4: if grasp object is true then
5:   {Initialize variables}
6:    $F \leftarrow 0$  {Interactive force}
7:    $D_{ij} \leftarrow 0$  {Distance between one atom pair}
8:   {Retrieve all atoms of ligand}
9:   for all  $i$  such that  $1 \leq i \leq M$  do
10:    {Retrieve all atoms of receptor}
11:    for all  $j$  such that  $1 \leq j \leq N$  do
12:       $S \leftarrow \sigma_{ij}$ 
13:       $E \leftarrow \varepsilon_{ij}$ 
14:       $D_{ij} \leftarrow P_{rj} - P_{li}$ 
15:       $r \leftarrow |D_{ij}|$  {Get the magnitude of distance vector}
16:       $r_{dir} \leftarrow \text{normalize}(D_{ij})$  {Get the direction of distance vector}
17:       $F \leftarrow f(r)$  {Calculate the interactive force between this atom pair}
18:       $F_{device} \leftarrow \text{Scale}(F)$  {scale the force magnitude to fit the haptic device}
19:    end for
20:  end for
21: end if
22: return  $F_{device}$ 

```

There are many different force field models that can be used to simulate proteins and other molecules. The model used and described in here is OPLS-aa [114-116] which is parameterized for small organic molecules in protein simulation.

For the homo-atomic pairs, there are published LJ parameters available (i.e. [117] for OPLS-aa). The interaction of hetero-atomic pairs, the effective values of σ and ε are calculated from those for the homo-atomic pairs. This way of calculation is called mixing rule. OPLS-aa uses the same non-bonded functional forms as AMBER [118], and the Lennard-Jones terms between unlike atoms are computed using the mixing rule [119] as

$$\sigma_{ij} = \sqrt{\sigma_{ii}\sigma_{jj}}, \quad (7.4)$$

$$\varepsilon_{ij} = \sqrt{\varepsilon_{ii}\varepsilon_{jj}}. \quad (7.5)$$

7.3.3 Haptic Torque

Most of the haptic rendering algorithms for molecular docking are based on three degree-of-freedom haptic devices which can only simulate force effects by moving along three coordinates. However, with 6-DOF haptic devices, the torque feedback of ligand can be calculated to produce the torque effects which could add rotation force around 3 axes.

For some previous torque calculations, the torque is only calculated around the position of haptic device's attachment point which is set to be the closest atom to the ligand's centre of geometry. In our algorithm, as shown in Alg. 2, we chose a more flexible way when the torque force can be calculated at any surface position of the ligand. As long as the HIP is attached to the ligand and the docking process started, the position of HIP is stored and updated to calculate the torque feedback. In this way, the user has more freedom in manipulation of the ligand. Especially in e-learning, the users could have a better understanding of interactions between molecules with changing position of HIP.

The calculation of the torque force requires atom forces derived from (7.3) and the position of the haptic interface point x_{HIP} on the ligand as well. Like the computation of force, the torque feedback is the sum of all torque effects from atom pairs between ligand and receptor. The haptic torque T is calculated as follows:

$$T = \sum_{j=1}^N [(x_j - x_{HIP}) \times \sum_{i=1}^M F_{ij}], \quad (7.6)$$

where x_j is the position of ligand atom j , x_{HIP} is the position of contact point between

haptic interface point and ligand. For the specific ligand atom j , $\sum_{i=0}^M F_{ij}$ represents the

interaction force between one ligand atom and all atoms of the receptor. The torque is

calculated by the cross product between a force vector and a displacement vector (vector from the point that torque is measured to the point where Lennard-Jones force is applied).

Algorithm 2 Torque Calculation

```

1: Input: Postition of ligand  $P_l$ , Position of receptor  $P_r$ , Grasp position  $P_{HIP}$ 
2: Output: Torque  $T_{device}$ 
3: if grasp object is true then
4:   {Initialize variables}
5:    $T \leftarrow 0$  {Torque vector}
6:    $T_d \leftarrow 0$  {Distance vector from ligand center to grasp position}
7:   {Retrieve all atoms of ligand}
8:   for all  $i$  such that  $1 \leq i \leq M, i \in ligand$  do
9:     {Retrieve all atoms of receptor}
10:    for all  $j$  such that  $1 \leq j \leq N, j \in receptor$  do
11:       $F_{ij} \leftarrow f(r)$  {Calculate the force on this ligand atom}
12:       $T_d \leftarrow P_i - P_{HIP}$  {Calculate direction vector for torque}
13:       $T_{ij} \leftarrow T_d \times F_{ij}$  {Calculate the torque using cross product}
14:    end for
15:     $T \leftarrow \sum_{i=1}^M F_i$  {Accumulate the torque value on one ligand atom}
16:  end for
17:   $T_{device} \leftarrow Scale(F)$  {Scale the torque magnitude to fit the haptic device}
18: end if
19: return  $T_{device}$ 

```

7.3.4 Stable Haptic Rendering

The force and torque are calculated from the sum interaction force of all possible atom pairs between two molecules. It is well known that this force is extremely sensitive to the distance between two molecules. When two molecules are close enough, the Lennard-Jones force changes from attraction to repulsion (a sudden change in both the direction and magnitude). This change can cause a force kicking effect in successive haptic frames. To improve stability of the haptic feedback, we use the proposed stable dynamic haptic rendering algorithm based on virtual coupling (described in Section 4.1) to calculate the interaction force/torque between molecules. Comparing with the direct rendering method which directly applies the interactive force and torque to the haptic device, the proposed algorithm based on the virtual coupling can improve stability and filter kicking caused by the sudden change of Lennard-Jones force. Furthermore, the

haptic force/torque can automatically saturate to the maximum force/torque values of the haptic device.

The purpose of bimolecular docking is to find the position with the minimal potential energy. However, in practice, this position is hard to be captured manually since there is a great change in the force magnitude from attraction force to repulsion force. As a result, the user with a haptic device is hard to stay in this minimal energy position in practice. To solve the problem of the potential unstable factor and to produce an accurate manipulation, we set a zero area when the magnitude of the molecular docking force is small. The zero area is shown as the red area in Fig. 7.2. The horizontal axis r is the distance between the particle pair and the vertical axis F is the interaction force. Therefore, the user can have a stable control of ligand and maintain it in the position of the minimal energy area. Otherwise, there would be an obvious vibration around this area.

Figure 7.2: The zero area (in red) is set when the potential energy approximates to zero and the force changes from attraction to repulsion.

7.4 System Implementation and Performance

In this section, we describe the HMolDock system and the results of molecular docking interaction. To display the force and torque feedback, the PHANROM Premium 1.5/6DOF designed by SensAble Technologies is used in our system.

7.4.1 Implementation

The proposed haptic force-torque rendering algorithm has been implemented in our prototype HMolDock. The system is developed and tested on a dual 1.86GHz CPU workstation using OpenHaptics® Toolkit, OpenGL® library and Visual C++ programming language. Fig. 7.3 shows the set-up of the HMolDock system with 6-DOF force-torque feedback haptic device in the laboratory.

Figure 7.3: HMolDock system with PHANToM 6-DOF haptic device.

Although there are different file formats for molecular structures, we use a Protein Data Bank (PDB) format for the input. Two molecules could be visualized on the screen as shown in Fig. 7.3. The user could assign a haptic interface point to one of the molecules and move this molecule towards/around of another one to feel both force and torque feedback around three axes.

An overall structure of a haptic rendering pipeline for bimolecular docking process is

shown in Fig. 7.4. The haptic rendering pipeline of the HMolDock system is mainly consists of force/torque calculation, stable rendering algorithm, and dynamic simulation of molecular models.

At first, when the user manipulates the ligand around the receptor, the system calculates the Lennard-Jones force depending on the distances between all atom pairs of two molecules. In addition, the interactive torque is calculated using the Lennard-Jones force and the position of the contact point x_{HIP} . Second, the interactive force and torque are sent to the stable rendering algorithm where the haptic force/ torque for haptic device are calculated, and the zero area testing will check the force values for stable manipulation at the docking position. Finally, the dynamic simulation engine uses interactive force/torque and haptic force/torque to calculate a new position of the molecule in the next haptic rendering frame.

Figure 7.4 The haptic rendering pipeline of the HMolDock system with both force and torque feedback.

7.4.2 System Performance

Fig. 7.5 shows a bimolecular docking process between one α IIb helix (with 154 atoms) and one designed antibody-like complementary peptide anti- α IIb (with 266 atoms). The coordinate grid is used to help the user to improve the accuracy of manipulation in 3D environment. After molecules are loaded into the system, the user can assign a haptic interface point to probe and grab the ligand which can be moved towards/around the receptor.

The position of contact point between haptic probe and ligand is used to calculate the torque force caused by all atoms pairs between ligand and receptor. The resulting attraction/repulsion forces and rotation torques forces can be displayed through the 6-DOF haptic device. Therefore, the molecule can be selected by the haptic mouse and moved around to let the user ‘feel’ the force-torque feedback in our bimolecular system. Values of force and torque magnitudes are shown on the right side of the screen to help the user to find the optimum docking position. In addition, the force direction and magnitude are visualized as a yellow vector, and the cyan vector indicates the change of torque vector. Here, the direction and length of the arrow indicates the attraction/repulsion force and its magnitude. Fig. 7.5(a) shows an attraction force between two separated molecules. As the distance becomes smaller, the repulsion force will appear and increase rapidly. Fig. 7.5(b) shows the force and torque directions and magnitudes when two molecules contact with each other. Both directions of force and torque change to the opposite, and the increased magnitudes can be read from the value shown on the right side of the screen.

With the intuitive vector representation and force-torque feedback, the user can experience more realistic force feeling during the docking process and analyze the optimal positions of minimal potential energy.

(a) Separate

(b) Contact

Figure 7.5: An interaction between two α -helices: the yellow and cyan arrows represent the force and torque vectors. In (a), two molecules are separated in a distance, and the force vector indicates the attractive force. In (b), two molecules contact with each other and the force vector changed to the inverse direction as a repulsive force.

7.4.3 Performance Analysis

Fig. 7.6 shows both force response and torque response in one molecular docking process. During this process, the haptic device manipulates the ligand approach to the receptor from a far apart distance to a contact status. We divide this molecular docking process into the following three time intervals:

- $T_0 < T < T_1$ (Separate): two molecules are far apart, the force and torque magnitudes are very weak.
- $T_1 < T < T_2$ (Approach): two molecules are separated with a limited distance.
