論文 / 著書情報
Article / Book Information

題目(和文)	
Title(English)	Haptic Proxy for Dynamic Deformable Objects
著者(和文)	丁 海陽
Author(English)	Ding Haiyang
出典(和文)	学位:博士(工学), 学位授与機関:東京工業大学, 報告番号:甲第11585号, 授与年月日:2020年9月25日, 学位の種別:課程博士, 審査員:長谷川 晶一,中本 高道,山村 雅幸,青西 亨,小野 功
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第11585号, Conferred date:2020/9/25, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

TOKYO INSTITUTE OF TECHNOLOGY

DOCTORAL THESIS

Haptic Proxy for Dynamic Deformable Objects

Author:
Haiyang Ding

Supervisor:
Shoichi Hasegawa

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Engineering

in the

Hasegawa Lab
Interdisciplinary Graduate School of Science and Engineering

August 18, 2020

TOKYO INSTITUTE OF TECHNOLOGY

Abstract

Hasegawa Lab
Interdisciplinary Graduate School of Science and Engineering

Doctor of Engineering

Haptic Proxy for Dynamic Deformable Objects

by Haiyang Ding

Previously, haptic rendering suffers from the "tunneling" problem while interacting with dynamic deformable objects. Also, the simulation of a deformable haptic tool generally induces the force artifacts problem. In this paper, we propose a continuous collision detection method of the virtual proxy for three Degree-of-Freedom haptic rendering to solve the "tunneling" problem. After that, we extend this to a general method for simulating a six Degree-of-Freedom haptic tool using a multi-sphere proxy. The force artifacts, i.e., the feeling of tool inertia, is eliminated. As a result, we can simulate a deformable haptic proxy while stably interacting with multiple complex and dynamic deformable triangular models without losing contact or suffering force artifacts.

Acknowledgements

Here, I greatly appreciate my supervisor, Associate Professor Shoichi Hasegawa, for his kind, patient, attentive guidance, and teaching throughout my Doctoral study. Also, I would like to thank Assistant Professor Hironori Mitake for his advice and suggestion on the problems.

This work is the fruit of the support of Haselab members. I am grateful to all lab members who helped me and all Springhead Physics Engine developers for their support.

Lastly, I would like to show my deep appreciation to my mother and other family members, who dedicate firmly support and trust.

Contents

List of Figures

List of Abbreviations

CCD	**C**ontinuous **C**ollision **D**etection
C-Obstacles	**C**onfiguration space **O**bstacles
DCD	**D**iscrete **C**ollision **D**etection
DHS	**D**egree-of-freedom **H**aptic **S**ystem
DoF	**D**egree-**o**f **F**reedom
FEM	**F**inite **E**lement **M**odel
FPS	**F**rame-**P**er-**S**econd
HIP	**H**aptic **I**nterface **P**oint
MSP	**M**ulti-**S**phere **P**roxy
OP	**O**riented **P**articles
PBD	**P**osition-**B**ased **D**ynamics
SMM	**S**hape-**M**atching **M**ethod
XPBD	e**X**terned **P**osition-**B**ased **D**ynamics
ZoD	**Z**ero-**o**rder **D**ynamics

List of Symbols

$\arg\min_{i \in S} F(x)$	argument of the minima of function $F(x)$ of set S of i
$\arg\max_{i \in S} F(x)$	argument of the maxima of function $F(x)$ of set S of i
α	stiffness of PBD constraint
$\mathbf{a}, \mathbf{b}, \mathbf{c}$	vertices of the triangle in triangle-point CCD
C_{haptic}	haptic constraint
C_{op}	OP deformation constraint
D	haptic device cofiguration
$\mathbf{d}_{dev}$	device position
$detection\_range$	the range of detection
$dist_{p\_to\_\triangle_j}$	distance from proxy to triangle j
$dist_{pop}$	distance of one operation of Proxy Pop-out
$\mathbf{e}$	the moving point in triangle-point CCD
$\mathbf{F}$	haptic contact force
$\mathbf{f}_i$	haptic feedback force of proxy particle i
$\mathbf{f}_{i,t}$	force of haptic torque of proxy particle i
$\mathbf{f}_{o,i}$	force applied on OP particle i
$\mathbf{f}_{vi}$	haptic force applied on vertex
$\mathbf{g}_i$	goal position of haptic constraint of proxy particle i
H	haptic feedback
$in\_range\_triangle\_list$	a set of triangles in a curtain range
$iteration\_limit$	the Proxy Pop-out iteration limit count
j_{shrink}	shrink ratio
k_{haptic}	spring of 3-DoF haptic feedback
L	intermediate representation
m_i	mass of particle i
$\mathbf{n}$	subject normal
$\mathbf{o}_i$	proxy particle original barycentric coordinates before deformation
$\mathbf{p}$	position of the proxy
$\mathbf{p}_i$	position of the proxy particle i
$\mathbf{p}_{start}$	start proxy position of the additional traditional proxy collision detection
Q_{dev}	device orientation matrix
$\Re$	set of real numbers
$\mathbf{r}_j$	distance vector of haptic torque of proxy particle i from the object center
r_{new}	new shrunk proxy sphere radius
r_{origin}	proxy sphere original radius
r_{proxy}	proxy sphere radius
S	the set of contact-on-face co-planar triangles
$solverIterationNum$	iteration number of PBD constraint solver
Δt	time interval between two time-steps
t_0	start moment of one time-step
t_1	end moment of one time-step
t_μ	contact moment of one time-step

$\mathbf{u}$	predicted OP particle position
$\mathbf{v}$	OP particle velocity
W	the set of contact-on-face co-planar triangles
w_a, w_b, w_c	barycentric weights of proxy contact point on co-planar triangles
$w_{j,i}^{blend}$	weight of linear blending skining
w_j^{mask}	weight of scalar pressure mask
$\mathbf{x}_i$	current position of OP particle

To my family, otherwise, I would not make this far;
To all my friends, who fill my life with joy and love.

Chapter 1

Introduction

1.1 Haptic Rendering with Deformable Objects

Starting from the 20th century, the research of haptics has been a popular topic over the decades since it can provide force feedback to users. Haptic rendering refers to the method of how to compute and present the interaction and force feedback. With this technique, users can haptically interact with objects in the virtual environment with a haptic device. The haptic tool is the representation of the device in the virtual environment controlled by the user.

For simulated virtual environments, the basic haptic rendering methods have been proposed for problems, such as how to interact with a rigid or deformable virtual object haptically. Recently, studies are focusing on establishing advanced haptic systems that support accurate physics-based simulation with realistic haptic interactions [Xia18].

Research on haptic rendering with deformable virtual objects is desirable since it has many exciting applications such as surgical training with soft tissue or muscles, education, demonstration, and entertainment [VPDL12][Xia18]. Nevertheless, the simulation of deformation is computationally intensive, which always results in a slow rate of updates. At the same time, haptic rendering requires a fast update rate for high fidelity and stability.

Previous studies have solved those problems using fast and less accurate physics models and a multi-rate simulation scheme. Despite previous studies' achievements, there are still limitations and challenges of a transparent haptic rendering with dynamic and deformable objects. The dynamic deformable objects in this thesis refer to the simulated virtual objects which can move and deform freely.

1.2 Challenges

1.2.1 The Tunneling Problem

As for the dynamic objects, haptic systems suffer from the "tunneling" problem. The "tunneling" problem describes collision detection failure while objects move towards each other when using discrete collision detection (DCD). DCD is a simple and efficient collision detection method, but contacts can only be discovered when intersection occurs in the exact moments of discrete simulation steps.

There are several compromising solutions to this problem: reducing the time-step size; performing multiple DCDs on the path of the motion. Alternatively, a more accurate but usually computational intensive solution is to detect all the collisions on the object motion path, which is called continuous collision detection (CCD). Compare to DCD, CCD can find all the contacts during the motion of the object, therefore, the object does not tunnel any object and the problem is solved.

However, most studies only employed a half-CCD for haptics, which only considered the motion of the haptic tool while the other objects are static. Therefore, the "tunneling" problem may happen when the user interacts with a thin and fast-moving object.

1.2.2 The Force Artifacts

A deformable haptic tool is generally simulated using second-order dynamics since the implementation of haptics is simple. Previous studies use a bidirectional viscoelastic coupling scheme between the device and the haptic tool to simply improve the stability of controlling this tool [CSB95]. However, this method can not eliminate the artificial forces, i.e., the feeling of tool inertia caused by second-order dynamics of the tool. During the simulation, forces artifacts disturb user's operation whenever the user moves. With the force artifacts, the user can not perceive the contact forces of the haptic tool directly while interacting with other objects, therefore, it reduces the haptic rendering transparency.

The proxy refers to the virtual proxy method proposed by Ruspini et al.[RKK97]. This method simulates the haptic tool, the proxy, with Zero-order Dynamics (ZoD) which produces no force artifacts, however, it is only for three-degree-of-freedom (3-DoF) haptic rendering and cannot catch dynamics objects (no CCD with moving objects).

1.3 Structure of Thesis

This study proposes a haptic proxy method to overcome the challenges mentioned above.

The structure of the dissertation is managed as follows: firstly, the context of this research, including the background knowledge and related works, is introduced in Chapter 2. After that, an overview of the proposal, including the system architecture and simulation scheme, are provided in Chapter 3. Then, this study offers a description of the two systems: a 3-DoF haptic system (3-DHS) which presents CCD haptic interactions with a single proxy sphere against deformable objects in Chapter 4; a 6-DHS as a general proposal for all kinds of haptic tools, which support deformation without force artifacts in Chapter 5. The first system explains how to correctly compute the collision between the proxy and the dynamic deformable objects. The second one describes how to render a general deformable haptic proxy without force artifacts. Finally, the proposal and discussion of the prospection are summarised in Chapter 6.

1.4 Contributions

This thesis presents a general haptic proxy which remains the good properties of the traditional proxy, such as supporting probing interactions and having no force artifacts. Simultaneously, several improvements are added to cover its defects, such as solving the "tunneling" problem with thin and dynamic objects and supporting 6-DoF haptic rendering.

This thesis has the following contributions:

- A haptic rendering method enables probing interactions with deformable triangular mesh models simulated using Oriented Particles (Section 4.1.3);
- A continuous collision detection and handling method of triangle-proxy contact (Section 4.2);
- A multi-sphere proxy model which enables CCD contacts for 6-DoF haptic interactions with dynamic deformable objects (Section 5.2);

- A zero-order dynamic simulation of a deformable 6-DoF haptic tool which eliminates the force artifacts of the tool inertia (Section 5.3);
- A multi-rate haptic rendering scheme that provides stable haptic interactions with mesh-level contact constraints (Section 4.3.1).

The proposed methods are open-sourced and can be easily adapted to medical applications, VR demonstrations, and game engines, such as Unity. This study focuses on the triangular model since it is the most simple and widely used model to simulate dynamic deformable objects.

Chapter 2

The Context

This chapter introduces the background and the related works of this dissertation.

2.1 Haptic Rendering with Zero-Order Dynamics

The haptic rendering of an interactive system of a virtual environment is to compute the force, such as the contact force or the object's stiffness. This thesis only focuses on how to render the contact force.

To compute the contact force, the collision between the user and the virtual object must be handled. Therefore, a virtual object controlled by the user, i.e., the haptic tool, represents the haptic device shape, position, and orientation. After that, a scheme of update the tool position or tool dynamics has to be selected.

Zero-order Dynamics (ZoD) [MN04] describes the method of the object dynamics simulation by directly manipulating the object position without considering its velocity, mass, and acceleration. ZoD is the simplest way to simulate moving objects compared to first-order dynamics [MN04; MN07] which considers the velocity, or second-order dynamics [BHB99; OL05b] which involves the mass and acceleration.

In the early days of haptic research, haptic devices only support the translational input and feedback, i.e., 3-Degree-of-Freedom haptic rendering. The haptic force is simply computed from the penetration of the device from the object surface [MS+94]. This method suffers from force discontinuities when the user travels between two surfaces and also difficult to interact with thin objects. As a solution to avoid penetrations, Zilles and Salisbury [ZS95] used a constrained-based god-object method to achieve intuitive haptic interactions with arbitrary geometries by controlling a god-like point. The term "god object" [DZ93] has been proposed to describe the object that only follows the user's control instead of physics laws. During the contact, Lagrange multipliers were used to find the new god object position which is the point that locally minimizes the distance from the god object and the place of the haptic device. The virtual point that represents the device location is called Haptic Interface Point (HIP). After that, the virtual proxy method proposed by Ruspini et al. [RKK97] replaced the god object with a finite-radius sphere to cover the small gap caused by the numerical errors and also help the user visually locate its position. Besides, it also improves the haptic interaction smoothness against sharp corners. Both god object [ZS95] and virtual proxy [RKK97] have no mass or velocity and were updated instantly when the new position is computed, which is the same as ZoD. A more detailed introduction about virtual proxy is provided in Section 4.1.

Three-DoF haptic rendering provides only the translational feedback of the shape and the virtual object's stiffness. In practice, many studies also intend to use a specific shape of the haptic tool with rotational input and feedback. For example, a haptic simulator of dental, endoscopy, laparoscopy, or orthopedics requires 6-DoF haptic rendering of the instruments [EC+16]. Proposals are started with penalty-based methods [Gre+00; Kim+02;

OL05a; Has+04; CSC04; OL05b]. In order to eliminate the interpenetrations, constraint-based methods [BHB99; Dur+05; ORC06; GO10; Gar+11; Wan+15] are proposed. Generally among all of those methods for 6-DoF haptic rendering, if the dynamics of the haptic tool is performed (whether first-order or second-order), the force artifacts, i.e, the feeling of the tool inertia, are also introduced. Therefore, virtual coupling[CSB95], a bidirectional viscoelastic coupling between the tool and HIP, is introduced to stabilize the haptic interactions. Ortega et al.[ORC06]'s method is an exception. They achieved no force artifact using a constraint-based quasi-static god object method, which is similar to ZoD. However, their method only supports rigid object simulation.

2.2 Haptic Rendering with Deformable Objects

The real-time simulation of deformable objects is difficult compared to the one simulating rigid objects because the simulator has to update the object geometry in each time-step. For a rigid object, once the external force and torque applied to the object is acquired, the new position and orientation can be integrated with respect to time. Vertices on the rigid object can be simply updated according to the original shape of the object. On the other hand, any inputs on a deformable object will trigger the intensive computation of the stress, stain, or constraint to update the shape and relative position of the model of a deformable object. At the same time, the local geometry changes for collision detection has also needed to updated. Those are the difficulties of simulating deformable objects in real-time.

A haptic system is generally a real-time system. The term real-time refers to the real-time computation of the simulation and real-time response of haptic interaction. With a real-time interactive system, users can make inputs and get outputs in a very short period (in the order of milliseconds). The human visual system perceives motion while the frame rate is higher than 12 frame-per-second (FPS) [RM00], therefore, the physics simulation should run at least higher than this level to provide the animation of the simulation.

Since haptic rendering requires real-time simulation, the method for deformation has to be fast enough for real-time response. Pre-computed models, such as pre-computed Green's functions, can boost the simulation of linear elastostatic models with point-like or modest multi-point haptic interactions[BNC96; CDA99; JP05; JP03; JCC05]. Nevertheless, those methods limited the haptic interactions on pulling on certain vertices instead of probing or sliding on the surface. For real-time models, mass-spring model[CCO02; dCL99; SML02; MML02] and Finite Element Model (FEM)[Ber+99; BJ08; GO10; Gar+11; Pet+11; TOT13] are most popular. Garre et al. [Gar+11] achieved haptic interactions with a deformable object using a deformable hand (displayed in Figure 2.1).

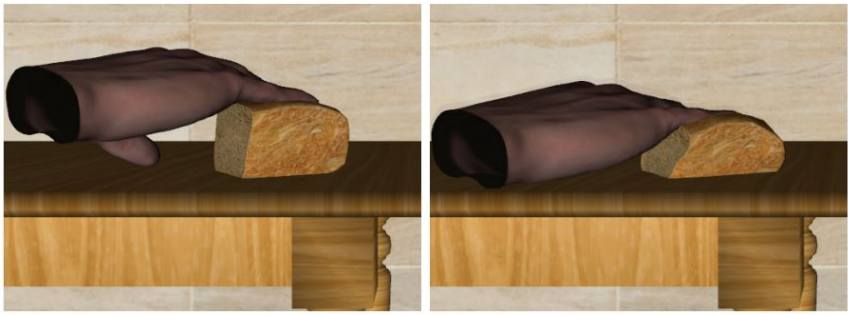

FIGURE 2.1: Study [Gar+11] on Haptic Interactions with a Deformable Object using a Deformable Hand ©[2011] IEEE

Recently, other fast methods with haptics are also drawing attention, such as using shape-matching[Tia+13; Tia+15], stretching, volume conservation, and energy preserving constraints[Pan+15] of Position-based Dynamics[Mül+07]. However, most of those methods cannot perform probing haptic interactions like virtual proxy[RKK97].

2.3 Haptic Rendering with Continuous Collision Detection

The contact force calculation requires correct collision detection and collision responses in real-time for a virtual environment with haptics. For a real-time simulator, the simulation is generally performed with discrete time-steps. Compare to continuous time simulation, a discrete one is easy to implement and compute, which is a better match for time-critical applications. Therefore, a straight forward collision detection method, called discrete collision detection (DCD), is introduced. The DCD finds collisions by testing intersections between all pairs of the objects in one time-step. However, a thin and fast-moving object may pop-through another object within one time-step resulting in no intersections. The contact between those objects can not be detected using DCD. This phenomena is also called the "tunneling" problem or pop-through effect.

As a example displayed on the left of Figure 2.2, the object A is moving fast towards object B. Using discrete time, i.e., time-steps, object A will transport its position from time-step t_0 to t_1 according to the velocity v and time interval. In both t_0 and t_1, the DCD found no intersections of the two objects. Therefore, even though object A go through object B, the collision between them can not be detected.

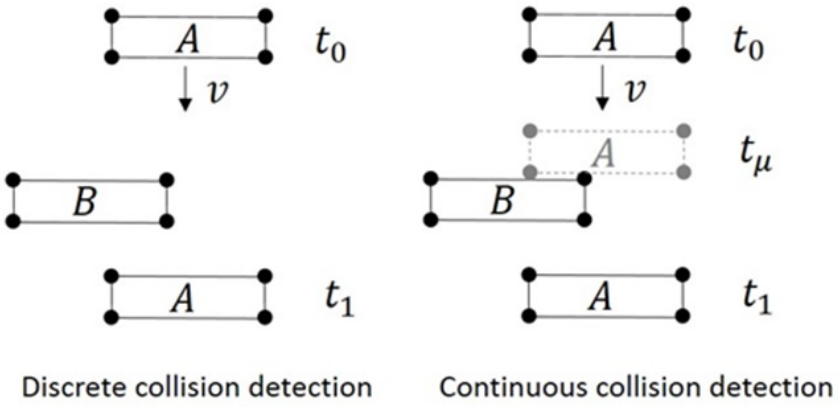

FIGURE 2.2: Discrete Collision Detection versus Continuou Collision Detection

To solve the "tunneling" problem, continuous collision detection (CCD) is proposed [Pro97; RKC02]. CCD can find the contact between the moving objects by detecting collisions on the path of the object motion with more computation cost. As illustrated on the right of Figure 2.2, the contact moment t_μ between the two objects can be found.

In case of haptic rendering, penalty-based methods[MS+94; Gre+00; Kim+02; OL05a; Has+04; CSC04; OL05b] only perform a DCD to find the penetration for haptic force calculation. On the other hand, constraint-based methods are using DCD[BHB99], CCD[ZS95; RKK97; DAK04; GO10; Gar+11; XB16], or configuration-based [Wan+12a] method. Some of them[ZS95; RKK97; Wan+12a; XB16] perform "half-CCD", which can find the collisions between the moving tool against the static object on the path from the tool start position to HIP. However, in order to find the contact with dynamic objects, a completed CCD is required. Garre et al.[GO10; Gar+11] utilize a CCD method for contacts between triangles (triangle-triangle contacts)[Ota+09] for both tool and object which are also both deformable.

2.4 Multi-rate Haptic Rendering

Human perception of haptics is sensitive to vibrations, even greater than 500Hz[BJ+88]. Therefore, the fidelity of haptic rendering is strongly related to the refresh rate of the device output and the communication rate between the device and the virtual environment. However, a simulation of a complex scene, which may consist of multiple physics properties and high-resolution models, will slow down the simulation rate.

A common solution to this problem is to use separated threads, i.e., multi-rated threads, for haptics and dynamic simulation. The dynamic thread computes the full collision detection and heavy dynamics simulation at a low rate. Only the local part as the intermediate representation is synchronized to the haptic thread. The haptic thread only computes the contact force and torque with a simplified model and communicates with the device at a high rate.

The earliest sample was the study proposed by Adachi et al.[AKO95], in which they only used a virtual plane for haptic force computation. For deformable objects, Murat Cenk Çavuşoğlu and Frank Tendick[cT00] proposed a linearized deformable sub-model while the rest is static. Force extrapolation was applied in [Pic+00]. Christan Duriez and Claude Andriot[DAK04] share the contact configurations and solve a linear system in the haptic thread for interactions between a deformable tool and rigid object. The contact linear complementary problem was shared between the two threads and solved in the haptic thread in [SDC08], which was later generalized for all kinds of constraints in [Pet+11].

2.5 Position Based Dynamics

Traditional second-order simulators accumulate forces, such as internal or external forces, to compute the accelerations following Newton's second law of motion. After that, the corresponding velocity and position can be integrated by time integration. Position-based Dynamics (PBD)[Mül+07], however, directly manipulates the position according to the constraints and then deduces the corresponding velocity. Therefore, PBD has direct control over the object to prevent the overshooting with a straightforward design of position constraints. This makes PBD a semi-implicit (also called symplectic) Euler method. Nevertheless, PBD does not guarantee the simulation accuracy. Multiple physics properties can be easily implemented in PBD [BMM15].

As for deformations, shape-matching[Mül+05] is a good example of deformation method used in PBD. It computes the configuration of the closest rigid frame of the deformed object and matching the object deformed-state to rigid-state using the rigid frame over stiffness control. Oriented Particles (OP)[MC11] is an extension of shape-matching. It uses particles with orientation information to reduce the particle number of shape-matching model and ellipsoid shape for approximate collision handling.

Tian et al.[Tia+13] implemented their haptic rendering methods with shape-matching. Their idea was constraint particle coupling, which naturally calculates the haptic feedback force as the derivative of the local collision energy. Later, Tian et al. [Tia+15] improved this method by adding ghost particles to maintain the original optimal translation. However, their system could not handle probing haptic interactions since they did not solve the collision constraint of the surface mesh.

Recently, Macklin et al.[MMC16] proposed an extension of PBD called extended position-based dynamics (XPBD). XPBD is an approximation method derived from the potential energy equation which provides an approximated constraint force, such as a spring force, cloth constraint, or haptic force. The stiffness of the XPBD constraints is then no longer dependent

on the time interval or the number of iterations. The method proposed in this dissertation can also be used in XPBD.

2.6 Relation Between This Thesis and Related Works

As mentioned above, only the studies, [GO10] and [Gar+11], can perform a "full-CCD" while both the object and the haptic tool is deformable. However, those methods cannot solve the force artifacts problem. Besides, the CCD is performed between triangles. Nevertheless, for 3-DoF haptic rendering, generally, a virtual sphere is used. To perform CCD, this sphere can only be modeled with meshes, therefore, it is more computationally expensive and less accurate in contact response than an ideal sphere. Furthermore, the mesh sphere will also create unsmooth force feedback during the probing interaction. Those are the motivations of this thesis.

Chapter 3

Overview of The Proposals

This chapter introduces an overview of the proposed methods, the two implemented systems: a 3-DoF haptic system (3-DHS) and a 6-DHS, and a general simulation scheme of systems.

3.1 The Proposals

To haptically interact with the deformation model, i.e., the OP model in this thesis, this study proposes a distributed point-like method to transfer haptic contact forces to the OP model in Section 4.1.3. After that, a Proxy Shrink method is presented to improve the collision robustness in Section 4.1.4.

For the "tunneling" problem, the proposal is a triangle-proxy CCD with the Proxy Pop-out method introduced in Section 4.2. These methods split the CCD process into two: find a secure proxy position by detecting collisions between the static proxy and the moving meshes; use this secure proxy position as a start and find collisions of the moving proxy against the static mesh. Problem of multi-deformable triangles is discussed and solved in Section 4.2.4 using the Proxy Pop-out method.

For the force artifacts, the traditional proxy method is a solution, however, it is limited with 3-DoF haptic rendering. This thesis extends the method proposed in Section 4.2.2 for 6-DHS in Section 5.2. This method utilizes multiple proxies as MSP model to represent the haptic tool. A haptic constraint is proposed to apply the device input. Nevertheless, this method triggers an extra "tunneling" problem when computing the position-based constraints. This problem is solved by presenting an additional half-CCD. Finally, the proposed 6-DHS system can solve the "tunneling" problem and eliminate the force artifacts.

3.2 System Architecture

For both systems, a multi-rate scheme is constructed to improve the fidelity of haptic rendering. The dynamics thread is used to simulate the full scene including collision detection, constraint computation, and time integration. The haptic thread only computes the force (for 3-DHS) or force and torque (for 6-DHS) of the feedback according to the intermediate representation synchronized from the dynamics thread.