- $T_2 < T < T_3$ (Contact): the ligand contacts with receptor, the repulsion force increases greatly.

Figure 7.6: Force and torque magnitude in a molecular docking process.

As shown in Fig.7.6, both intermolecular force magnitude and torque magnitude change according to the distance between two molecules with the same trend. In addition, the force magnitude is greater and more sensitive than the torque magnitude in the process of molecular docking. Through our biomolecular docking system HMolDock, the force and torque change from attraction to repulsion can be directly experienced by the user while performing drug design or molecular docking simulation. Therefore, the optimum docking position can be found by the intuitive haptic feeling instead of expensive computation of docking algorithms.

To solve the instable and vibration phenomenon appeared during the molecular docking process, we have applied a linear smoothing method to improve the stability of docking manipulation. Associate with the zero area that we have set when the interaction force change from attraction to repulsion the user can experience an accurate manipulation. Since the changes of force and torque feedbacks have the similar trends in docking process and force is much more sensitive to distance between molecules, we use the force feedback to analysis how stable methods improve the stability of molecular docking manipulation.

Fig. 7.7 shows the simulation results of a complete molecular docking process. The stable methods were used in the docking process shown in Fig. 7.7(a). Correspondingly, the docking processing without stable methods is shown in Fig. 7.7(b). The same ligand and receptor were used in these two molecular docking experiments. The docking manipulations in the experiments were also implemented with the same docking path and docking position within the certain time range.

When the ligand approaches to the receptor from a far distance, the interaction force should be a mildly attractive force. In the stable method, this attractive force gradually falls from the 10th to the 15th second as shown in Fig. 7.7(a). In contrast, the approaching process without stable methods contains vibrations during the same process. Due to such vibrations, the original attractive force can even become repulsion force as can be seen at the 14th second shown in Fig. 7.7(b).

As the molecules get closer, the interaction force will have a sudden change from attraction to repulsion force. In the stable simulation, although this change is fast after the attraction force reaches its maximum value, the changing process is relatively smooth as can be seen at the 15th in Fig. 7.7(a). However, in the simulation without the stable method, the maximum attraction cannot be felt distinctly because of the continuous vibrations in the previous frames. Moreover, these vibrations will not disappear during the force changing process. Therefore, for the user with a haptic device, a sudden force kicking phenomenon will happen at the 17th second as it is seen in Fig. 7.7(b).

When the distance between visual representations of two molecules is close enough, and two molecules contact with each other, the interaction force will strongly increase. Although there is some tiny vibration appeared in the stable simulation, the user can feel

the continually increase of the interaction force and where is the maximum repulsion force. In contrast, the simulation without the stable method has strong vibrations around the 20th second with the great vibration amplitude as shown in Fig. 7.7(b). In practice, the vibration could be so strong that the user felt hard to control the haptic device during the manipulation. So, the maximum repulsion force cannot be clearly felt by the user.

Figure 7.7: The Simulation results of a molecular docking process using same ligand and receptor. (a) The optimized interaction force feedback during a molecular docking manipulation with stable method. (b) The interaction force feedback without stable method during a docking process.

7.5 Collaborative Molecular Docking

In addition to implementing both force and torque feedback in molecular docking system, we implemented a collaborative virtual environment for multi-user molecular docking

manipulation with multiple haptic devices. In this collaborative molecular docking system, we use the CHAI library for computer haptics and real-time simulation. The CHAI library supports most commercially available desktop haptic devices [120].

Haptic rendering is performed on a quad processor to update position and orientation information and to calculate force and torque values. Haptic feedback is provided by two different kinds of haptic devices. As shown in Fig. 7.8, the left 3-DOF device is a Novint Falcon haptic interface made by Novint Technologies, and Inc. A PHANToM Premium 1.5/ 6-DOF haptic device made by SensAble Technologies, Inc is on the right side of the screen. In this virtual environment, the receptor is controlled by the Phantom haptic device and the ligand is controlled by the Falcon haptic device. During the molecular docking process, both users can feel the interaction force between ligand and receptor.

Figure 7.8: Collaborative molecular docking with two different haptic devices. Falcon haptic device is shown on the left, and PHANToM Premium 1.5/6-DOF haptic device is shown on the right of the screen. The ligand and receptor can be controlled in the same collaborative virtual environment.

It has been proved that the multi-user games can be used to produce an accurate protein structure model [89]. Compared with previous molecular docking methods, multi-user human search has much better performance in creativity and complexity. In this collaborative system we combine the virtual environment and various haptic devices to improve the problem-solving ability for exploring molecular structures.

Since haptic devices from different companies were used in our system, the stability

problem became more complex than the system with single device. In order to maintain stability of different devices, our program can recognize the type of device automatically and set corresponding parameters to optimize force/torque output.

7.6 Discussion

In this chapter, the application of the haptic rendering technology in molecular docking with both force and torque feedback is proposed. It enables the user to experience 6-DOF haptic manipulation of the molecular systems in a virtual environment. To provide an additional flexibility of the torque display in the molecular docking process, the user can change the attachment position where to apply device to move a ligand. In addition, the stabilization method is implemented to calculate smooth force and torque feedback.

We also developed a multi-user molecular docking application. The proposed system supports collaborative molecular docking for multiple users with different types of haptic devices. Compared with the previous molecular docking methods, collaborative multi-user search allows users come up with more creative solutions to docking of complex molecular structures.

For the future work, we are planning to use this collaborative biomolecular docking system in e-learning and implement the haptic rendering for more complex molecular models. Moreover, the advanced algorithms can be integrated to the system to help user to find the appropriate docking positions quickly.

Chapter 8

Applications of Haptic Rendering Algorithms

In this chapter, the architecture of the haptic rendering system and applications implemented with the proposed 3-DOF/6-DOF haptic rendering algorithms are described. In Section 8.1, our haptic rendering system based on the CHAI 3D library is presented. In Section 8.2, we introduce a “T Puzzle” haptic based game. In the game "T Puzzle", the virtual blocks are assigned with small mass values, and the stable dynamic 6-DOF algorithm is implemented to provide stable haptic manipulation in the virtual environment. In Section 8.3, an EEG-enabled haptic-based serious game for post stroke rehabilitation is presented. The adaptive haptic rendering algorithm is implemented to guarantee accurate haptic manipulation of the virtual objects with different mass values.

8.1 Haptic Rendering System

Our haptic rendering system is developed based on the CHAI Library [121] which is an open source haptic rendering API. It provides haptic interfaces for various haptic devices. The graphic rendering, collision detection, and physical simulation engine are integrated for effective development of haptic applications. However, the CHAI library only provides the 3-DOF haptic rendering algorithm for the point-based haptic rendering. Thus,

in our work, we proposed and implemented new 6-DOF object-based haptic rendering algorithms. The proposed haptic rendering algorithms can also be used for 3-DOF haptic rendering without torque calculation when a virtual tool has mass property.

Figure 8.1: The CHAI library integrated with 3-DOF/6-DOF haptic rendering and prediction algorithm.

As shown in Fig. 8.1, the haptic rendering pipeline is composed with the following four threads:

- **Graphic thread:** It reads the primitives of the 3D models and renders the graphic effects of the virtual environment, such as texture, light, transparency, and reflection. During the haptic rendering the graphic thread updates the scene in a low update rate about 60Hz.
- **Physics thread:** Based on the position and orientation of the virtual tool and the virtual object, the collision detection algorithm checks the contacts between models and calculates contact points. The contact information is used for physical simulation which can calculate the movement of the virtual tool after contact. the Open Dynamic Engine [85] is used in CHAI library.

- **Haptic thread:** The input of the haptic thread is the haptic device's configuration read from sensors and the virtual tool's configuration calculated by the physical simulation engine. The virtual coupling algorithm calculates the interactive forces and torques for the virtual tool with different mass values. The saturation algorithm can guarantee the magnitude of haptic force/torque smaller than the limitations of haptic devices. If the prediction algorithm is used, the interpolation algorithm applies the spline function to interpolate smooth force/torque values for haptic device. The haptic thread works in a high constant update rate (1000 Hz) to make sure stable and transparent haptic feedback.
- **Prediction thread:** It reads and records force/torque calculated from the haptic thread. The prediction algorithm can predict the next force/torque value in the next haptic frame. When the auto-regressive model is used for prediction, the coefficients calculation algorithm can update the coefficients of model in real-time to improve the accuracy of haptic force/torque prediction.

Comparing with the original CHAI library, our haptic rendering system has the following features. At first, it supports both 3-DOF and 6-DOF haptic rendering. The user can use the haptic device to manipulate a virtual tool instead of a point in the virtual environment. Secondly, the stability is improved with the proposed haptic rendering algorithms based on virtual coupling. Thirdly, the virtual tool can be assigned with mass property from light to heavy. Finally, the proposed prediction algorithm improves the haptic performance while rendering complex and/or deformable models.

Haptic-based interaction could add a new dimension to “serious games” development. The user could “feel” objects surfaces, complex objects interaction in a 3D virtual environment. In the following sections, we propose two haptic-based serious games for medical applications.

8.2 Haptic-Enabled Puzzle Game

8.2.1 T Puzzle Game

The original board game that was taken as a starting point for this work is named “T

puzzle” or “Four Piece Tangram”. The idea of this game is now involved into various new versions all over the world. The earliest known version of this puzzle appeared in 1903, when it was used to advertise White Rose Ceylon Tea in the USA. Fig. 8.2(a) shows four wooden pieces of the puzzle. The “Four Pieced Tangram” likes the Seven Pieced puzzle in China, which is with 7 boards. It was thought to originate from China in ancient time, and has been popular in many cultures since. Among the various figures that the “Four Pieced Tangram” can form, one representative figure is like the capital letter “T”, so people usually call this game the “T puzzle”. Fig. 8.2(b) shows “T” shape formed by all the pieces and other 3 shapes from junior, middle to senior levels. In Fig.8.2(c), it shows our user interface of game with four boards with different colors.

Figure 8.2: “T Puzzle” Game: (a) Four wooden pieces of the puzzle. (b) Different levels: From junior, middle to senior level of game. (c) User interface of the haptic-based game.

CHAI 3D library is used to develop this game. We use the proposed stable dynamic haptic rendering algorithm for haptic manipulation. Because the models are simple in this game, we can put the collision detection and physical simulation in the haptic thread and maintain a high update rate. The pipeline used for haptic rendering is shown in Fig. 8.3.

Figure 8.3: Haptic rendering pipeline of the “T Puzzle” game.