This structure is shown in Figure 3.1. Both dynamics thread and haptic thread retrieve the configuration D of the device. For 3-DHS, D is the device position, while for 6-DHS, D is the device position and orientation. After the collision detection in dynamics thread, the intermediate representation L is transferred to the haptic thread. In 3-DHS, L is the contact constraints. Differently, L is the proxy particle position and contact information in 6-DHS. The haptic thread computes force for both 3-DHS and 6-DHS, while torque is only calculated for 6-DHS. Those feedback is marked as H, and passed to the haptic device. For more details, refer to Chapter 4 for 3-DHS and Chapter 5 for 6-DHS.

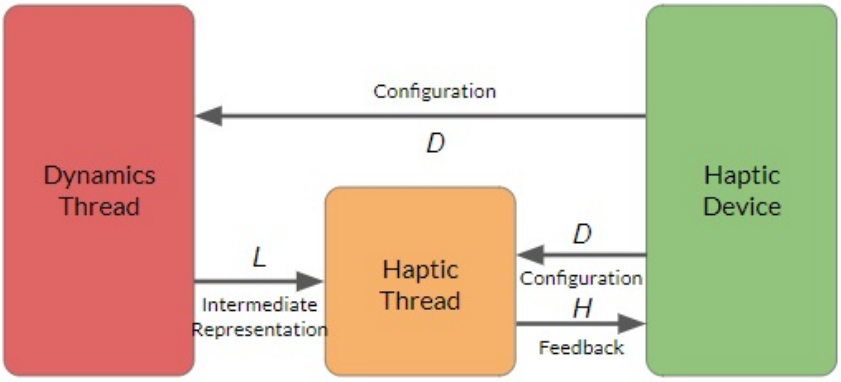

FIGURE 3.1: System Architecture

3.3 Simulation Scheme

Both the two proposed systems simulate the dynamics using PBD[Mül+07]. As displayed in Figure 3.2, this study separates the simulation of the haptic proxy and other virtual objects. For the haptic proxy, this study performs CCD between the proxy and the triangular meshes to secure the start of proxy collision detection. Then, the traditional proxy method is executed as the proxy dynamics and compute the contact forces. After that, this study applies those forces to the contact objects as external forces and simulate the rest.

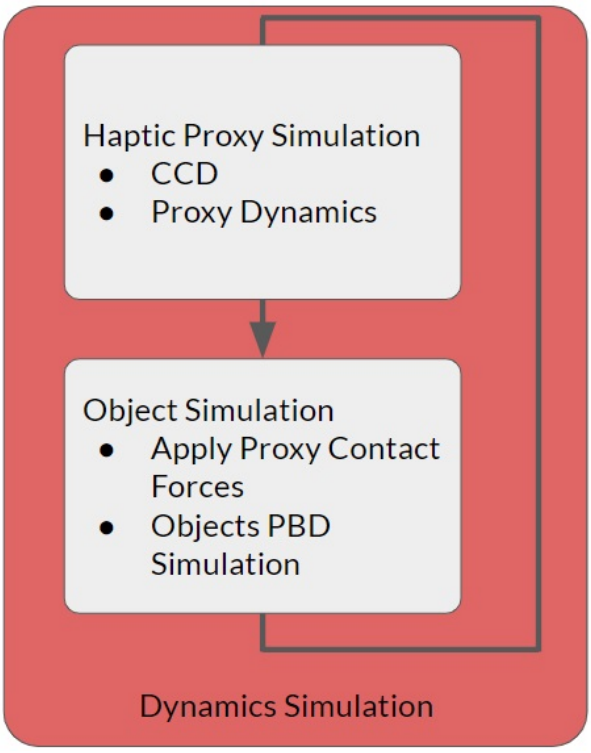

FIGURE 3.2: Simulation Scheme of The Dynamics

The reason for computing the proxy dynamics first is to remain a semi-implicit simulation. Suppose both the proxy and other objects are simulated at the same time. In that case, the contact forces can only be applied in the next time-step, which will turn the simulator into an explicit one and greatly reduce the system stability.

Chapter 4

Three-Degree-of-Freedom Haptic Proxy with Dynamic Deformable Objects

This chapter describes the method of a 3-DoF haptic proxy for interaction with dynamic deformable objects. First, the background, including the virtual proxy method and the OP method, is introduced. Then, the haptic force computation method for OP is proposed. After that, the main problem during the proxy interaction with dynamic deformable objects is addressed together with the solution. Finally, the evaluation and results of the proposed system of the solution are presented.

4.1 Basic Methods for Haptic Rendering

4.1.1 Virtual Proxy Method

The virtual proxy method[RKK97] is a widely used haptic rendering method that simulates a virtual sphere to represent the haptic device in the virtual environment. As shown in Figure 4.1, a finite-radius sphere is called the proxy. The haptic interface point (HIP) indicates the actual position of the device controlled by the user.

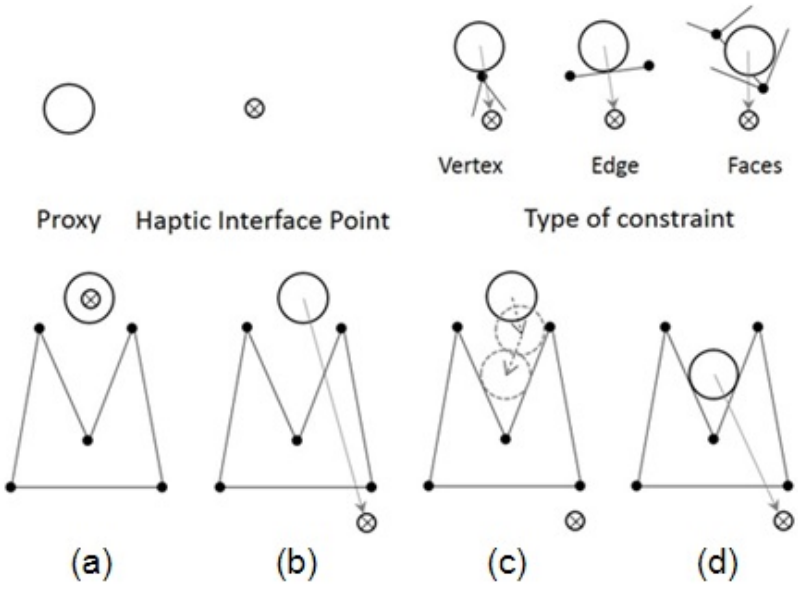

FIGURE 4.1: Virtual Proxy Method

Before the simulation starts, the proxy is initialized to be the same position as the HIP (Figure 4.1 a). In one time-step, the user moves the device which triggers the HIP to move, and the proxy attempts to follow until a collision occurs. The collision is detected on the line segment from the proxy to the HIP between the proxy sphere and the object configuration space obstacles (C-Obstacles) (Figure 4.1 b).

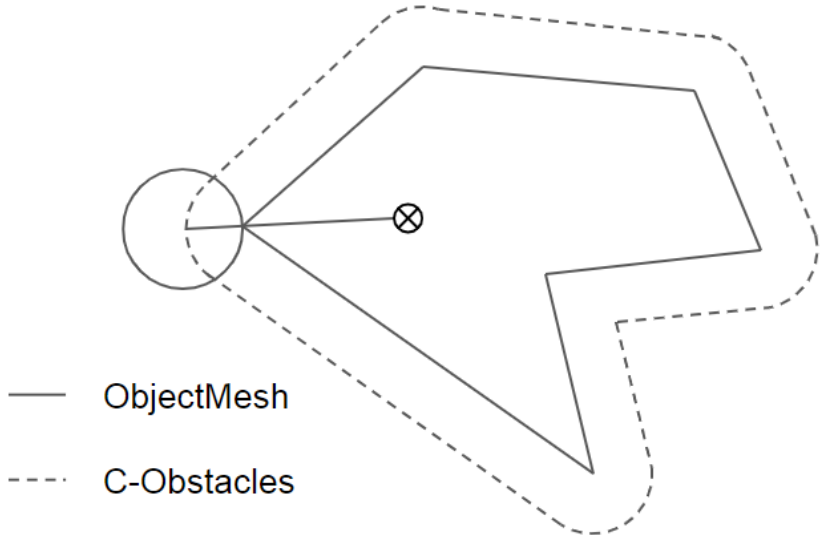

FIGURE 4.2: The Configuration Space Obstacles

As shown in Figure 4.2, the C-Obstacles is the configuration space of all the points of the proxy center position on the object surface, therefore, triangles are protruded, edges become cylinder, and vertices are converted to spheres. The collision between the proxy sphere and the mesh object is transformed into the proxy center point against the C-Obstacles.

After the first contact is found, the proxy tries to minimize the distance to the HIP by calculating a sub-goal constrained by the object surface primitives (vertex, edge, or face) using the Lagrange multipliers (Figure 4.1 c). This process is performed iteratively until a local solution is found.

Finally, the proxy is directly updated to the position without considering its mass, acceleration, or velocity (Figure 4.1 d). The haptic contact force can then be computed from the vector from the HIP to the proxy.

This dynamics of the proxy is also called Zero-order Dynamics (ZoD), which used to describe a dynamic object controlled directly through the manipulation of the position, instead of integration from velocity, acceleration, or force.

4.1.2 Oriented Particles

Oriented Particles (OP) uses particles with an orientation to simulate object deformation. It is based on the shape-matching method (SMM) [Mül+05]. SMM will be briefly introduced next.

For a deformable object, the way helps the object to return to its original shape when it deforms has to be considered. Mass-spring model and FEM compute the stress to achieve it. Unlike them, the SMM computes the optimal rotation and translation of a rigid frame (or "ghost") of the object original shape according to the current configuration. Then, a factor α ranged from 0 to 1, of describing the degree of the matching from the object current shape to the rigid "ghost", is used as the object stiffness. If α is 0, the object receives no force and behaves like sand. If α is 1, the object is rigid since it returns to its original shape by

matching the rigid "ghost" perfectly. A deformable object is, then, represented by setting α between 0 and 1.

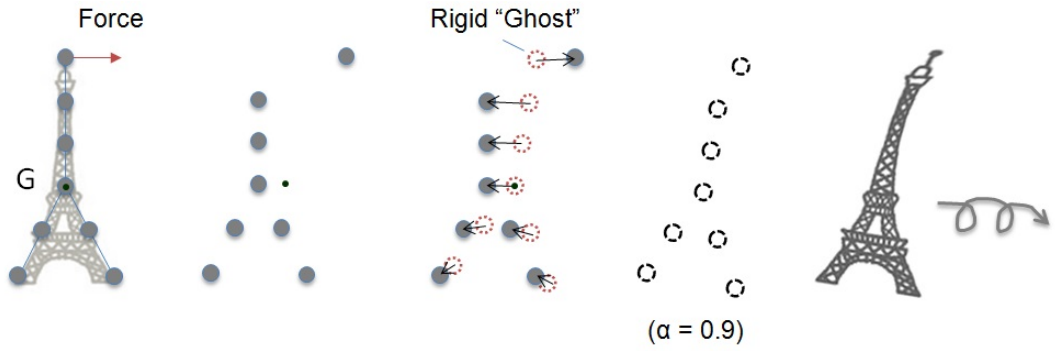

FIGURE 4.3: Deformation Computation of Shape-matching

An example is shown in Figure 4.3. When an object is deformed by a force, the new center of mass G is recomputed. This displacement is the translation of the rigid "ghost". The optimal rotation is calculated as the orientation of the rigid "ghost" when the distance of each pair of corresponding vertices is the shortest. After that, each vertex is matched to the corresponding goal position, i.e., the rigid "ghost", according to the stiffness factor α (0.9 in this example).

The OP utilizes a particle to represent a sub-set of the object vertices. The object is, then, consist of multiple particles. The deformation is performed by executing SMM between those particles according to the group construction of OP. The orientation information is attached to each particle to formulate a full rank moment matrix. This lets the designer be able to use less structure unit to compute deformation compared to SMM. In addition, It also can be used to adjust the balance between accuracy and efficiency flexibly.

4.1.3 Apply Haptics to Oriented Particles

OP method skins the triangular meshes using linear blend skinning. Therefore, the haptic contact force, i.e., the force computed using the virtual proxy method, has to be transferred to OP particles. Firstly, all vertices related to the contact, such as the contact vertex, the two vertices of the contact edge, and the three vertices of the contact triangle are collected. After that, a distributed point-like contact force is computed according to the weight of the distance from the proxy. This is similar to *scalar pressure mask* [JP00]. Using these weights and the blend-skinning weights, the force applied to the corresponding particles can be computed via the following equation:

$$\mathbf{f}_{o,i} = k_{haptic} \cdot w_j^{mask} \cdot w_{j,i}^{blend} \cdot \mathbf{F}, \tag{4.1}$$

where $\mathbf{f}_{o,i}$ refers to the haptic contact force applied to particle i. The term k_{haptic} denotes the spring of virtual coupling [CSB95] to adjust the stiffness, while w_j^{mask} stands for the scalar pressure mask weight for vertex j, and $w_{j,i}^{blend}$ is the blend-skinning weight of particle i for vertex j. An example is shown in Figure 4.4. The proxy contact force $\mathbf{F}$ is distributed to contact related vertices as $\mathbf{f}_{v1}, \mathbf{f}_{v2}$ and $\mathbf{f}_{v3}$, where vertices 1 and 2 belong to particle 1 on the left and vertex 3 belong to particle 2 on the right. According to Equation 4.1, the OP particles accumulate all forces applied on their belong vertices as $\mathbf{f}_{o1}$ and $\mathbf{f}_{o2}$.

Also, for the model has tiny meshes, it is recommended to use a larger range of related vertices around the proxy contact to distribute the force in case of large input applied on a small spot causing instability problems.

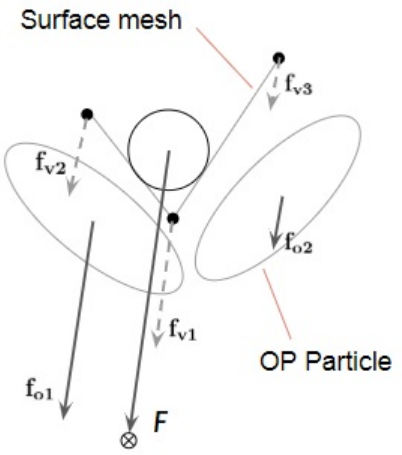

FIGURE 4.4: Computing Proxy Forces Applied to Oriented Particles

4.1.4 Robust Collision Detection for Proxy

Haptic proxy computes collision detection with triangular meshes. Considering the numerical error, such as the rounding error, collision detection may fail to find the contact in extreme cases, such as sharp caves. A typical solution is the tolerance [BFA02] λ which is a small positive value used to keep objects always away from each other. However, the tolerance will produce visual artifacts, i.e. the two objects cannot visually touch each other. Here, a special "tolerance" is proposed for the haptic proxy named Proxy Shrink. Proxy Shrink utilizes a relatively smaller proxy sphere for collision detection, but, when updating the proxy final position, the radius is used as the original one. Using Proxy Shrink, the collision failure caused by the error can be avoided and the proxy is visually stick on the surface mesh without visual artifacts. Typically in this study, a proxy with its shrunk radius:$r_{new} = r_{origin} * j_{shrink}$ is used while the shrink ratio j_{shrink} is 0.999.

A limitation of Proxy Shrink is the reduction of the proxy sphere collision accuracy. For example, it may fall into a hole that is the same size as the sphere. Also, when interacting in extreme cases, the shrink ratio may have to be reduced to handle the increasing numerical error.

4.2 Haptic Proxy with Continuous Collision Detection

This section presents the proposal to solve the "tunneling" problem for the haptic proxy. A clear definition of the problem is presented after the introduction of the background. After that, the solution to this problem is proposed.

4.2.1 Problem Description

The "Tunneling" Problem for Haptic Proxy

As mentioned in 2.3, the "tunneling" problem refers to the failure of DCD. As for the virtual proxy, the CCD is only performed for the moving proxy against static objects. The problem happen when the object moves towards the proxy (Figure 4.5 a) and goes through the proxy sphere without intersection in the current time-step (Figure 4.5 b). The proxy will find no collisions along the path to the HIP. Thus, the proxy will be directly updated to the HIP's position resulting no force feedback (Figure 4.5 c).

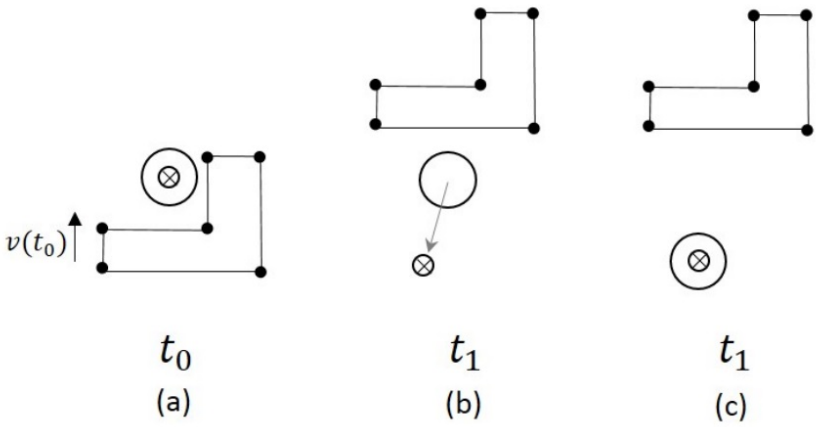

FIGURE 4.5: The "Tunneling" Problem of Virtual Proxy

To solve this problem, a CCD method which covers the moving object against the proxy has to be utilized.

Continuous Collision Detection For Deformable Objects

To find the contact between the deformable moving object and the proxy, a proper CCD method for deformable objects must be used. A fundamental CCD method for deformable triangular mesh model is proposed by Provot [Pro97], in which the contact is handled between triangles, i.e., triangle-triangle contact. Generally, the motion of the triangle is treated linearly due to the computation complexity [Pro97; Tan+09].

In [Pro97], Provot solves the CCD between a point and a triangle using the coplanar condition in the following manner: Let $\mathbf{e}(t)$, $\mathbf{a}(t)$, $\mathbf{b}(t)$, and $\mathbf{c}(t)$ be the point and the three vertices of a triangle, respectively. The variable with a bold font in this thesis refers to a 3x1 vector. During the period from t_0 to t_1, since the contact occurs when the point is coplanar to the triangle, the contact moment t_μ can be found by solving the coplanar condition:

$$(\mathbf{e}(t_\mu) - \mathbf{a}(t_\mu)) \cdot \mathbf{n}(t_\mu) = 0, \tag{4.2}$$

where $\mathbf{n}(t_\mu)$ is the triangle normal calculated by the following equation:

$$\mathbf{n}(t_\mu) = (\mathbf{b}(t_\mu) - \mathbf{a}(t_\mu)) \times (\mathbf{c}(t_\mu) - \mathbf{a}(t_\mu)), \tag{4.3}$$

where $\times$ is the cross-product. Note that, the $\mathbf{n}(t_\mu)$ here is not normalized since it is not necessary. The contact moment t_μ can then be solved in Equation 4.2 easily as a solution of a cubic equation. After that, the collision can be identified by checking if $\mathbf{e}(t_\mu)$ is located inside $\triangle\mathbf{abc}(t_\mu)$.

The Problem In Case Of Haptic Proxy

In the case of the proxy, the contact between the moving triangle and the proxy sphere, i.e., triangle-proxy contact, has to be found. The proxy $\mathbf{p}(t_\mu)$ is represented as a sphere with radius r_{proxy}. The problem can be formulated following the same principle as Equation 4.2 like:

$$(\mathbf{p}(t_\mu) - \mathbf{a}(t_\mu)) \times \frac{\mathbf{n}(t_\mu)}{|\mathbf{n}(t_\mu)|} = r_{proxy}. \tag{4.4}$$

However, to solve this equation, the triangle normal must be normalized as $\frac{\mathbf{n}(t_\mu)}{|\mathbf{n}(t_\mu)|}$. The normalization introduces a square root of a quadratic polynomial in t; hence, the equation does not have an algebraic solution and it cannot be solved with a direct method.

4.2.2 Triangle-Proxy Continuous Collision Detection

The Idea of Solving The "Tunneling" Problem

Since the proxy uses ZoD to updates its position and it has no mass, its dynamics can be considered independently from other objects. Therefore, instead of solving the CCD between the moving proxy against moving object, the idea is to sub-divide this problem into two: moving object against static proxy and static object against moving proxy. First, find the secure proxy position by performing CCD between the moving object and the static proxy as a secured start. Second, performing the traditional proxy method to find the contact of the moving proxy and the static object.

The secure proxy position is where all the potential objects popping the proxy out of the contact. If the traditional proxy method start from this position, all the collisions can be found. As illustrated in Figure 4.6, if a object is moving towards the proxy with a fast speed (Figure 4.6 a), a CCD between the moving object and the static proxy from t_0 to t_1 is performed to find the contact between them in t_μ (Figure 4.6 b). Then, the proxy is transported to the secure position as the start of the traditional proxy method above the moved object (Figure 4.6 c). After that, the traditional proxy method is performed and the contact can be found to update the proxy to the right place (Figure 4.6 d). Here, the term t_μ indicates the contact moment between the moving triangle and the static proxy center. More details will be explained later.

4.2.3 Solving The Continuous Collision Detection Problem for Haptic Proxy

As mentioned in Section 4.2.1, the CCD between the triangle and the proxy can not be easily solved using a direct method. Instead of using iterative methods to solve the equation, which is generally computationally intensive, this thesis proposes an alternative and easier way to solve it.

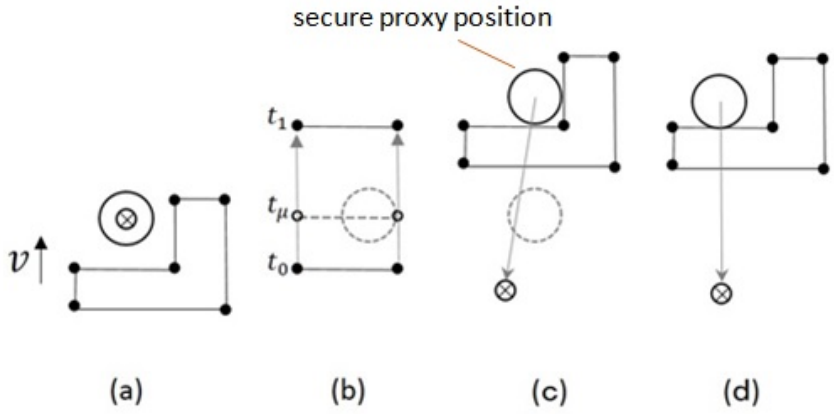

FIGURE 4.6: Solving the "tunneling" problem by securing the start of the proxy update.

Problem Definition

According to the background, the problem is defined as given:

- **Virtual Object:**
  - **The virtual object is modelled by closed triangular meshes.**
- **Object Motion:**
  - **object moves from the initial position at t_0 to the final position at t_1, and the motion of each object's vertex is linear.**
- **Proxy Initial Position:**
  - **The proxy initial position of each time-step is a location at which the proxy has no intersection against other objects,**

find a secure proxy position as the result of the sweeping done by the moving objects whose swept volume overlaps the proxy sphere, and the proxy has no intersection at the secure position.

To solve this problem, firstly, the situation of the contact is identified using the following equation:

$$(\mathbf{p} - \mathbf{a}(t_\mu)) \times \mathbf{n}(t_\mu) = 0. \tag{4.5}$$

This equation solves the coplanar condition with the proxy center instead of the proxy sphere, therefore, the normalization of $\mathbf{n}$ is no longer required. After that, t_μ is used to help separate different situations:

- Situation (i): when $t_0 < t_\mu < t_1$ and $t_\mu \in \Re$, the proxy center $\mathbf{p}$ is coplanar with the triangle at t_μ during the displacement of the triangle;
- Situation (ii): when $t_1 \leq t_\mu$ and $t_\mu \in \Re$, the proxy center $\mathbf{p}$ is coplanar with the triangle at t_μ, which is located on the extension of the displacement of the triangle;
- Situation (iii): other t_μ values.

Since the proxy initial position has been already defined in the place where there is no intersections, the case when $t_\mu < t_0$ will not exist.

In Situation (i), if proxy $\mathbf{p}$ lies inside the face of $\triangle\mathbf{abc}(t_\mu)$ (indicated as $\mathbf{p}_1$ in Figure 4.7), contact occurs on the triangle face. If the proxy is located outside of the coplanar triangle,

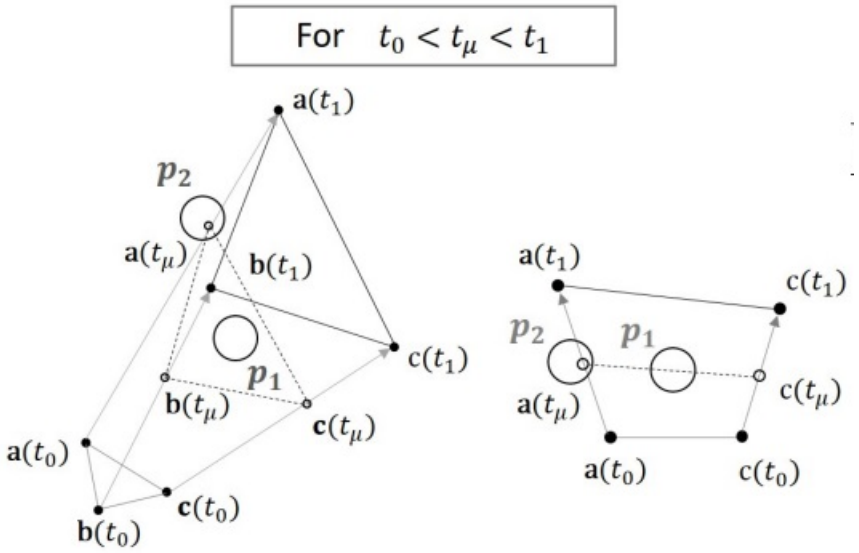

FIGURE 4.7: Situation (i) of Triangle-Proxy Continuous Collision Detection. When the contact is found at t_μ, while $t_0 < t_\mu < t_1$, the coplanar proxy location with potential contact are $\mathbf{p}_1$ and $\mathbf{p}_2$ (left). The side view of the same situation as the left, in which the vertex $\mathbf{b}$ is omitted (right)

but the distance to the triangle is still smaller than the proxy radius (indicated as $\mathbf{p}_2$ in Figure 4.7), contact occurs on the triangle's edge or vertex. In Situation (ii), the proxy collides with the triangle if the distance from $\mathbf{p}$ to $\triangle\mathbf{abc}(t_1)$ is smaller than r_{proxy} (see Figure 4.8).