For haptic rendering, first we need to get the position of the interaction tool, and then, to calculate the collision detection among different objects in the virtual environment. From the contact between the interaction tool and other objects, we get the response force and torque after computation using the stable dynamic haptic rendering algorithm. The program ends both force and torque feedback to the haptic device.

8.2.2 Force and Torque Calculation

In “T puzzle”, the user can hold or release each polyhedron, and place polyhedrons in different positions in the virtual environment. Four polyhedrons can be differentiated from their shapes and colors. For collision detection and dynamic movement, the ODE engine is used to calculate the movement of the dynamic virtual objects.

Since this game is haptic-based, force feedback is necessary to be programmed. For the original CHAI 3D haptic rendering, the feedback force is modelled as a spring, the object grasped keeps vibrating and flicking, which means this force feedback model is not stable, and it cannot make the user feel like grasping an object realistically. To solve this problem and improve the stability and flexibility of manipulation, the 6-DOF stable dynamic algorithm based on virtual coupling (proposed in Chapter 4) is used, which connects the user’s haptic motions with the motions of the dynamic object through a

virtual coupling link.

8.2.3 Implementation and Performance

As shown in Fig. 8.4, we used the CHAI library and ODE game engine for this haptic game design. The game is implemented on a Windows XP Professional PC with Intel® Core™2 Quad Q9400 2.66 GHz CPU and 3.25GB memory. For the haptic device, we use PHANToM Premium 1.5/6DOF from SensAble Technologies which provides 6-DOF force and torque feedback.

Figure 8.4: The user interface of “T puzzle” game. The PHANToM 6-DOF haptic device is used to manipulate virtual objects in game.

One of the main objectives of this work is to propose, design and implement a game to be played in a 3D virtual environment using the feedback from haptic device. In the proposed “T puzzle” game, the player could feel the force feedback and collisions among boards or between a board and the ground as in a real world. When the user starts the game, the user interface (shown in Fig. 8.5) comes out. The user can use the middle button of the mouse to adjust the viewpoint, which would be the distance between objects and the display window on the computer screen. The user can have a more flexible view of the playing process with this action, which makes the game easier and more comfortable to play.

Figure 8.5: Screenshot of haptic-based “T puzzle” game. It is developed based on OpenGL and Glut.

When the game starts, there is not gravity in the virtual environment. If “1” is pressed, the gravity will be set and objects would drop on the ground. Then players can use these boards to form the figures they want on the ground. If “2” is pressed, objects would be weightless and float around in the virtual environment. To play this game using haptic device, players can use buttons to control the functions to grasp and drop objects. Users can also rotate the boards in the virtual environment and feel the torque feedback when collision happens.

There are six buttons on the right column of the display window. They are used to show three shapes and solutions of “T puzzle” game. As introduced at the beginning of chapter 8, there are many different difficulty levels for the “T puzzle”, and more tasks can be added into this game environment. The game can be used for the post stroke patient’s rehabilitation.

8.3 Emotion-enabled Haptic-based Serious Game for Post Stroke Rehabilitation

In this section, we propose and develop a novel emotion-enabled haptic-based serious game for post stroke rehabilitation. The 3-DOF adaptive haptic rendering algorithm is implemented to guarantee accurate haptic manipulation of the virtual objects with different mass values (see in Section 5.2). Real-time patient's emotion monitoring based on the Electroencephalogram (EEG) is used as an additional game control. A subject-dependent algorithm recognizing negative and positive emotions from EEG was implemented. The emotion recognition algorithm consists of two parts: features extraction and classification. Force feedback is implemented in the game. The proposed haptic-based serious game could help to promote rehabilitation of motor deficits after stroke. Such games could be used by the patient for post stroke rehabilitation even at home convenience without a nurse presence.

8.3.1 Stroke Rehabilitation Game Background

In recent years, easily available new technologies such as Electroencephalogram (EEG) reading devices and haptic devices brought new opportunities in serious game design and development. EEG is a non-invasive technique recording the electrical potential over the scalp which is produced by the activities of brain cortex, and reflects the state of the brain [122]. EEG technique gives us an easy and portable way to monitor brain activities such as the user's emotions [123] and level of concentration [124] by using suitable signal processing and classification methods and algorithms. Haptic devices give an opportunity to the user "to feel" the simulated virtual environment in the way similar to the real world. For example, the user could feel the weight of objects in virtual worlds or the forces of the objects interaction. Thus, the player could have a possibility to act in the game in a similar way as in the real world. Using both technologies such as EEG reading devices and haptic devices allow propose new approaches in serious game design and development.

We propose an adaptive haptic-based post stroke rehabilitation game with emotions monitoring. “Rehabilitation helps stroke survivors relearn skills that are lost when part of the brain is damaged. For some stroke survivors, rehabilitation will be an on-going process to maintain and refine skills and could involve working with specialists for months or years after the stroke” [125]. That is why, it is important to propose new treatments that need less assistance from therapists. Current treatments for stroke rehabilitation employ multidiscipline approaches such as chemical (drugs), physical (therapeutic exercise, acupuncture, etc.), psychological approach (relaxation with music, etc.) or a combination of the above-mentioned approaches. Physical exercises need engagement of the nurse who could access the physical and mental state of the patient during the exercise and/or assist the patient during the rehabilitation session. There are three types of motion exercises: passive range of motion, active assistive range of motion and active range of motion [126]. In the first case, the patient could move his/her body parts by him/herself if possible (for example, with the healthy hand) or only with the help of the nurse. In the second type of the exercises, the nurse assists in the exercises, for example, to lift the arms if the patient could move the arm only partially. At active range of motion stage, the patient could do the movements without assistance but still the nurse presence could be needed to monitor the patient mental state and his/her recovery progress. Such exercises should help to promote joint flexibility, strengthening, increased muscular endurance, coordination and even improve the patient’s emotional state.

Since it is often difficult to maintain patient’s motivation with the traditional stroke rehabilitation exercises, there is a need for research on novel technologies and games design. These technologies may be effective in optimizing the patient engagement during the rehabilitation process. There are several games for upper limb post-stroke rehabilitation which use novel technology such as webcam images, Nintendo Wii technology, etc. The main advantages of such games are that it would automatically store the patient profile, for example, the patient reaching ability, play time, scores and so on. Although a great improvement is reported when haptic devices were used in the games, fundamental research is still needed to validate clinical experiments. In this work, we propose to use novel interaction technologies for better user’s engagement in the game: haptic interface and EEG-based interface. The user’s emotions recognized from EEG are

used as an additional game control to adapt the game level. Emotion recognition algorithms with adequate accuracy are needed to be implemented to make the game fully adaptive to the patient mental state. We developed a multi-level computer game following the traditional exercise that includes putting different objects into the basket. The objectives of the game are to aid the patient's concentration, upper limb strength at the same time monitoring the user's emotional state and adapting the game level according to the patient current emotion and score.

8.3.2 Methods and Materials

The proposed and implemented game combines haptic-enabled 3-dimensional physically-based virtual environment and real-time emotion monitoring based on the users EEG. The following two devices are used: an EEG device and a haptic device. The overall diagram of the game design is shown in Fig. 8.6. After the game starts, in the EEG pipeline, raw EEG signals read from the user are filtered and analyzed by an emotion recognition algorithm, and the resulting parameter such as negative or positive emotion is passed to the game control module. Meanwhile, the player uses his/her hand to control the haptic device in the physically-based virtual environment. The haptic rendering module reads the position, orientation and control command from the haptic interface, then the haptic interaction force is calculated and is conveyed to the user through the haptic device. The game score is calculated and recorded during the haptic manipulation.

The game control module checks the users score and emotional statistics on the completion of the current level of the game and automatically adapts the difficulty level of the next round of the game. Thus, the decision about the next difficulty level is made based on the players score and the emotion dominated during the current game level. This is especially important for rehabilitation games as negative emotions could eliminate the therapeutic effect of the game or even harm the patient's health. If the negative emotion dominates, the game goes to the easy level if the score is low or stays at the same level if the score is high or even terminated in the case of exceeding the time limit for negative emotions. The nurse could be called if the game is played in the hospital environment.

Figure 8.6: The overall diagram of the EEG-enabled haptic-based serious game for post stroke rehabilitation.

A therapeutic effect of the game is achieved by combination of the physical exercise done by the patient using haptic device with force feedback and the users emotion monitoring. The target of the game is not only to do prescribed physical exercises but at the same time stay positive during the exercise. To implement the user's emotions monitoring, a real-time emotion recognition algorithms is needed to be integrated. Real-time EEG-based emotion recognition algorithms development and its integration in the game design is a new challenging multidisciplinary research topic. In work [127], an emotion recognition algorithm was proposed. The algorithm can recognize up to 8 emotions such as happy, surprised, satisfied, protected, angry, fear, unconcerned, and sad in Valence Arousal-Dominance emotion model. The algorithm consists of two parts: features extraction and classification. The combination of features such as statistical and fractal dimension features that gave the best emotion classification accuracy was chosen for the algorithm implementation in the game. The algorithm was tested by using two

experimental EEG databases: one with audio stimuli and another with visual stimuli and by using the DEAP benchmark database with video stimuli [128]. The best accuracy of 100% was obtained for two levels of valence recognition (negative and positive) with controlled dominance level using 14 channels [127].

The patient plays the game with the force feedback that is felt through the haptic device. CHAI library is chosen for the game implementation. The library is able to integrate Open Dynamics Engine which is used to model physical properties of 3D objects, for example, to assign weights to the objects in the developed game. The dynamic engine provides the user with a full immersion experience of the physical virtual environment that could improve the therapeutic effect of the game.

8.3.3 Game Implementation and Result

In order to play the developed emotion-enabled haptic based game, the user needs EEG reading device, haptic device, and PC computer. EEG data is read by Emotiv device [129] with 14 electrodes locating at AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC6, F4, F8, AF4 standardized by the American Electroencephalographic Society [130]. The technical parameters of the device are given as follows: bandwidth 0.2 - 45Hz, digital notch filters at 50Hz and 60Hz; A/D converter with 16 bits resolution and sampling rate of 128Hz. NOVINT Falcon [31] device was used for haptic feedback in the game. EEG signal is transmitted to the computer with Bluetooth. The data is filtered with bandpass filter 2 - 42 Hz. After that, statistical features and fractal dimension values are calculated with algorithms implemented with the sliding window of 512 samples size. Then, based on the calculated emotion recognized and the score gained by the user the command to the game is given. The game was developed with the realistic graphics and friendly user interface.

Table 8.1: Decision for changing the difficulty levels of the “Basket” game.

Score	Dominant emotion	Difficulty level of the next round
High	Negative	Same level
High	Positive	Higher level
Low	Negative	Lower level
Low	Positive	Same level

The scoring system was implemented to keep the track of the patient’s performance and time spent for each exercise. Currently, three levels of the game were implemented. The patient has to pick up objects and put them into the basket using the haptic device handle. For each successful object being picked and put into the basket, the score points are rewarded. The objects have different appearance (colour and shape) to entertain the user. Different weight and size is assigned to the 3D objects at different levels of difficulty to make exercises more challenging at different levels. For example, the basket could become wider or narrower in different difficulty levels depending on the users performance and emotions. The moving between the levels depends on the users time spent for the completing the task and emotional state. Conditions for changing the difficulty levels are shown in Table 8.1. At each level, high or low score can be achieved. The score for each level is calculated based on the time spent by the player and the number of objects collected in the basket. If the player puts all objects into the basket in the predefined time, the high score is assigned to the player. If the player exceeds the time the low score is assigned. There is also the overall time limit for playing the game and overall time limit if negative emotion is dominant through the game. In such case, the game is stopped. If negative or positive emotion is recognized and occupies more than 50% of the playing time at the current level, this emotion is considered as the dominant emotion. If the user completes the current game level in the appropriate time but the emotional state is negative one then the game is continued at the same difficulty level but

(a)

(b)

Figure 8.7: EEG-based and haptic-based stroke rehabilitation 3D game. (a) User with EEG and haptic devices. (b) The hard level of game with a small basket and four objects with different mass values.

with more fancy shapes and colours and at the same time the music is played to improve the players mood. If emotions are positive during the game then the game level changes to higher or lower difficulty level directly according to the players score. The design of the game follows the rules and regulations in maintaining the standard procedure of progressive recuperation.

In Fig. 8.7(a), an example of the game setting with EEG device and haptic device is shown. The Emotiv device is mounted on the patient's head for his/her real-time emotion monitoring. The user plays the game using NOVINT Falcon haptic device. The player has to put different objects into the basket. In Fig. 8.7(b), the screenshot of the highest difficulty level 3 is shown when objects are heavier and the basket is smaller.

Currently, the "Basket" game is implemented to promote only active range of motions. The final objective is to implement series of "Basket" games corresponding to passive and active range of motions as well.

8.3.4 Discussion of Stroke Rehabilitation Game

The EEG-enabled haptic-based serious game for stroke rehabilitation could be played by the patient not only in the presence of the nurse but at home convenience as well. The proposed game is adapted to the current emotional state of the patient that is very important for post stroke therapy efficiency improvement. Currently, the game was tested by 3 students (23-25 years old) and by one post stroke patient (69 years old) to get the preliminary feedback on the game. The post stroke patient gave the following positive comments comparing to the traditional "Basket" non-computer game he used for his rehabilitation: a better choice of objects in the computer game (better appearance in colors and size), automatic tracking the play time, automatic score keeping, and finally, a positive feedback to the user with rewards points. More research should be done in future on the assessment of the therapeutic effect of the game with more subjects, and statistics should be collected. Currently, the implemented game is targeting active range of motion. As the haptic technology is involved in the game, it could be possible to implement passive range of motion and active assistive range of motion that usually are used at early

stages of post stroke rehabilitation. The proposed game could be used as a part of the home-based rehabilitation programs.

Thus, we proposed a cost-effective EEG-based and haptic-based technology for post stroke rehabilitation. The cost of the developed software is included in the current project cost and can be minimized just to the cost of CD copy. The final cost of the treatment could include one EEG headset, one haptic device with access to PC computer that currently has a tendency to reduce. Then, the patient could have any recommended amount of sessions prescribed by the doctor free of charge. As the game is adaptive to the user's abilities, the length of the sessions and the level of difficulty could be changed according to the monitored patient's emotional state and the score gained. Stroke is the leading cause of serious, long-term disability in the United States [131]. Each year, more than 700,000 people suffer a stroke in US. Stroke not only causes physical discomfort, but also interferes with social relationships, family life and self-esteem. Considering all above the proposed post stroke rehabilitation tools could improve quality of life giving the patients a good alternative to more expensive traditional treatment.

8.4 Discussion

In this chapter, various applications implemented with the proposed haptic rendering algorithms are presented. At first, the architecture of our 3-DOF/6-DOF haptic rendering system used for haptic-based applications is described. The CHAI library is introduced which is an open source haptic API and is compatible with various haptic devices. We extended CHAI 3D library with 6-DOF haptic rendering algorithms and implemented the proposed 3-DOF/6-DOF algorithms to improve the stability and accuracy of haptic rendering.

Second, a "T Puzzle" haptic-based serious game is developed for intellectual human development. This game is implemented with the 6-DOF haptic device that provides both force and torque feedback to make the user feel the mass of virtual objects and the contacts between virtual objects.

Third, we developed an emotion-enabled haptic-based serious game for post stroke rehabilitation exercises. The EEG-based real-time emotion monitoring can recognize the

patient's emotion and control the game's difficulty level to improve the efficiency of the rehabilitation exercises. In future research, we will test the proposed game with more subjects and assess the therapeutic effect of the proposed game.

Chapter 9

Conclusion and Future Work

9.1 Conclusion

In this research, we focus on the stable haptic rendering methods and algorithms to generate continuous force and torque feedback in dynamic virtual environments. In haptic rendering, many algorithms have been proposed for 3-DOF or 6-DOF haptic rendering, such as the god-object [34], virtual proxy [14], spring-damper model [48], QSA [49], constraint-based algorithm, etc. Although these methods have greatly improved performance of haptic rendering, there are still unsolved problems such as buzzing (it appears as continuous high frequency vibrations when the virtual tool has small mass property), inaccurate manipulation (large displacement exists between the virtual tool and the virtual object when the virtual tool has a large mass property that introduces inaccurate movement during the haptic manipulation), and discontinuous force update (when there are complex models and/or deformable models and the physical simulation may produce a low update rate of interaction force which causes discontinuous force output on haptic device). The aim is to solve existing problems and improve the stability of haptic rendering by proposing general stable haptic rendering algorithms.

At first, we proposed a stable dynamic algorithm based on virtual coupling for 6-DOF haptic rendering. It can overcome the “buzzing” problem when a virtual tool with small mass values is used. The novelty is that a nonlinear force/torque algorithm is proposed to calculate the haptic interaction when the collision happens between the

virtual tool and virtual objects. The force/torque magnitude is automatically saturated to the maximum force/torque value of the haptic device. The algorithm is tested on standard peg-in-hole benchmark with 180/176 polygons and on the Stanford Bunny benchmark with 20,898 polygons. The stability of the proposed stable dynamic algorithm outperforms spring-damper algorithm and QSA algorithm.

Second, we proposed an adaptive haptic rendering algorithm based on virtual coupling to overcome the inaccurate manipulation problem caused by the large mass values of the virtual tool. The algorithm can automatically adjust virtual coupling parameters according to the mass values of the simulated virtual tools. In addition, the force/torque magnitude is saturated to the maximum force/torque values of the haptic device when large interaction force is generated. The algorithm was tested on the standard haptic rendering benchmarks. We tested the algorithms with different mass values from 0.05kg to 0.2kg. The average displacement of the adaptive algorithm is reduced by 52.43% comparing with the spring-damper algorithm, and is reduced by 42.51% comparing with the QSA algorithm.

Third, we propose a new prediction algorithm for smooth haptic rendering to overcome the low update rate of the force during physical simulation of complex and/or deformable models. We propose to use a prediction method combined with an interpolation method to calculate smooth haptic interaction force. The auto-regressive model with time complexity of $O(n)$ is used to predict the force value from the previous haptic force calculation. We introduce a spline function to interpolate smooth force values for the haptic force output. The proposed method can provide smooth and continuous haptic force feedback in a high update rate during virtual manipulation of complex and/or deformable objects. It outperforms other force estimation/prediction methods. The RMS force error of the proposed algorithm is 0.0289 N which is smaller than the linear prediction algorithm's RMS force error 0.031 N. In addition, a run-time coefficient calculation algorithm with time complexity of $O(n^3)$ is used to update the coefficients of auto-regressive model during the haptic rendering. It can further improve accuracy of the force prediction. For common auto-regressive prediction algorithm using the real-time coefficients update, the error is reduced by 12.36% comparing with the algorithm using the constant pre-set coefficients. For the auto-regressive prediction algorithm with spline

interpolation using the real-time coefficient update, the error is reduced by 11.3% comparing with the algorithm using constant pre-set coefficients.

Finally, the proposed algorithms are implemented in various applications. A haptic enabled molecular docking system HMolDock is developed to find the correct docking positions between ligand and receptor. Here, a stable haptic rendering algorithm is implemented at the application level of the system to enable stable haptic manipulation of large molecules. HMolDock can help the drug designer to find the correct docking positions between molecular systems. For medical applications, we developed a haptic-based serious game "T Puzzle" and EEG-enabled haptic-based serious game "Basket". In the game "T Puzzle", the virtual blocks are assigned with small mass values, and the stable dynamic algorithm is implemented to provide stable haptic manipulation in the virtual environment. This game can be used for intellectual development and post stroke rehabilitation exercises. The EEG-enabled haptic based post stroke rehabilitation serious game "Basket" is developed to help patients to perform rehabilitation activities. In the game, the haptic device is used to manipulate various virtual objects and move them into the basket. The adaptive haptic rendering algorithm is implemented to guarantee an accurate haptic manipulation of the virtual objects with different mass values. The EEG based emotion recognition algorithm is implemented to recognize the emotions of the patient and automatically adjust the difficulty level of the game. The proposed haptic rendering algorithms are also integrated in CHAI 3D library.

9.2 Future Work

Nowadays, the speed of the network is faster and faster and collaborative haptic simulation becomes a new direction for haptics research. It allows more than two users to use several haptic devices to manipulate objects in the same virtual co-space. Collaborative haptic simulation has a great potential in applications, for example, such as collaborative surgery that allows the doctors do surgery operation in the remote location and in collaboration with other surgeons. Providing stability in haptic rendering becomes even more important task in such applications. Some feasibility study of collaborative

3-DOF haptic rendering was already done and proved the concept. However, the time requirements of 6-DOF haptic rendering for network and data transformation are extremely high because of time consuming collision detection, force/torque calculation, and required high haptic update rate. To investigate all these challenges, new research should be done, and fast and efficient algorithms are needed to be proposed for collaborative 6-DOF haptic rendering. The proposed predictive haptic rendering algorithm could be adapted and applied in such collaborative environment to solve the problems caused by delays in the network.