Two methods, Triangle-proxy CCD (Algorithm 1) and Proxy Pop-out (Algorithm 2), are proposed to handle the contact occurred in those situations.

Solving The Problem According to The Situations

Since multiple triangles may go through the proxy in one time-step, the first triangle which collides with the proxy has to be found. In Situation (i), two kinds of contact triangles can be observed in Figure 4.7: the contact-on-face triangles (collided with $\mathbf{p}_1$ in Figure 4.7) and the contact-on-edge-or-vertex triangles (collided with $\mathbf{p}_2$ in Figure 4.8).

As described in Algorithm 1, firstly, all the triangles are looped to find the first contact-on-face triangle $\triangle\mathbf{abc}_i$ and the first contact-on-edge-or-vertex triangle $\triangle\mathbf{abc}_j$, where i and j represent the corresponding triangle's indices, respectively. Here, $\triangle\mathbf{abc}_i$ is found by solving:

$$if\ S \neq \varnothing, \tag{4.6}$$

$$i = \arg\min_{i \in S} F(\triangle\mathbf{abc}_i), where \tag{4.7}$$

$$S = \{i | t_\mu = F(\triangle\mathbf{abc}_i), \mathbf{p} \in interior\ of\ \triangle\mathbf{abc}_i(t_\mu)\}. \tag{4.8}$$

The function $F(\triangle\mathbf{abc}_i)$ solves Equation 4.5 and returns the smallest value of t_μ satisfying $t_0 < t_\mu < t_1$. This step is necessary because Equation 4.5 may have multiple solutions, and only the smallest one is needed between t_0 and t_1. Similarly, $\triangle\mathbf{abc}_j$ is found by solving:

$$if\ W \neq \varnothing, \tag{4.9}$$

$$j = \arg\min_{j \in W} F(\triangle\mathbf{abc}_j), where \tag{4.10}$$

$$W = \{j | t_\mu = F(\triangle\mathbf{abc}_j),\ dist_{p\_to\_\triangle_j} < r_{proxy}\}. \tag{4.11}$$

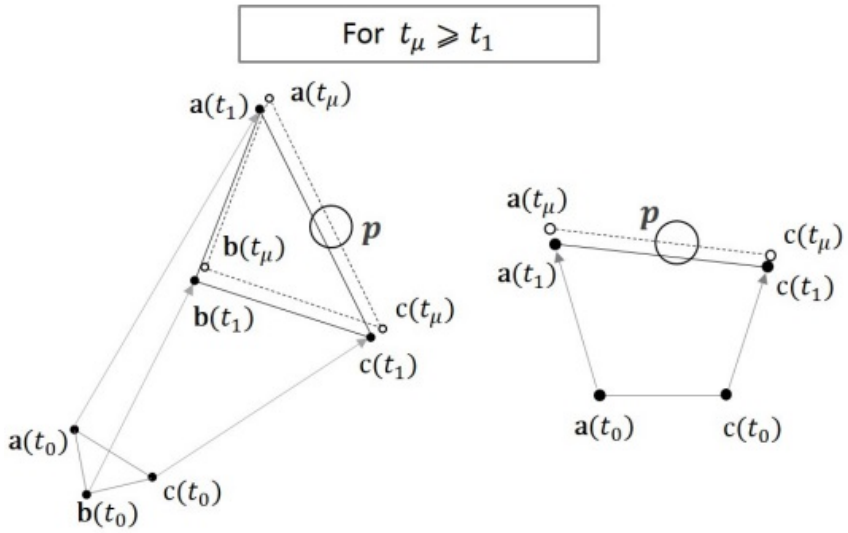

FIGURE 4.8: Situation (ii) in Triangle-Proxy Continuous Collision Detection. While the contact is found at t_μ, which is equal to or over t_1, the proxy may still collide with the triangle on the proxy sphere (left). The side view of the same situation from the left, in which the vertex **b** is omitted (right)

Here, $dist_{p\_to\_\triangle_j}$ and $\mathbf{n}_{p\_to\_\triangle_j}$ represent the closest distance and direction from the proxy to the closest point on $\triangle\mathbf{abc}_j(t_\mu)$, respectively.

Next is to find the secure proxy position and update the proxy the according to the collision. This position is calculated by using the following rule: if a contact-on-face triangle is found, the secure proxy position is above the triangle at t_1, i.e., $\triangle\mathbf{abc}(t_1)$ by the barycentric weights $w_a(t_\mu), w_b(t_\mu)$ and $w_c(t_\mu)$ of $\triangle\mathbf{abc}(t_1)$ (Algorithm 1, Line 4);otherwise, if only a contac-on-edge-or-vertex triangle is identified, the secure proxy is calculated based on the penetration of the proxy (Algorithm 1, Line 6). The notation $\mathbf{n}_i(t_1)$ indicates the normal of $\triangle\mathbf{abc}(t_1)$. The procedure is illustrated in Figure 4.9.

Algorithm 1: Triangle-Proxy Continuous Collision Detection

1 Find i by solving $argmin_{i\in S}F(\triangle\mathbf{abc})$;
2 Find j by solving $argmin_{j\in W}F(\triangle\mathbf{abc})$;
3 **if** *i exists then* **then**
4 $\quad\mathbf{p} \leftarrow w_a(t_\mu)\mathbf{a}_i(t_1) + w_b(t_\mu)\mathbf{b}_i(t_1) + w_c(t_\mu)\mathbf{c}_i(t_1) + r_{proxy}\mathbf{n}_i(t_1)$;
5 **end**
6 **else if** *j exists* **then**
7 $\quad\mathbf{p} \leftarrow \mathbf{p} + (r_{proxy} - dist_{p\_to\_\triangle_j})\mathbf{n}_{p\_to\_\triangle_j}$;
8 **end**

For Situation (ii), performing an additional DCD can identify and solve potential collisions. However, considering the deformation of other triangles nearby, the problem can not simply be solved by a one-time DCD. Therefore, a complementary process called Proxy Pop-Out is utilized, which will be introduced next.

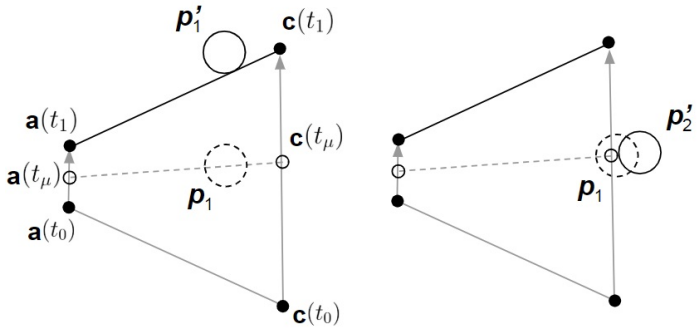

FIGURE 4.9: The Side View of Collision Handling of Situation (i). The collided proxy is resolved in the contact-on-face case (left) as $\mathbf{p}_1$, and contact-on-edge-or-vertex case (right) as $\mathbf{p}_2$. The vertex **b** is omitted.

4.2.4 Towards Multiple Triangular Meshes

The "Tunneling" Problem with Multiple Triangular Meshes

Using triangle-proxy CCD, the "tunneling" problem can be avoided for one moving triangle against the proxy. Nevertheless, considering multiple triangles with deformation, the "tunneling" problem may still happen. The reason is also that the traditional proxy collision is solved with C-Obstacles mentioned in Section 4.1.

An example of a 2-dimensional version of the problem is displayed in Figure 4.10. The edge **ba** and edge **ac** represent the object surface, while vertices **b** and **c** are fixed. The HIP moves left, pushing the vertex **a** and causing it to deform. In the next time-step, the Triangle-proxy CCD is performed to find the new proxy position $\mathbf{p}'$ according to the contact with the moving edge **ac** (Figure 4.10). However, the new proxy position $\mathbf{p}'$ penetrates edge $\mathbf{a}(t_1)\mathbf{b}(t_1)$ at t_1. If we transfer the situation from mesh to C-Obstacles, as shown in Figure 4.10 c, $\mathbf{p}'$ is located below $\mathbf{a}(t_1)\mathbf{b}(t_1)$'s C-Obstacles (Figure 4.10). This makes the traditional proxy collision detection fail to find the contact with $\mathbf{a}(t_1)\mathbf{b}(t_1)$ and go through the object surface since the collision detection is performed on the line segment between the proxy center and the C-Obstacles. According to the definition of the problem, this situation also violate the definition of secure proxy position, at which the proxy should has no intersections.

To solve this problem, formulating the local geometric configuration is computationally intensive and sometimes impossible. Therefore, this study proposes a simple iterative method called Proxy Pop-Out to find a secure proxy position gradually.

Proxy Pop-Out

The Proxy Pop-Out pops the proxy iteratively out of the penetrating mesh until it reaches the secure proxy position. This method is capable and straightforward to solve complicated mesh configurations. The detail is provided in Algorithm 1.

In Algorithm 2, the list of the triangles which are closer to the proxy than *detection_range* is created as *in_range_triangle_list* (Line 1 to Line 5 of Algorithm 2) to reduce the computation cost of each iteration. This range indicates the distances from the triangles to the proxy. Next, for each triangle on the list, if the proxy penetrates that triangle, $dist_{pop}$ and $\mathbf{n}_{pop}$ is calculated. Those variables are the distance and direction from the proxy to the closest point of this triangle, respectively (Line 8 to Line 10 of Algorithm 2). After that, a new proxy position is calculated as the popping destination using the penetration information (Line 11 of Algorithm 2). This process is iteratively executed until no penetration is detected, or the number of iterations exceeds the *iteration_limit* defined by the user (Line 14 of Algorithm 2). After the iteration ends, the proxy is updated to the Proxy Pop-Out result $\mathbf{p}'$.

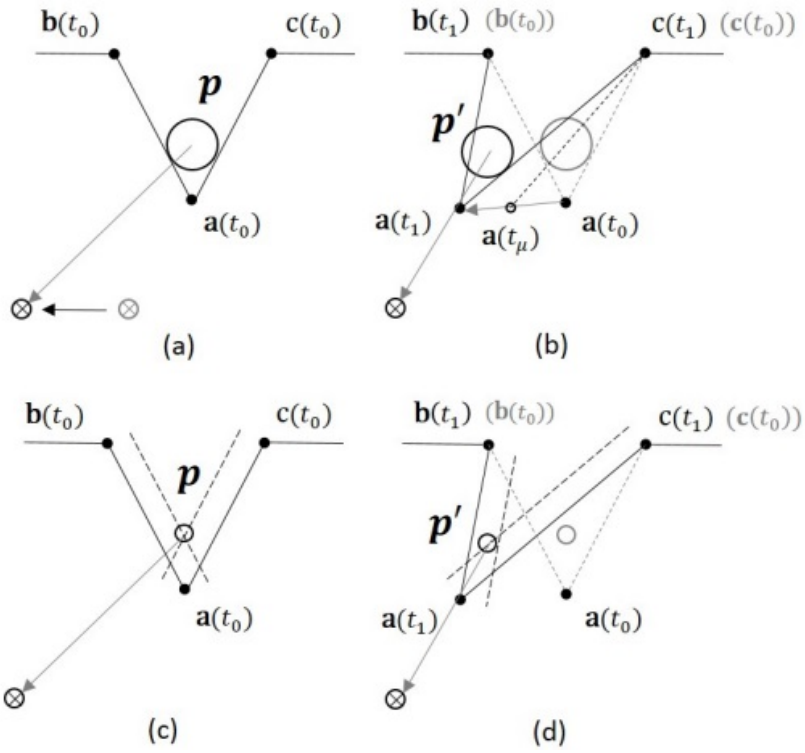

FIGURE 4.10: The Side View of Collision Handling of Situation (i). The collided proxy is resolved in the contact-on-face case (left) as $\mathbf{p}_1$, and contact-on-edge-or-vertex case (right) as $\mathbf{p}_2$. The vertex **b** is omitted.