References

- [1] A. El Saddik, "The potential of haptics technologies," *IEEE Instrumentation and Measurement Magazine*, vol. 10, pp. 10-17, 2007.
- [2] M. A. Otaduy and M. C. Lin, "High fidelity haptic rendering " *Synthesis Lectures on Computer Graphics and Animation*, vol. 2, pp. 1-110, 2006.
- [3] C. Basdogan, C. H. Ho, and M. A. Srinivasan, "Virtual environments for medical training: Graphical and haptic simulation of laparoscopic common bile duct exploration," *IEEE/ASME Transactions on Mechatronics*, vol. 6, pp. 269-286, 2001.
- [4] A. Petersik, B. Pflesser, U. Tiede, K. H. Hühne, and R. Leuwer, "Realistic Haptic Interaction in Volume Sculpting for Surgery Simulation," *Lecture Notes in Computer Science*, vol. 2673, pp. 194-202, 2003.
- [5] S. Andrews, J. Mora, J. Lang, and W. S. Lee, "Hapticast: A physically-based game with haptic feedback," *Proceeding of FuturePlay 2006*, pp. 1-8, 2006.
- [6] L. T. De Paolis, M. Pulimeno, and G. Aloisio, "An interactive and immersive 3D game simulation provided with force feedback," in *International Conference on Advances in Computer-Human Interaction*, Sainte Luce, 2008, pp. 26-30.
- [7] B. Baxter, V. Scheib, M. C. Lin, and D. Manocha, "DAB: Interactive haptic painting with 3D virtual brushes," in *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, 2001, pp. 461-468.
- [8] N. Zonta, I. J. Grimstead, N. J. Avis, and A. Brancale, "Accessible haptic technology for drug design applications," *Journal of Molecular Modeling*, vol. 15, pp. 193-196, 2009.
- [9] F. G. Hamza-Lup and I. A. Stanescu, "The haptic paradigm in education: Challenges and case studies," *Internet and Higher Education*, vol. 13, pp. 78-81, 2010.
- [10] K. Salisbury, D. Brock, T. Massie, N. Swarup, and C. Zilles, "Haptic rendering: programming touch interaction with virtual objects," presented at *the Proceedings of the 1995 symposium on Interactive 3D graphics*, Monterey, California, United States, 1995.
- [11] D. Morris, N. Joshi, and K. Salisbury, "Haptic battle pong: High-degree-of-freedom haptics in a multiplayer gaming environment," *Proceedings of Experimental Gameplay Workshop at Game Developers Conference (GDC)'04*, 2004.
- [12] M. A. Otaduy, "6-dof haptic rendering using contact levels of detail and haptic textures," Ph. D. dissertation, University of North Carolina at Chapel Hill Chapel

- Hill 2004.
- [13] C. B. Zilles and J. K. Salisbury, "Constraint-based god-object method for haptic display," in *International Conference on Intelligent Robots and Systems*, 1995, pp. 146-151.
 - [14] D. C. Ruspini, K. Kolarov, and O. Khatib, "Haptic display of complex graphical environments," in *Proceedings of the 1997 Conference on Computer Graphics, SIGGRAPH*, 1997, pp. 345-352.
 - [15] A. Gregory, A. Mascarenhas, S. Ehmann, M. Lin, and D. Manocha, "Six degree-of-freedom haptic display of polygonal models," in *Proceedings of Visualization*, 2000, pp. 139-145+549.
 - [16] Y. J. Kim, M. C. Lin, and D. Manocha, "DEEP: Dual-space expansion for estimating penetration depth between convex polytopes," in *IEEE International Conference on Robotics and Automation*, 2002, pp. 921-926.
 - [17] Y. J. Kim, M. A. Otaduy, M. C. Lin, and D. Manocha, "Six-degree-of-freedom haptic rendering using incremental and localized computations," *Presence: Teleoperators and Virtual Environments*, vol. 12, pp. 277-295, 2003.
 - [18] D. D. Nelson, D. E. Johnson, and E. Cohen, "Haptic rendering of surface-to-surface sculpted model interaction," *American Society of Mechanical Engineers, Dynamic Systems and Control Division (Publication) DSC*, vol. 67, pp. 101-108, 1999.
 - [19] D. E. Johnson, P. Willemsen, and E. Cohen, "Six degree-of-freedom haptic rendering using spatialized normal cone search," *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, pp. 661-670, 2005.
 - [20] W. A. McNeely, K. D. Puterbaugh, and J. J. Troy, "Six degree-of-freedom haptic rendering using voxel sampling," *Proc. of ACM SIGGRAPH*, pp. 401-408, 1999.
 - [21] M. Wan and W. A. McNeely, "Quasi-Static Approximation for 6 Degrees-of-Freedom Haptic Rendering," in *Proceeding of the IEEE Visualization Conferece*, 2003, pp. 257-262.
 - [22] J. Barbič and D. L. James, "Six-DoF haptic rendering of contact between geometrically complex reduced deformable models," *IEEE Transactions on Haptics*, vol. 1, pp. 39-52, 2008.
 - [23] J. Barbic and D. L. James, "Six-dof haptic rendering of contact between geometrically complex reduced deformable models: Haptic demo," in *Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2009, pp. 393-394.
 - [24] M. A. Otaduy, N. Jain, A. Sud, and M. C. Lin, "Haptic rendering of interaction between textured models," in *Proceedings of the conference on Visualization '04*, 2004, pp. 297-304.
 - [25] M. A. Otaduy and M. C. Lin, "Stable and responsive six-degree-of-freedom haptic manipulation using implicit integration," in *Proceedings of the World Haptics*

- Conference, Washington, DC, USA, 2005, pp. 247-256.
- [26] M. A. Otaduy and M. C. Lin, "A modular haptic rendering algorithm for stable and transparent 6-DOF manipulation," *IEEE Transactions on Robotics*, vol. 22, pp. 751-762, 2006.
 - [27] Z. Wang, D. Wang, Y. zhang, and M. C. Lin, "Analysis on increasing transparency for penalty-based six degree-of-freedom haptic rendering," in *World Haptics Conference (WHC), 2011 IEEE*, 2011, pp. 83-88.
 - [28] K. Salisbury, F. Conti, and F. Barbagli, "Haptic rendering: introductory concepts," *Computer Graphics and Applications, IEEE*, vol. 24, pp. 24-32, 2004.
 - [29] V. Hayward, O. R. Astley, M. Cruz-Hernandez, D. Grant, and G. Robles-De-La-Torre, "Haptic interfaces and devices," *Sensor Review*, vol. 24, pp. 16-29, 2004.
 - [30] D.-I. T. A. Kern, "Engineering Haptic Devices - A Beginner's Guide for Engineers," *Springer Science & Business Media*, 2009.
 - [31] The Novint Falcon haptic device. Available: <http://www.novint.com/index.php/novintfalcon>
 - [32] A. Petersik, B. Pflesser, U. Tiede, Karl-Heinz, and R. Leuwer, "Realistic haptic interaction in volume sculpting for surgery simulation," presented at *the Proceedings of the 2003 international conference on Surgery simulation and soft tissue modeling*, Juan-Les-Pins, France, 2003.
 - [33] M. A. S. Cagatay Basdogan, "Haptic Rendering in Virtual Environments," *California Institute of Technology*, 2003.
 - [34] C. B. Zilles and J. K. Salisbury, "Constraint-based god-object method for haptic display," in *Proceedings of the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Pittsburgh, PA, USA, 1995, pp. 146-151.
 - [35] A. Liu, F. Tendick, K. Cleary, and C. Kaufmann, "A Survey of Surgical Simulation: Applications, Technology, and Education," *Presence: Teleoperators and Virtual Environments*, vol. 12, pp. 599-614, 2003.
 - [36] I. Shakra, M. Orozco, A. El Saddik, S. Shirmohammadi, and E. Lemaire, "VR-based hand rehabilitation using a haptic-based framework," in *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, 2006, pp. 1178-1181.
 - [37] A. Barghout, A. Alamri, M. Eid, and A. El Saddik, "Haptic rehabilitation exercises performance evaluation using automated inference systems," *International Journal of Advanced Media and Communication*, vol. 3, pp. 197-214, 2009.
 - [38] S. K. Lai-Yuen and Y. S. Lee, "Computer-aided design and simulation for nano-scale assembly," *Transactions of the North American Manufacturing Research Institution of SME*, vol. 34, pp. 357-364, 2006.
 - [39] M. B. Stocks, S. Hayward, and S. D. Laycock, "Interacting with the biomolecular solvent accessible surface via a haptic feedback device," *BMC Structural Biology*,

- vol. 9, 2009.
- [40] E. Subasi and C. Basdogan, "A new haptic interaction and visualization approach for rigid molecular docking in virtual environments," *Presence: Teleoperators and Virtual Environments*, vol. 17, pp. 73-90, 2008.
 - [41] J. E. Stone, J. Gullingsrud, and K. Schulten, "A system for interactive molecular dynamics simulation," in *Proceedings of the Symposium on Interactive 3D Graphics*, 2001, pp. 191-194.
 - [42] O. Sourina, J. Torres, and J. Wang, "Visual haptic-based biomolecular docking," in *Proceedings of the 2008 International Conference on Cyberworlds, CW 2008* 2008, pp. 240-247.
 - [43] J. K. D. Lisa, J. M. Nicholas, M. W. David, R. J. Chris, and M. H. John, "SCIRun haptic display for scientific visualization," in *Third Phantom User's Group Workshop*, MIT, 1998.
 - [44] J. Cha, Y. S. Ho, Y. Kim, J. Ryu, and I. Oakley, "A framework for haptic broadcasting," *IEEE Multimedia*, vol. 16, pp. 16-27, 2009.
 - [45] A. Kretz, R. Huber, and M. Fjeld, "Force feedback slider (FFS): Interactive device for learning system dynamics," in *Proceedings - 5th IEEE International Conference on Advanced Learning Technologies, ICAIT 2005*, 2005, pp. 457-458.
 - [46] M. Faust, "Haptic Feedback in Pervasive games," *Third International Workshop on Pervasive gaming applications, PerGames*, 2006.
 - [47] K. Salisbury, F. Conti, and F. Barbagli, "Haptic rendering: Introductory concepts," *IEEE Computer Graphics and Applications*, vol. 24, pp. 24-32, 2004.
 - [48] W. A. McNeely, K. D. Puterbaugh, and J. J. Troy, "Six degree-of-freedom haptic rendering using voxel sampling," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, New York, NY, USA, 1999, pp. 401-408.
 - [49] M. Wan and W. A. McNeely, "Quasi-Static Approximation for 6 Degrees-of-Freedom Haptic Rendering," in *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, Washington, DC, USA, 2003, pp. 257-262.
 - [50] J. Barbic, "Real-time Reduced Large-Deformation Models and Distributed Contact for Computer Graphics and Haptics," PhD Thesis, Computer Science Department, Carnegie Mellon University, 2007.
 - [51] D. C. Ruspini, K. Kolarov, and O. Khatib, "Haptic display of complex graphical environments," in *Proceedings of the 1997 Conference on Computer Graphics, SIGGRAPH*, 1997, pp. 345-352.
 - [52] C. H. Ho, C. Basdogan, and M. A. Srinivasan, "Efficient point-based rendering techniques for haptic display of virtual objects," *Presence: Teleoperators and Virtual Environments*, vol. 8, pp. 477-491, 1999.
 - [53] R. S. Avila and L. M. Sobierajski, "A haptic interaction method for volume visualization," in *Visualization '96. Proceedings.*, 1996, pp. 197-204.

- [54] B. A. Payne and A. W. Toga, "Distance field manipulation of surface models," *Computer Graphics and Applications, IEEE*, vol. 12, pp. 65-71, 1992.
- [55] M. A. Otaduy and M. C. Lin, "Stable and Responsive Six-Degree-of-Freedom Haptic Manipulation Using Implicit Integration," in *Proceedings of the First Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, Washington, DC, USA, 2005, pp. 247-256.
- [56] M. Ortega, S. Redon, and S. Coquillart, "A six degree-of-freedom god-object method for haptic display of rigid bodies with surface properties," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, pp. 458-468, 2007.
- [57] X. Hou and O. Sourina, "Stable adaptive algorithm for Six Degrees-of-Freedom haptic rendering in a dynamic environment," *The Visual Computer*, pp. 1-13, 2013.
- [58] J. Barbic and D. L. James, "Six-DoF haptic rendering of contact between geometrically complex reduced deformable models," *IEEE Transactions on Haptics*, vol. 1, pp. 39-52, 2008.
- [59] Y. J. Kim, M. C. Lin, and D. Manocha, "DEEP: dual-space expansion for estimating penetration depth between convex polytopes," in *Proceeding of ICRA '02 IEEE International Conference on Robotics and Automation*, 2002, pp. 921-926.
- [60] D. Wang, X. Zhang, Y. Zhang, and J. Xiao, "Configuration-Based Optimization for Six Degree-of-Freedom Haptic Rendering for Fine Manipulation," *IEEE Transactions on Haptics*, vol. 6, pp. 167-180, April 2013.
- [61] P. M. Hubbard, "Collision detection for interactive graphics applications," *IEEE Transactions on Visualization and Computer Graphics*, vol. 1, pp. 218-230, 1995.
- [62] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, "R-tree. An efficient and robust access method for points and rectangles," in *Proceeding of the 1990 ACM SIGMOD international conference on Management of data*, 1990, pp. 322-331.
- [63] S. Gottschalk, M. C. Lin, and D. Manocha, "OBBTree: A hierarchical structure for rapid interference detection," in *Proceedings of the International Conference on Robotics and Automation*, 2000, pp. 3719-3726.
- [64] S. A. Ehmann and M. C. Liu, "Accurate and fast proximity queries between polyhedra using convex surface decomposition," *Computer Graphics Forum*, vol. 20, pp. C/500-C/510, 2001.
- [65] E. Larsen, S. Gottschalk, M. C. Lin, and D. Manocha, "Fast distance queries with rectangular swept sphere volumes," in *Proceedings. ICRA '00. IEEE International Conference on Robotics and Automation*, 2000, pp. 3719-3726.
- [66] M. De Pascale and D. Prattichizzo, "A framework for bounded-time collision

- detection in haptic interactions," in *ACM symposium on Virtual reality software and technology*, New York, 2006, pp. 305-311.
- [67] P. K. Agarwal, L. J. Guibas, H. Edelsbrunner, J. Erickson, M. Isard, S. Har-Peled, *et al.*, "Algorithmic issues in modeling motion," *ACM Computing Surveys*, vol. 34, pp. 550-572, 2002.
 - [68] J. Kim, Y. Kim, and J. Kem, "Real-time haptic rendering of slowly deformable bodies based on two dimensional visual information for telemanipulation," in *Proceeding of the International Conference on Control, Automation and Systems, ICCAS 2007*, 2007, pp. 2448-2451.
 - [69] J. E. Colgate, M. C. Stanley, and J. M. Brown, "Issues in the haptic display of tool use," in *Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, 1995, pp. 140-145 vol.3.
 - [70] W. McNeely, K. Puterbaugh, and J. Troy, "Voxel-Based 6-DOF Haptic Rendering Improvements," *Haptics-e*, vol. 3, 2006.
 - [71] G. Picinbono and J.-C. Lombardo, "Extrapolation: A Solution for Force Feedback?," in *Proceedings of the 7th GTRV congress*, 1999, pp. 117-125.
 - [72] G. Picinbono, H. Delingette, and N. Ayache, "Non-linear and anisotropic elastic soft tissue models for medical simulation," in *IEEE International Conference on Robotics and Automation*, 2001, pp. 1370-1375.
 - [73] G. Picinbono, J. C. Lombardo, H. Delingette, and N. Ayache, "Improving realism of a surgery simulator: Linear anisotropic elasticity, complex interactions and force extrapolation," *Journal of Visualization and Computer Animation*, vol. 13, pp. 147-167, 2002.
 - [74] G. Picinbono, J.-C. Lombardo, H. Delingette, and N. Ayache, "Anisotropic elasticity and force extrapolation to improve realism of surgery simulation," in *IEEE International Conference on Robotics and Automation*, 2000, pp. 596-602.
 - [75] J. Hu, C. Y. Chang, N. Tardella, J. Pratt, and J. English, "Effectiveness of haptic feedback in open surgery simulation and training systems," in *Studies in Health Technology and Informatics*, 2006, pp. 213-218.
 - [76] J. Wu, A. Song, and J. Li, "A time series based solution for the difference rate sampling between haptic rendering and visual display," in *Proceeding of the IEEE International Conference on Robotics and Biomimetics, ROBIO 2006* 2006, pp. 595-600.
 - [77] K. Lee and D. Y. Lee, "Real-time haptic rendering using multi-rate output-estimation with ARMAX model," in *International Conference on Control, Automation and Systems, ICCAS '07*, 2007, pp. 1821-1826.
 - [78] F. Mazzella, K. Montgomery, and J. C. Latombe, "The forcegrid: A buffer structure for haptic interaction with virtual elastic objects," in *IEEE International Conference on Robotics and Automation*, 2002, pp. 939-946.

- [79] J. Fousek, T. Golembiovsky, J. Filipovic, and I. Peterlik, "Haptic Rendering Based on RBF Approximation from Dynamically Updated Data," in *Sixth Doctoral Workshop on Mathematical and Engineering Methods in Computer Science, MEMICS'10*, Dagstuhl, Germany, 2011, pp. 26-31.
- [80] Y. Liu, W. Chou, and S. Yan, "Proxy position prediction based continuous local patch for smooth haptic rendering," *Journal of Computing and Information Science in Engineering*, vol. 12, 2012.
- [81] M. Ortega, S. Redon, and S. Coquillart, "A six degree-of-freedom god-object method for haptic display of rigid bodies," in *IEEE Virtual Reality 2006*, p. 27.
- [82] G. Turk and M. Levoy, "Zippered polygon meshes from range images," presented at the *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, 1994.
- [83] A. Gregory, A. Mascarenhas, S. Ehmann, M. Lin, and D. Manocha, "Six degree-of-freedom haptic display of polygonal models," in *Proceedings of the conference on Visualization '00*, 2000, pp. 139-146.
- [84] E. Ruffaldi, D. Morris, T. Edmunds, F. Barbagli, and D. K. Pai, "Standardized evaluation of haptic rendering systems," in *Proceedings of the 14th Symposium on Haptics Interfaces for Virtual Environment and Teleoperator Systems 2006*, 2006, pp. 225-232.
- [85] R. Smith. Open Dynamics Engine Available: <http://www.ode.org/>
- [86] C. Duriez and C. Andriot, "A multi-threaded approach for deformable/rigid contacts with haptic feedback," in *Proc. of Haptics Symposium*, 2004, pp. 272-279.
- [87] G. M. J. George E. P. Box, Gregory C. Reinsel *Time Series Analysis: Forecasting and Control, 4th Edition*: Wiley, 2008.
- [88] O. Sourina, J. Torres, and J. Wang, "Visual Haptic-based Biomolecular Docking and its Applications in E-learning," *Transactions on Edutainment II*, vol. LNCS 5660, pp. 105-118, 2009.
- [89] S. Cooper, F. Khatib, A. Treuille, J. Barbero, J. Lee, M. Beenen, *et al.*, "Predicting protein structures with a multiplayer online game," *Nature*, vol. 466, pp. 756-760, 2010.
- [90] O.-y. Ming, D. V. Beard, and F. P. Brooks Jr, "Force display performs better than visual display in simple 6-D docking task," in *IEEE International Conference on Robotics and Automation*, Scottsdale, AZ, USA, 1989, pp. 1462-1466.
- [91] N. Férey, J. Nelson, C. Martin, L. Picinali, G. Bouyer, A. Tek, *et al.*, "Multisensory VR interaction for protein-docking in the CoRSAIRE project," *Virtual Reality*, vol. 13, pp. 273-293, 2009.
- [92] J. Heyd and S. Birmanns, "Immersive structural biology: A new approach to hybrid modeling of macromolecular assemblies," *Virtual Reality*, vol. 13, pp. 245-255, 2009.

- [93] J. E. Stone, J. Gullingsrud, and K. Schulten, "A system for interactive molecular dynamics simulation," presented at the *Proceedings of the 2001 symposium on Interactive 3D graphics*, 2001, pp. 191-194.
- [94] O.-y. Ming, M. Pique, J. Hughes, N. Srinivasan, and F. P. Brooks Jr, "Using a manipulator for force display in molecular docking," presented at the *IEEE International Conference on Robotics and Automation*, Philadelphia, PA, USA, 1988, pp. 1824 - 1829.
- [95] Y. G. Lee and K. W. Lyons, "Smoothing haptic interaction using molecular force calculations," *CAD Computer Aided Design*, vol. 36, pp. 75-90, 2004.