Iterations are necessary because a single proxy pop-out operation may induce another penetration. In addition, because the pop distance will decrease as the iterations proceed, replacing $dist_{pop}$ with a minimum pop distance when necessary can help boost the progress. More iterations are needed when the proxy lies in a narrow space with large penetrations. Fortunately, the deformation in one time-step is small and induces only small penetrations.

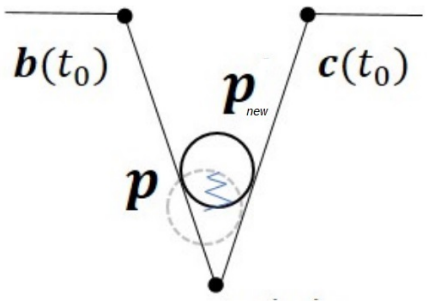

FIGURE 4.11: Proxy Pop-Out Process. When the proxy penetrates edge **ab**, Proxy Pop-Out is able to revise the proxy's position to the secure position $\mathbf{p}_{new}$ following the blue trail.

As an example of the Proxy Pop-Out process, consider Figure 4.11. Here, when the proxy penetrates **ab**, Proxy Pop-Out is able to revise the proxy position to the secure proxy position $\mathbf{p}_{new}$ following the blue trail. In total, five iterations are performed. Note that, if a small value of *detection_range* is used, the proxy may stop the iterations around the triangles near the end of *detection_range* before it arrives at the secure proxy position.

Algorithm 2: Proxy Pop-Out

```
foreach △abc do
    if p is closer to △abc than detection_range then
        Add △abc to in_range_triangle_list;
    end
end
initialize p' ← p;
repeat
    foreach △abc in in_range_triangle_list do
        if p' penetrates △abc then
            dist_pop, n_pop ← CalculatePenetration(△abc, p');
            p' ← p' + dist_pop · n_pop
        end
    end
until No intersection or Over iteration_limit;
p ← p'
```

Algorithm 2 is similar to "zone of impact" in [Pro97] and "the ”untangling cloth” addressed in [BWK03] and [Ye+17]. The method proposed in this thesis belongs to Global-Intersection Analysis [BWK03], while the intersection curve is on the proxy sphere. Compared to "zone of impact", only the proxy is popped while the meshes remain; therefore, there is no guarantee of convergence when there is no space for the proxy. Since we are dealing with real-time simulation, we cannot iterate the global collision detection process until no new collision occurs. The specific limitation cases will be discussed in Section 4.6.1.

4.3 System Implementation

4.3.1 Multi-rate Haptic Proxy Rendering with Triangular Model

Since multi-deformable object simulation requires a large amount of computation, the simulation of the dynamics can only be run at a low update rate. However, human is sensitive to a broad range of vibrations; therefore, haptic rendering requires a high update rate for fine stability and fidelity. Therefore, a multi-rate system is established. The structure is shown in Figure 4.12. This fundamental structure is similar to the one in [OG07]. Instead of updating the contact Jacobian, this study synchronizes the contact constraints of the C-Obstacle of the primitives as the intermediate representation of the contact. The traditional proxy collision is solved with all meshes, while only the constraint planes (up to three) of the contact are synchronized to the haptic thread. The haptic thread computes proxy collision detection only with those constraint planes, which is simple and efficient, and the feedback force, scaled by the virtual coupling spring, is transferred to the device. The virtual coupling is only used to adjust the magnitude of the feedback. In this manner, the method proposed in this thesis can provide the haptic feedback of the surface shape constraints at a high update rate.

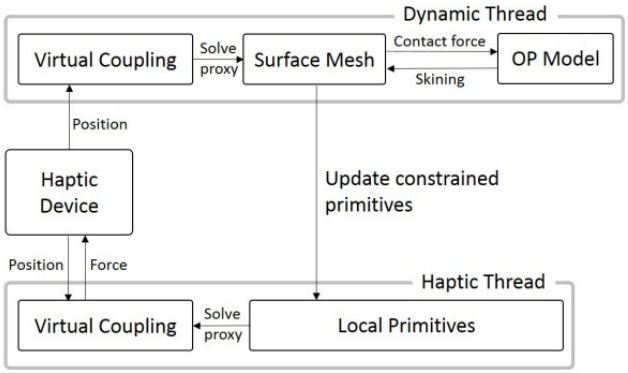

FIGURE 4.12: Multi-Rate Haptic Rendering System Structure

4.3.2 Position-Based Simulation Procedure

The system is based on the PBD simulation. In the initialization, OP particles and edges are created on the mesh model manually. After the main loop start, triangle-proxy CCD and Proxy Pop-Out are performed in *SolveCCD()* and *SolveProxyPopOut()*, respectively. The traditional virtual proxy [RKK97] is solved for the haptic contact force in *SolveVirtualProxy()*. Next, the particle velocities $\mathbf{v}$ are calculated from the time interval Δt, the haptic contact force $\mathbf{f}_{haptic}$ and the particle mass m. Afterward, the predicted particle positions $\mathbf{u}$ are calculated from those velocities. Object-object collision detection is then performed in *EllipsoidCollsionDetections()*. Next, the PBD constraints, such as the cloth constraint and the shape-matching constraint, are calculated. This process is iterated $solverIterationNum$ times following a Gauss-Seidel scheme. The final particle velocities $\mathbf{v}$ and positions $\mathbf{x}$ are updated. Finally, the current mesh vertices are recorded for the CCD for the next time-step and use the

blend skinning method in *BlendSkinning()* to update the vertex position.

Algorithm 3: Simulation Procedure of A 3-DoF Haptic Proxy Simulation

Initialization: Create OP model on the mesh model;
Dynamic Simulation Loop Start:
do p $\leftarrow$ SolveCCD(); //Algorithm 1
do p $\leftarrow$ SolveProxyPopout(); //Algorithm 2
do p $\leftarrow$ SolveVirtualProxy();
foreach *particle i* **do**
    $\mathbf{f}_{o,i} \leftarrow$ CalculateHapticContactForce(); //Equaiton 4.1
end
foreach *particle i* **do**
    $\mathbf{v}_i \leftarrow \mathbf{v}_i + \Delta t \mathbf{f}_{o,i} / m_i$;
end
foreach *particle i* **do**
    $\mathbf{u}_i \leftarrow \mathbf{x}_i + \Delta t \mathbf{v}_i$;
end
EllipsoidCollisionDetections();
define $j = 0$;
repeat
    projectConstraints($C_{op}, \cdots, \mathbf{u}_i, \mathbf{u}_{i+1}, \cdots$);
    $j = j + 1$;
until $j \geq solverIterationNum$;
foreach *particle i* **do**
    $\mathbf{v}_i \leftarrow (\mathbf{u}_i - \mathbf{x}_i) / \Delta t$;
    $\mathbf{x}_i \leftarrow \mathbf{u}_i$;
end
foreach *triangle h* **do**
    Record vertex j;
end
BlendSkinning();
Dynamic Simulation Loop End.

4.4 Evaluations and Results

This section presents the evaluation of the approaches introduced above. All evaluations were executed on an Intel Core i5-4210U (1.7Ghz) CPU. Note that the GPU was not used for the simulation. Furthermore, a Spidar-G6 was used as the haptic device.

4.4.1 Triangle-Proxy CCD Evaluation

To evaluate the triangle-proxy CCD, the experiment was conducted as a simulation of catching a dropping cloth. A double-faced, 1-meter-edge cloth (as the "thinnest" object) consisting of 1089 vertices and 4096 triangles was dropped from a 10-meter height. Both the cloth constraint and the shape-matching constraint were simulated. Each vertex was set as an OP particle and connected it to the nearby eight particles. The proxy radius was set to 0.03 meters, and the simulation time interval was defined as 0.01s. The HIP and the proxy were initialized at the origin. The force feedback is recorded, while the HIP was fixed, and user input was cut off to present the contact force directly without user interruption.

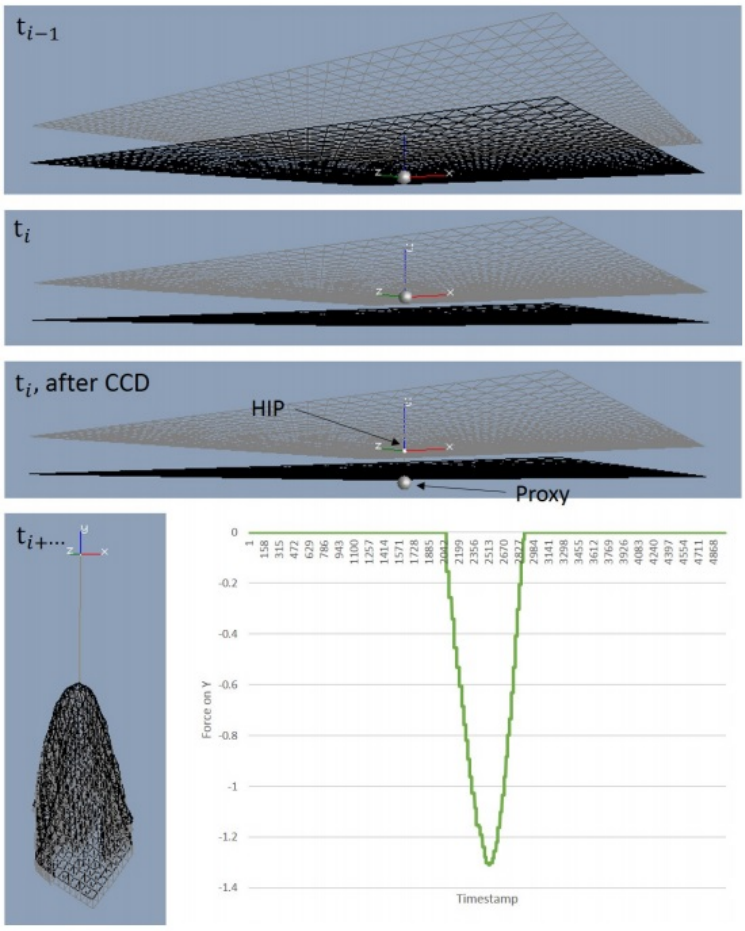

FIGURE 4.13: Evaluation of Catching A Dropping Cloth. The screenshots shown from top to bottom are captured in one time-step before the contact t_{i-1}, during the contact time-step t_i before and after CCD is performed, and few time-steps later. The black and gray meshes represent the model at the current and the previous time-steps, respectively. The big, gray ball indicates the proxy, while the small, white ball represents the HIP. The force feedback graph of the contact is presented, and the HIP is fixed.

Because the dropping cloth went through the proxy at a high speed of about 14 m/s, no intersections occurred in one time-step. Hence, the performance of the triangle-proxy CCD can be investigated (displayed in Figure 4.13). From Figure 4.13, we can conclude that triangle-proxy CCD was able to help the small proxy detect the collision with a thin object going entirely through it. The triangle-proxy CCD calculation in this evaluation took about 0.3 ms to 1 ms. Without using the proposed methods, contact with the proxy cannot be detected, and the user cannot catch the cloth model.

4.4.2 Proxy Pop-Out Evaluation

This evaluation creates a typical situation to investigate the capability of Proxy Pop-Out (see Figure 4.14). The model used was single-faced and consisted of 33 vertices with 32 faces, which can trigger multiple Proxy Pop-out iterations. The particle construction was the same as in Section 6.1. The proxy radius was set to 0.03 meters, and the minimum pop distance was set to 0.002 times the proxy radius. The edge vertices were fixed and left the center vertex free to deform. If the HIP were to push the model center, a narrow space consisting of multiple triangle faces would be created. The evaluation was executed by moving the HIP from left to right at three different depths of the center vertex vertical coordinate at -1, -2, and -3 meters from the origin to see how many pop-out iterations would be needed for large deformations. The HIP's velocity was set to 3m/s. The Proxy Pop-Out iteration counts were recorded and displayed in Figure 4.14. The figure shows that more iterations are required for the narrow space if the deformation is large. The peaks on the line in Figure 4.14 represent the iteration times required for the bouncing of the model.