- [96] P. B. Persson, M. D. Cooper, L. A. E. Tibell, S. Ainsworth, A. Ynnerman, and B. H. Jonsson, "Designing and evaluating a haptic system for biomolecular education," in *Proceedings IEEE Virtual Reality 2007*, pp. 171-178.
- [97] S. K. Lai-Yuen and Y. S. Lee, "Computer-Aided Molecular Design (CAMD) with force-torque feedback," in *Ninth International Conference on Computer Aided Design and Computer Graphics*, 2005, pp. 199-204.
- [98] R. A. Sayle and E. J. Milner-White, "RASMOL: Biomolecular graphics for all," *Trends in Biochemical Sciences*, vol. 20, pp. 374-376, 1995.
- [99] W. L. DeLano, "The pymol molecular graphics system," Available: <http://www.pymol.org>
- [100] Jmol: an open-source Java viewer for chemical structures in 3D. Available: <http://jmol.sourceforge.net>.
- [101] O. Sourina and N. Korolev, "Visual Mining and Spatio-Temporal Querying in Molecular Dynamics," *Special issue on Computational Intelligence for Molecular Biology and Bioinformatics of the Journal of Computational and Theoretical Nanoscience*, vol. 2, pp. 492-498, 2005.
- [102] S. K. Lai-Yuen and Y. S. Lee, "Interactive computer-aided design for molecular docking and assembly," *Computer-Aided Design and Applications*, vol. 3, pp. 701-709, 2006.
- [103] H. Nagata, H. Mizushima, and H. Tanaka, "Concept and prototype of protein-ligand docking simulator with force feedback technology," *Bioinformatics*, vol. 18, pp. 140-146, 2002.
- [104] G. Sankaranarayanan, S. Weghorst, M. Sanner, A. Gillet, and A. Olson, "Role of haptics in teaching structural molecular biology," in *Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2003. HAPTICS 2003. Proceedings. 11th Symposium 2003*, pp. 363-366.
- [105] J. F. Prins, J. Hermans, and G. Mann, "A virtual environment for steered molecular dynamics," *Future Generation Computer Systems*, vol. 15, pp. 485-495, 1999.
- [106] B. Daunay, A. Micaelli, and S. Régnier, "Energy-field reconstruction for haptic-based molecular docking Using Energy minimization processes," in *IEEE*

- International Conference on Intelligent Robots and Systems* 2007, pp. 2704-2709.
- [107] R. A. Davies, N. W. John, J. N. MacDonald, and K. H. Hughes, "Visualization of molecular quantum dynamics - A molecular visualization tool with integrated Web3D and haptics," in *Web3D Symposium Proceedings* 2005, pp. 143-150.
 - [108] Q. Liu and A. Sourin, "Function-defined shape metamorphoses in visual cyberworlds," *Visual Computer*, vol. 22, pp. 977-990, 2006.
 - [109] B. E. Miller, J. E. Colgate, and R. A. Freeman, "Guaranteed stability of haptic systems with nonlinear virtual environments," *IEEE Transactions on Robotics and Automation*, vol. 16, pp. 712-719, 2000.
 - [110] L. Wei, A. Sourin, and O. Sourina, "Function-based haptic interaction in cyberworlds," in *Proceedings - 2007 International Conference on Cyberworlds, CW'07* 2007, pp. 225-232.
 - [111] L. Wei, A. Sourin, and O. Sourina, "Function-based visualization and haptic rendering in shared virtual spaces," *Visual Computer*, vol. 24, pp. 871-880, 2008.
 - [112] PDB - Protein Data Bank, Brookhaven National Laboratory. Available: <http://www.rcsb.org/pdb>
 - [113] J. U. Bowie, "Helix packing in membrane proteins," *Journal of Molecular Biology*, vol. 272, pp. 780-789, 1997.
 - [114] W. L. Jorgensen, D. S. Maxwell, and J. Tirado-Rives, "Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids," *Journal of the American Chemical Society*, vol. 118, pp. 11225-11236, 1996.
 - [115] W. Damm, A. Frontera, J. Tirado-Rives, and W. L. Jorgensen, "OPLS all-atom force field for carbohydrates," *Journal of Computational Chemistry*, vol. 18, pp. 1955-1970, 1997.
 - [116] R. C. Rizzo and W. L. Jorgensen, "OPLS all-atom model for amines: Resolution of the amine hydration problem," *Journal of the American Chemical Society*, vol. 121, pp. 4827-4836, 1999.
 - [117] OPLS-aa force field parameter. Available: http://egad.berkeley.edu/EGAD_manual/EGAD/examples/energy_function/ligands/oplsaa.txt
 - [118] S. J. Weiner, P. A. Kollman, D. A. Case, U. C. Singh, C. Ghio, G. Alagona, *et al.*, "A new force field for molecular mechanical simulation of nucleic acids and proteins," *Journal of the American Chemical Society*, vol. 106, pp. 765-784, 1984.
 - [119] M. G. Martin, "Comparison of the AMBER, CHARMM, COMPASS, GROMOS, OPLS, TraPPE and UFF force fields for prediction of vapor-liquid coexistence curves and liquid densities," *Fluid Phase Equilibria*, vol. 248, pp. 50-55, 2006.
 - [120] F. Conti, F. Barbagli, D. Morris, and C. Sewell, "Chai 3d: An open-source library for the rapid development of haptic scenes," presented at the *IEEE World Haptics* 2005.

- [121] F. Conti, F. Barbagli, D. Morris, and C. Sewell, "CHAI 3D: An Open-Source Library for the Rapid Development of Haptic Scenes," in *IEEE World Haptics*, Pisa, Italy, 2005.
- [122] P. L. Nunez and R. Srinivasan, *Electric Fields of the Brain*, 2 ed.: Oxford University Press.
- [123] L. Yisi, O. Sourina, and N. Minh Khoa, "Real-Time EEG-Based Human Emotion Recognition and Visualization," in *2010 International Conference on Cyberworlds (CW'10)*, 2010, pp. 262-269.
- [124] W. Qiang, O. Sourina, and N. Minh Khoa, "EEG-Based "Serious" Games Design for Medical Applications," in *2010 International Conference on Cyberworlds (CW'10)*, , pp. 270-276.
- [125] M. Sato, X. Liu, J. Murayama, K. Akahane, and M. Isshiki, "A Haptic Virtual Environment for Molecular Chemistry Education," in *Transactions on Edutainment I*. vol. 5080, Z. Pan, *et al.*, Eds., ed: Springer Berlin Heidelberg, 2008, pp. 28-39.
- [126] R. Palluel-Germain, F. Bara, A. H. de Boisferon, B. Hennion, P. Gouagour, and E. Gentaz, "A Visuo-Haptic Device - Telemaque - Increases Kindergarten Children's Handwriting Acquisition," in *EuroHaptics Conference, 2007 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics 2007. Second Joint*, 2007, pp. 72-77.
- [127] Z. Minchev, G. Dukov, and S. Georgiev, "EEG Spectral Analysis in Serious Gaming: An Ad Hoc Experimental Application," *BIO Automation*, vol. 13, pp. 79-88, 2009.
- [128] Stroke Rehabilitation. Available: <http://www.ninds.nih.gov/disorders/stroke>
- [129] The Emotiv EEG device. Available: <http://www.emotiv.com/eeg/features.php>
- [130] F. Sharbrough, G.E. Chatrian, R.P. Lesser, H. Luders, M. Nuwer, and T. W. Picton., "American Electroencephalographic Society Guidelines for Standard Electrode Position Nomenclature," *Journal of Clinical Neurophysiology*, vol. 8, pp. 200-202, 1991.
- [131] Stroke Statistics. Available: <http://www.strokecenter.org/patients/about-stroke/strokestatistics>

Appendix A. Haptic Devices

A.1 PHANTOM Premium 1.5/6DOF haptic device

The Premium 1.5 haptic device provides a range of motion approximating lower arm movement pivoting at the elbow, shown in Fig. A.1. It fulfills the requirements of a vast range of research and commercial applications. This high-precision device can provide the largest workspaces and highest forces in the Phantom line while offering a broad range of force feedback workspaces, various ranges of motion and varying stiffness, the specifications shown in Table A.1. This device includes a passive stylus and thimble gimbal and provides three degree-of-freedom positional sensing and three degree-of-freedom force-feedback. An encoder stylus gimbal can be purchased separately, enabling the measurement of pitch, roll & yaw.

Figure A. 1: PHANTOM Premium 1.5 haptic device. It comes in 6-DOF models, which offer six degree-of-freedom (3 translational, 3 torque) in output capabilities.

Table A. 1: Specifications of Phantom Premium 1.5/6DOF, 1.5 High Force/6DOF haptic devices.

			Premium 1.5/6DOF	Premium 1.5 High Force/6DOF
Workspace	Translational		15 W x 10.5 H x 7.5 D in	15 W x 10.5 H x 7.5 D in
			381 W x 267 H x 191 D mm	381 W x 267 H x 191 D mm
	Rotational	Yaw	297 degrees/5.18 radians	297 degrees/5.18 radians
		Pitch	260 degrees/4.54 radians	260 degrees/4.54 radians
Roll		335 degrees/5.85 radians	335 degrees/5.85 radians	
Range of motion			Lower arm movement pivoting at elbow	Lower arm movement pivoting at elbow
Nominal position resolution	Translational		860 dpi/0.03mm	3784 dpi/0.007 mm
	Rotational	Yaw	0.0023 degrees/0.00004 radians	0.0023 degrees/0.00004 radians
		Pitch	0.0023 degrees/0.00004 radians	0.0023 degrees/0.00004 radians
		Roll	0.0080 degrees/0.00014 radians	0.0080 degrees/0.00014 radians
Maximum exertable force and torque at nominal position	Translational		1.9 lbf/8.5 N	8.4 lbf/37.5 N
	Rotational	Yaw	73 oz-in/515 mNm	73 oz-in/515 mNm
		Pitch	73 oz-in/515 mNm	73 oz-in/515 mNm
		Roll	24 oz-in/170 mNm	24 oz-in/170 mNm
Stiffness			20 lbf in ⁻¹	20 lbf in ⁻¹
			3.5 N mm ⁻¹	3.5 N mm ⁻¹
Force feedback (6 Degrees of Freedom)			x, y, z, Tx, Ty, Tz	x, y, z, Tx, Ty, Tz
Position sensing/input (6 Degrees of Freedom)			x, y, z, roll, pitch, yaw	x, y, z, roll, pitch, yaw
Interface			Parallel port	Parallel port
Optional end effectors			Thumb pad (pinch), scissors	Thumb pad (pinch), scissors

A.2 Novint Falcon 3-DOF Haptic Device

Novint Falcon is a USB haptic device intended to replace the mouse in video games and other applications, shown in Fig. A.2. The Falcon has removable handles, or grips, that the user holds onto to control the Falcon to move in three dimensions, Table A.2 shown the specifications. Novint has developed several grip accessories. On the consumer side, Novint developed a pistol grip, which is the shape of a pistol handle and attaches to the Falcon in place of the spherical grip. It has a main trigger button, and 3 side buttons. It is intended for use in First Person Shooter (FPS) games, and it is generally an ergonomic grip that can be used for many applications.

Figure A. 2: Novint Falcon haptic device. It is a consumer product for the control of haptic games.

Table A. 2: Specifications of the Novint Falcon haptic device.

Technical Specifications for Novint Falcon	
3D Touch Workspace	4" x 4" x 4"
Force Capabilities	> 2 lbs
Position Resolution	> 400 dpi
Quick Disconnect Handle	< 1 second change time
Communication Interface	USB 2.0
Size	9" x 9" x 9"
Weight	6 lbs
Power	30 watts, 100V-240V,50Hz-60Hz
Minimum System Requirements:	
Processor:	1.0 GHz Processor
OS:	Windows XP Service Pack 2, Windows Vista
Graphic card:	128Mb 3D hardware accelerated graphics card
DirectX Version:	DirectX 9.0c
Hard Drive:	250 MB free disk space
Memory:	512 MB RAM
Other:	USB 2.0 connection

Appendix B. Molecular Docking

B.1 Protein Data Bank

The Protein Data Bank (PDB) is a repository for the 3-D structural data of large biological molecules, such as proteins and nucleic acids. The PDB file format consists of lines of information in a text. Each line of information in the file is called a record as shown in Table A.3 and Table A.4. A PDB file generally contains several different types of records, arranged in a specific order to describe the three dimensional structures of molecules held in the PDB.

Table A. 3: Protein Data Bank Record Types

Record Type	Data Provided by Record
HEADER, TITLE, AUTHOR	Provide information about the researchers who defined the structure; numerous other types of records are available to provide other types of information.
ATOM	Describe the coordinates of the atoms that are parts of the protein. the first three floating point numbers are its x, y and z coordinates. The next three columns are the occupancy, temperature factor, and the element name, respectively.
HETATM	Describe coordinates of hetero-atoms that are those atoms which are not part of the protein molecule.
TER	Indicates the end of a chain of residues. For example, a hemoglobin molecule consists of four subunit chains that are not connected. TER indicates the end of a chain and prevents the display of a connection to the next chain.
HELIX	Indicates the location and type (right-handed alpha, etc.) of helices.
SHEET	Indicates the location, sense (anti-parallel, etc.) and registration with respect to the previous strand in the sheet (if any) of each strand in the model. One record per strand.
SSBOND	Defines disulfide bond linkages between cysteine residues.

Table A. 4: Protein Data Bank format

Record Type	Columns	Data	Justification	Data Type
ATOM	1-4	"ATOM"		character
	7-11 [#]	Atom serial number	right	integer
	13-16	Atom name	left [*]	character
	17	Alternate location indicator		character
	18-20 [§]	Residue name	right	character
	22	Chain identifier		character
	23-26	Residue sequence number	right	integer
	27	Code for insertions of residues		character
	31-38	X orthogonal Å coordinate	right	real (8.3)
	39-46	Y orthogonal Å coordinate	right	real (8.3)
	47-54	Z orthogonal Å coordinate	right	real (8.3)
	55-60	Occupancy	right	real (6.2)
	61-66	Temperature factor	right	real (6.2)
	73-76	Segment identifier [¶]	left	character
	77-78	Element symbol	right	character
	79-80	Charge		character
HETATM	1-6	"HETATM"		character
	7-80	same as ATOM records		
TER	1-3	"TER"		character
	7-11 [#]	Serial number	right	integer
	18-20 [§]	Residue name	right	character
	22	Chain identifier		character
	23-26	Residue sequence number	right	integer
	27	Code for insertions of residues		character

HELIX	1-5	"HELIX"		character
	8-10	Helix serial number	right	integer
	12-14	Helix identifier	right	character
	16-18 [§]	Initial residue name	right	character
	20	Chain identifier		character
	22-25	Residue sequence number	right	integer
	26	Code for insertions of residues		character
	28-30 [§]	Terminal residue name	right	character
	32	Chain identifier		character
	34-37	Residue sequence number	right	integer
	38	Code for insertions of residues		character
	39-40	Type of helix [†]	right	integer
	41-70	Comment	left	character
	72-76	Length of helix	right	integer
SHEET	1-5	"SHEET"		character
	8-10	Strand number (in current sheet)	right	integer
	12-14	Sheet identifier	right	character
	15-16	Number of strands (in current sheet)	right	integer
	18-20 [§]	Initial residue name	right	character
	22	Chain identifier		character
	23-26	Residue sequence number	right	integer
	27	Code for insertions of residues		character
	29-31 [§]	Terminal residue name	right	character
	33	Chain identifier		character
	34-37	Residue sequence number	right	integer
	38	Code for insertions of residues		character
	39-40	Strand sense with respect to previous [‡]	right	integer
	The following fields identify two atoms involved in a hydrogen bond, the first in the current strand and the second in the previous strand. These fields should be blank for strand 1 (the first strand in a sheet).			
	42-45	Atom name (as per ATOM record)	left	character
	46-48 [§]	Residue name	right	character
	50	Chain identifier		character
	51-54	Residue sequence number	right	integer

	55	Code for insertions of residues		character
	57-60	Atom name (as per ATOM record)	left	character
	61-63 [§]	Residue name	right	character
	65	Chain identifier		character
	66-69	Residue sequence number	right	integer
	70	Code for insertions of residues		character
SSBOND	1-6	"SSBOND"		character
	8-10	Serial number	right	integer
	12-14	Residue name ("CYS")	right	character
	16	Chain identifier		character
	18-21	Residue sequence number	right	integer
	22	Code for insertions of residues		character
	26-28	Residue name ("CYS")	right	character
	30	Chain identifier		character
	32-35	Residue sequence number	right	integer
	36	Code for insertions of residues		character
	60-65	Symmetry operator for first residue	right	integer
	67-72	Symmetry operator for second residue	right	integer
	74-78	Length of disulfide bond	right	real (5.2)

Chimera allows (nonstandard) use of columns 6-11 for the integer atom serial number in ATOM records, and in TER records, only the "TER" is required.

* Atom names start with element symbols right-justified in columns 13-14 as permitted by the length of the name.

§ Chimera allows (nonstandard) use of four-character residue names occupying an additional column to the right.

¶ Segment identifier is obsolete, but still used by some programs.

Appendix C. Calculation of Motion and Friction

C1. Motion equation for rigid body

For a rigid body, we define the state vector of the rigid body is $\mathbf{Y}(t)$. The position $x(t)$, orientation $R(t)$, linear momentum $P(t)$, and angular momentum $L(t)$ can be defined as follows:

$$\mathbf{Y}(t) = \begin{bmatrix} x(t) \\ R(t) \\ P(t) \\ L(t) \end{bmatrix} \quad (\text{A.1})$$

The derivative $\frac{d}{dt}\mathbf{Y}(t)$ is:

$$\frac{d}{dt}\mathbf{Y}(t) = \frac{d}{dt} \begin{bmatrix} x(t) \\ R(t) \\ P(t) \\ L(t) \end{bmatrix} = \begin{bmatrix} v(t) \\ w(t) * R(t) \\ F(t) \\ \tau(t) \end{bmatrix} \quad (\text{A.2})$$

where $v(t)$ is the linear velocity, $w(t)$ is the angular velocity, $F(t)$ is the total force, and $\tau(t)$ is the total torque, The mass M and inertia I_b are defined before simulation as constant values. At the simulation time, the quantities $v(t)$, $w(t)$ and $I(t)$ are calculated as:

$$v(t) = \frac{P(t)}{M} \quad (\text{A.3})$$

$$w(t) = I(t)^{-1} L(t) \quad (\text{A.4})$$

$$I(t) = R(t) I_b R(t)^T \quad (\text{A.5})$$

C2. Friction calculation

For the static friction, we assume the surface has a static friction constant μ_s . The contact force can be divided into two components: force tangential to the contact surface f_t and force normal to the contact surface. The virtual tool will keep static, if:

$$\|f_t\| \leq \mu_s \|f_n\| \quad (\text{A.5})$$

For the dynamic friction on a surface, it can be calculated from the mass M , the viscous damping b , and the Coulomb friction:

$$f_t - \mu_d f_n = M\ddot{x} + b\dot{x} \quad (\text{A.6})$$

Author's Publications

1. O. Sourina, and **X. Hou**, Visual Haptic-enabled Biomolecular Docking System Design, *Scientific Visualization*, 2(1), pp.49-58, 2010
2. **X. Hou** and O. Sourina, Haptic Rendering Algorithm for Biomolecular Docking with Torque Force. In Proc. 2010 Int. Conf. on Cyberworlds, Singapore, 20-22 Oct, 2010, pp.25-31, 2010
3. **X. Hou** and O. Sourina, Six Degree-of-Freedom Haptic Rendering for Biomolecular Docking, *Transactions on Computational Science*, vol. LNCS 6670, pp. 98-117, 2011.
4. **X. Hou** and O. Sourina, Stable Dynamic Algorithm Based on Virtual Coupling for 6-DOF Haptic Rendering, in Proc. of 2012 International Conference on Cyberworlds Germany, 2012, pp. 157-164.
5. **X. Hou** and O. Sourina, "Stable adaptive algorithm for Six Degrees-of-Freedom haptic rendering in a dynamic environment," *The Visual Computer*, vol. 29, 2013, pp. 1063-1075.
6. **X. Hou** and O. Sourina, "Emotion-enabled Haptic-based Serious Game for Post Stroke Rehabilitation", In Proceeding of 19th ACM Symposium on Virtual Reality Software and Technology (VRST2013), Singapore, 2013, pp 31-34.
7. **X. Hou** and O. Sourina, "A Prediction Method Using Interpolation for Smooth Six-DOF Haptic Rendering in Multirate Simulation", in Proceedings of 2013 International Conference on Cyberworlds (CW2013), Japan, 2013, pp 294-301.
8. **X. Hou** and O. Sourina, "Visual Haptic-based Collaborative Molecular Docking", In Proceeding of the 15th International Conference on Biomedical Engineering (ICBME 2013), Singapore, December 2013, pp 360-363.