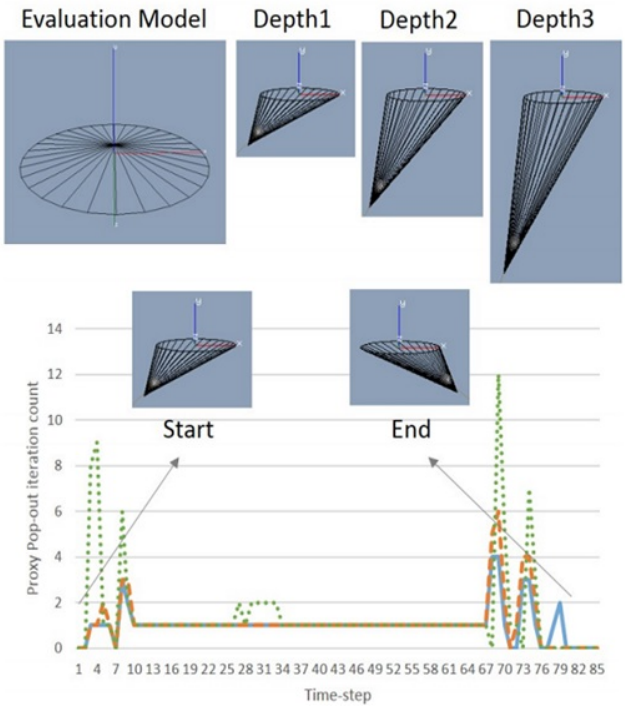

FIGURE 4.14: Evaluation of Proxy Pop-Out. The screenshots on the top indicate the evaluation model and the interactions with three different depths of the HIP vertical coordinate at -1, -2, and -3 meters from the origin. The table on the bottom refers to the Proxy Pop-Out iteration counts during these interactions.

Computation time (Microsecond)		Iteration count		
		10	100	1000
Number of in-range mesh	10	0.01	0.134	0.989
	20	0.018	0.166	1.533
	30	0.028	0.269	2.234

FIGURE 4.15: Efficiency test result of the Proxy Pop-Out algorithm. Searching for in-range triangles is ignored, and all possible situations are considered equally.

The cloth constraint was not included in this simulation with the large deformation. Without the Proxy Pop-Out, the proxy would go through the model whenever an intersection between the proxy and the potential contact constraint occurs. Because the computation time for the Proxy Pop-Out in this evaluation was too short and variable to measure, this thesis presents the following evaluation to assess the algorithm's efficiency.

Figure 4.15 presents the computation time for the Proxy Pop-out based on the number of iteration and triangle meshes inside the detection range of Algorithm 2. The results presented were the average of a ten times repeated test of Algorithm 2 done without in range triangle searches, and all branches were performed equally. Note that if a large model with many meshes is used, searching will take an unignorable amount of time. In that case, other advanced detection algorithms are recommended for this part.

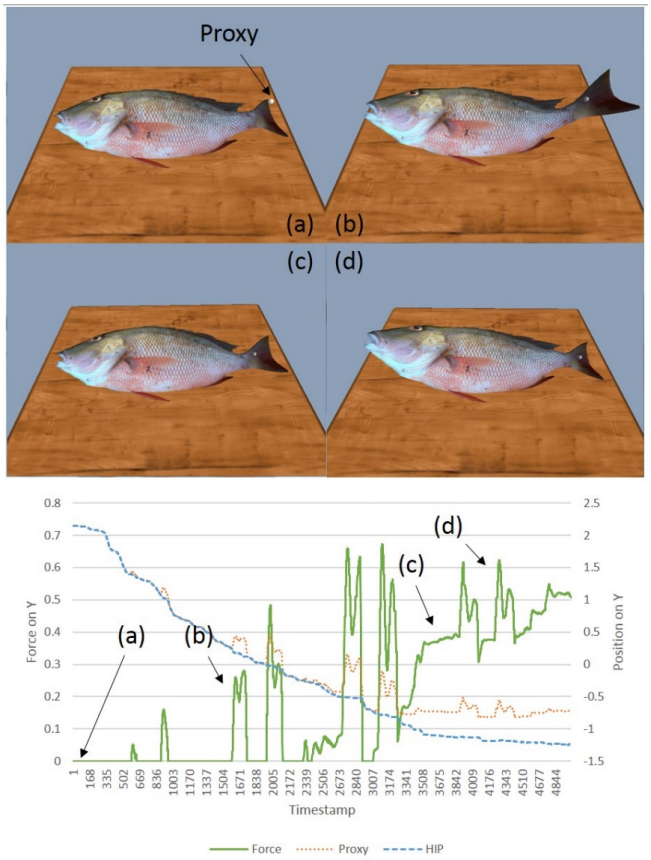

FIGURE 4.16: Haptic Rendering Evaluation of Holding The Fish Tail Simulation. The force and the proxy and HIP positions are recorded on Y coordinates. The four screenshots indicate the corresponding moments of this task.

4.4.3 Haptic Rendering Evaluation

Two scenarios are prepared to evaluate the haptic rendering results of the proposed system. In the first scene, a living fish on a table was simulated. The fish was trying to escape and shaking its head and tail. The user's task was to push its tail and hold it on the table. The force and the positions of the proxy and HIP on Y coordinates were recorded. The results are displayed in 1kHz in Figure 4.16, where each peak of the force indicates one shake of the tail. In the system, the Y-axis represents the vertical direction. For easier visualization of the force chart, the fish was stick to the table by fixing particles around its belly.

The fish model was a free model from the Internet [Fis], which included 1047 vertices. The physics model consisted of 137 OP particles, all of which connected to eight other particles nearby. The particle stiffness of the SMM was set as follows: 0.9 for the head; 0.6 for the bully and tail. The fish was animated by moving the particles on the neck and the upper part of the tail up and down two times repeatedly. The rest of the movements were simulated with OP. The stiffness of the virtual coupling spring was set to 10N/m. No virtual coupling damping was performed. The results indicate that even if the fish shakes its tail dramatically with strong forces, the user can stably touch its tail on the air and hold it on the table.

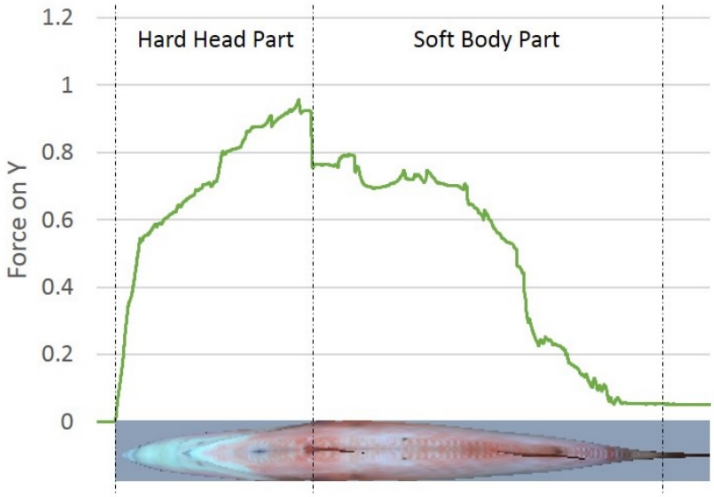

FIGURE 4.17: Haptic Rendering Evaluation. Probing haptic interactions. The upper chart refers to the recorded force on the Y coordinates of sliding the fish model surface while the corresponding fish model is attached below.

The second scene used the same model as the first one. This time the user touched the fish and gently slid down the fish from head to tail. No animation was performed. The force on the Y coordinates was recorded and displayed in Figure 4.17. From the figure, we can see that the fish model's stiffness and shape are demonstrated through the haptic feedback force. The results indicate that the proposed system is able to present probing haptic interactions with a deformable object. Note that the stiffness was adjusted with the stiffness of SMM, which was only plausible compared to the real material behavior. The shape of the force graph in the figure is not the same as the model. Both the quality of the model and the number of OP will affect the result.

4.4.4 Result

Figure 4.18 displays the screenshot and performance of a fully dynamic scene simulation of the proposed system. This scene included four fish models with 4188 vertices and 6800 triangles, together with 548 OP particles. The PBD constraint iteration was performed five times. The result showed that, while there was no contact with the proxy, the simulation ran

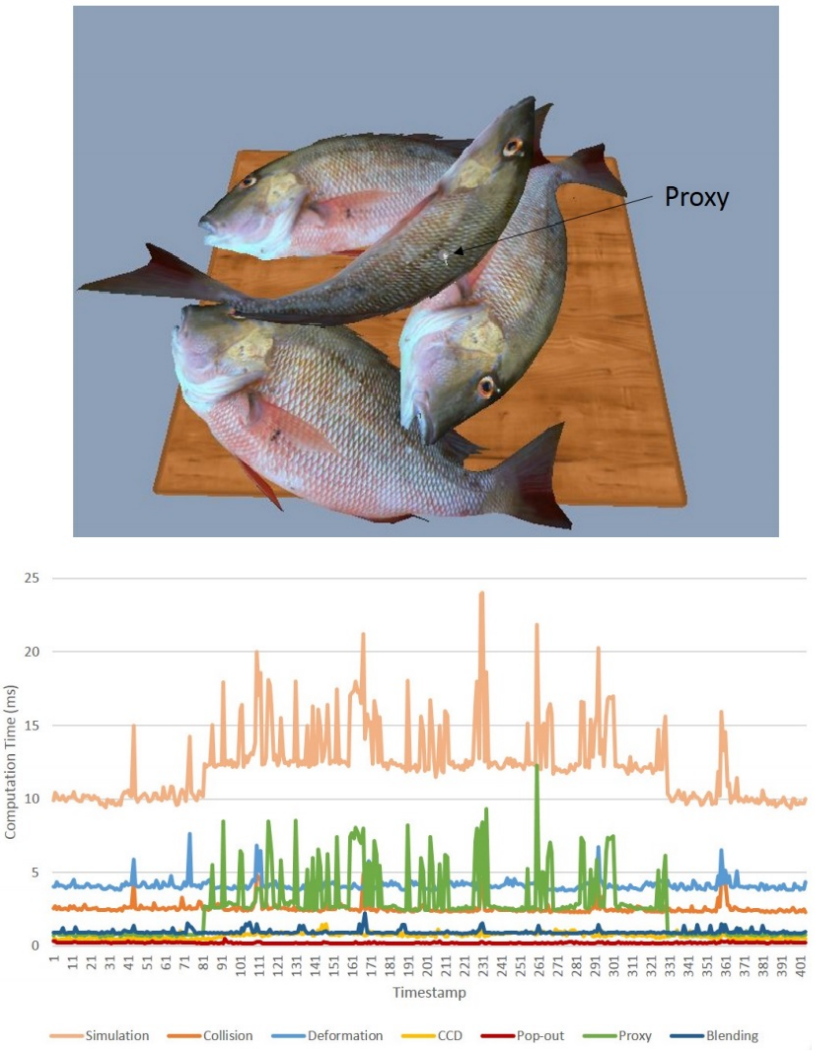

FIGURE 4.18: Performance of The Proposed System. The graph on the bottom describes the computation time of each process in a multi-soft-object simulation with haptics, while the screenshot of that simulation is displayed at the top.

at about 80 frames per second. When the proxy touched the object, it dropped to 55 frames a second. In the graph in Figure 4.18, collisions represented just the computation time for the particle collisions, which took about 2.5 ms. The deformation (shape-matching) and the blend-skinning calculation took 4 ms and 1 ms, respectively. The triangle-proxy CCD took only around 0.5 ms. With less than two pop-out iterations, the Proxy Pop-Out took less than 0.3 ms. The haptic proxy calculation was the most intensive one since it involved an iterative process, and a large number of triangles were used for virtual proxy calculation.

4.5 Summary

In this chapter, a CCD method for 3-DoF point-object haptic rendering of the haptic proxy, called the triangle-proxy CCD, is proposed. Together with a complementary process, Proxy Pop-Out, the ”tunneling” problem that occurs during haptic interactions with dynamic and deformable objects modeled by triangular meshes, can be solved. This approach can handle probing haptic interactions and 1kHz haptic contacts with the surface mesh. The evaluation results indicate that the proposed system is efficient and stable.

4.6 Discussion

In this proposal, our method only focus on triangle mesh; therefore, other kinds of object representations such as point clouds and Spline Models, have to transform their model into triangles to use our methods. Also, the triangle-proxy CCD method can only handle linear triangle motions, which is a common limitation of most deformable object simulation systems with CCD.

4.6.1 Limitations of Proxy Pop-Out

Proxy Pop-Out can move the proxy gradually out of the penetrated triangles. However, since it pops any triangle that closer to the proxy in distance than the proxy radius, it can pop the proxy to the wrong side of a thin object when another triangle clips the proxy. As shown on the left of Figure 4.19, this problem arises when the execution order for the left triangle is earlier than that for the middle one. Moreover, since only the proxy is moved, the proxy can get stuck in cases like triangles facing each other in parallel. The distance between them is less than the diameter of the proxy (Figure 4.19, middle), or the proxy is inside a small ”pocket” of triangles (Figure 4.19, right). As the consequence shown in Figure 4.19, the proxy is popped to the left rather than to the middle in the first case. In the second and third cases, the iteration limit will be reached, and the proxy may be popped by one of the intersected triangles according to the value of iteration maximum. Therefore, we will get unexpected results when using Proxy Pop-Out to deal with these cases.

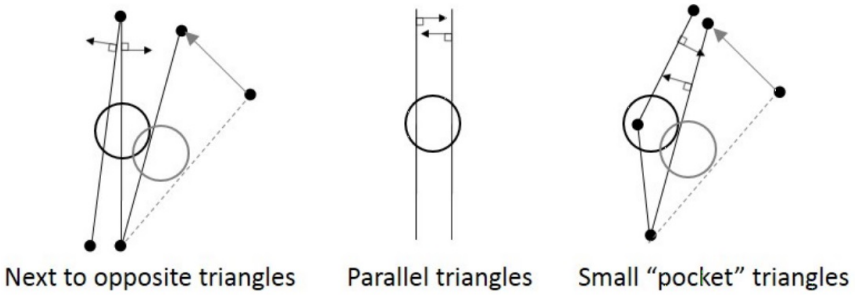

FIGURE 4.19: Proxy Pop-Out Limitations. Special cases in which the Proxy Pop-Out may fail to avoid the "tunneling" problem.

A possible solution to the first case of Figure 4.19, is an additional condition, i.e., the ”positive half-space” condition, which is used to build the in-range triangle list. Using this condition, only the triangles for which the proxy center lies on the half-space of the positive direction of the triangle normal are selected, and the first case can be solved. However, this only guarantees small proxy penetrations (penetration less than the proxy radius). If the penetration is equal or deeper than the proxy radius, ”tunneling” will occur. Fortunately, a triangle’s deformation and motion in one time-step are generally small. Hence, the user can choose whether to use the positive half-space condition or not depending on the case.

For other cases, temporarily reducing the proxy radius to fit the narrow space can be considered as the solution. However, the next question is when and how much to perform this reduction and what the consequences are.

Other common limitations for both two proposed systems are introduced in Section 6.1.1.

Chapter 5

Extension to A Six-Degree-of-Freedom Deformable Haptic Proxy

This chapter will introduce a more general system: a 6-DoF haptic system that employed a deformable haptic proxy via an extension of the method proposed in Chapter 4. This thesis overcomes both the force artifacts problem and the "tunneling" problem, which is supervising compared to previous studies.

5.1 The Force Artifacts Problem

As mentioned before, the virtual proxy uses ZoD for its update. This is an advantage for haptic rendering compared to first or second-order dynamics since they may induce force artifacts, i.e., the feeling of the inertia, and reduce the haptic control transparency. As an example of the difference, when playing table-tennis, a person is able to control and hit the ball directly with the racket holding in his/her hand instead of connecting through a spring (as shown in Figure 5.1).

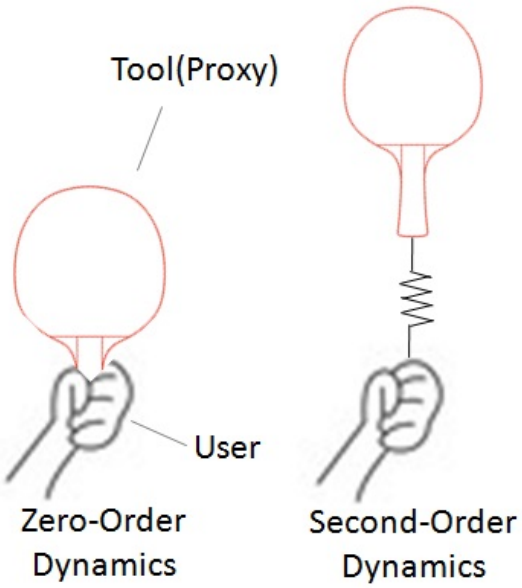

FIGURE 5.1: Comparison Between Zero-Order Dynamics and Second-Order Dynamics. User has a direct control with the tool/proxy instead of connecting through a spring.

However, the traditional virtual proxy method is only for 3-DoF haptic rendering. To overcome this limitation, this study proposes a Multi-Sphere Proxy (MSP) model, which will be introduced in the following section.

5.2 Modeling Haptic Tool with Multiple Proxies

5.2.1 Multi-Sphere Proxy Model

The MSP model uses multiple spheres rather than a single sphere to represent a haptic tool. As for the spheres, this study directly uses the particles of the OP[MC11] model as the MSP model. In an MSP model, two kinds of particles are used: the particle of the haptic tool is called the proxy particle, and the particle representing the haptic device is called the device particle. An example is shown in Figure 5.2.

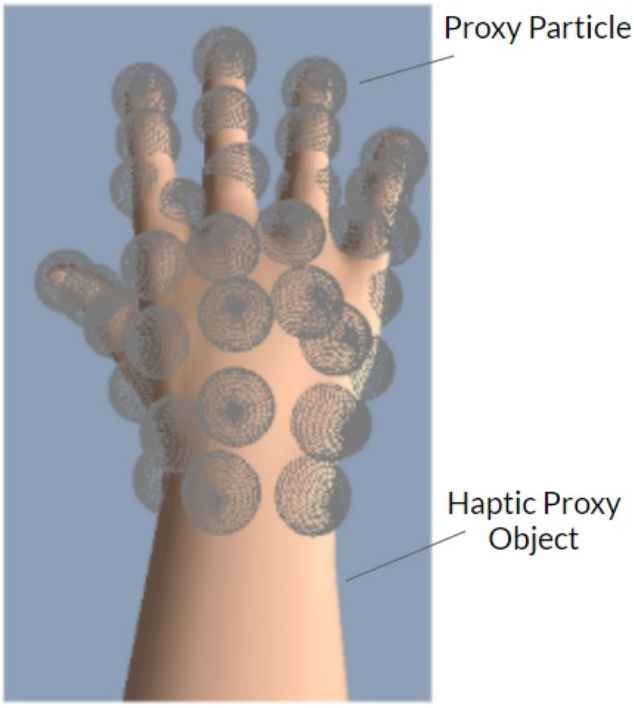

FIGURE 5.2: A Example of Multi-Sphere Proxy Model.

Since the CCD is handled between the sphere proxy and triangular meshes (introduced in Section 4.2.2), spherical particles are selected rather than ellipsoidal ones for better computational performance. However, since using ellipsoidal particle is a choice with more accuracy than using spheres, it will be interesting to conduct further research on CCD of using ellipsoids.

Because the number of proxy particles is related directly to both collision accuracy and simulation speed, collision accuracy, and system efficiency should be balanced. In this thesis, the OP model which is manually created similar to [MC11].

This method is similar to that employed in [Cir+13] and [Wan+15], which also utilize spheres. Nevertheless, their methods cannot handle CCD contact between the sphere and the triangular mesh, and the method in [Cir+13] cannot avoid force artifacts caused by the tool's inertia.

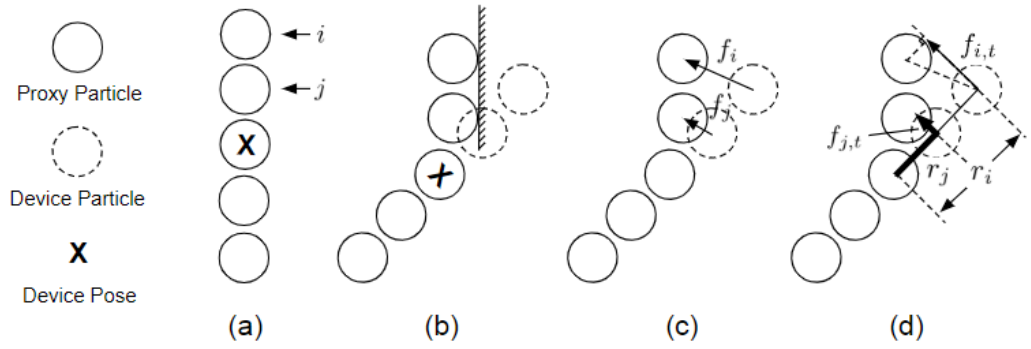

FIGURE 5.3: Force and Torque Calculation of The Multi-Sphere Proxy Model. (a) The start. (b) The deformable proxy tool encounters a contact. (c) The force $\mathbf{f}_i$ and $\mathbf{f}_j$ and, (d) the torque $\mathbf{r}_i \times \mathbf{f}_{i,t}$ and $\mathbf{r}_j \times \mathbf{f}_{j,t}$ of proxy particle i and j, respectively.

5.2.2 Feedback Force and Torque Calculation

The force and torque feedback is calculated from the discrepancy between the haptic proxy tool and the device. As shown in Figure 5.3, the force and torque are calculated based on the discrepancy of each pair of proxy and device particles. Only the collided proxy particles are considered in the calculation. Next, all the forces and the torques are summed, respectively. Last, each of these sums is divided by the total number of proxy particles to get the final feedback. Both translational and rotational springs are used to adjust the feedback magnitude.

5.2.3 Haptic Contact Force Computation

The contact force applied to the virtual object is computed using the same method introduced in Section 4.1.3, which uses a distributed point-like contact force applied as an external force of PBD to the corresponding OP particle of the virtual object which is similar to the procedure introduced in Section 4.3.2.

5.2.4 Haptic Constraint

This study proposes a haptic constraint to apply the haptic device input, i.e., the position and orientation, to the haptic proxy tool. The haptic constraint computes the current position of device particles as the goal position $\mathbf{g}$ by the following equation:

$$C_{haptic} : \mathbf{g}_i = Q_{dev}\mathbf{o}_i + \mathbf{d}_{dev}. \tag{5.1}$$

After the goal position is computed, the traditional proxy method [RKK97] is performed to update the proxy particle position from the current position to the goal position, i.e., the position of device particle (shown in Figure 5.4 (b)). Compared to other position-based constraints, which generally use a stiffness factor α (ranged from 0 to 1) to control their stiffness, the infinite stiffness ($\alpha = 1$) is used in this study. This process helps us to produce a non-delayed update of the tool.

Since a stiff haptic constraint is used in PBD, it cancels the effect of the other constraints. To solve this problem, this study applies the haptic constraint only once before the other constraints. Therefore, the other constraints, e.g., the deformation constraint, can be performed correctly. Besides, all the constraints compute the result only based on the stiff haptic constraint. Therefore, in this study, only the contacts detected by the proxies can trigger deformation and produce force and torque feedback. In this way, the tool is only sensitive to the

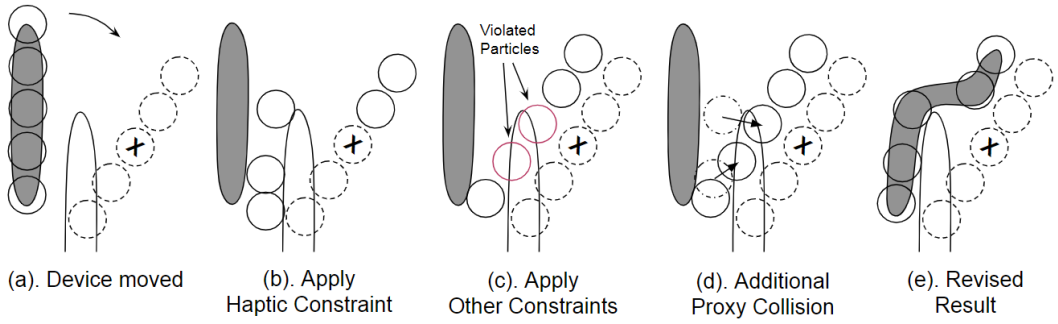

FIGURE 5.4: Simulation of a Zero-Order Dynamic Deformable Haptic Proxy Tool

contacts and ignoring the force of tool inertia caused by the tool dynamics, i.e., the force artifacts.

However, this kind of position-based constraint may induce new "tunneling" problems. This problem will be discussed next.

5.3 Solving The "Tunneling" Problem of Constraint Computation

5.3.1 The Problem of Position-Based Constraints

As mentioned before, an infinite stiff haptic constraint is used and computed ahead of all other constraints. If we perform the constraint calculation, the result may violate the collision constraints and cause the "tunneling" problem (shown in Figure 5.4 (c)). Unfortunately, it is impossible to identify all collision constraints in advance in the traditional proxy collision detection since they have different starts and goals.

5.3.2 The Solution to The Problem

To solve the "tunneling" problem caused by the constraint calculation, the proposal runs an additional half-CCD, which is just the same as the traditional proxy method, for the constraint calculation (shown in Figure 5.4 (d)) to revise the particle positions. To do so, this study recorded the proxy particle positions before and after the PBD constraint calculation. After that, the proxy method is performed from the recorded start to the end. The whole progress is shown in Figure 5.4.

5.4 System Implementation

5.4.1 Multi-Rate Haptic Rendering

A multi-rate architecture is developed to improve interaction stability. First, if we only synchronize all the proxy particle positions from the dynamic thread to the haptic thread, the delay caused by synchronization may induce "dragging" forces. Therefore, the collision information (collision occurs or not) should also be synchronized. This will help us to eliminate the force artifacts when no contact occurs. The feedback force and torque are calculated according to Section 5.2.2 in the haptic thread.

5.4.2 Simulation Procedure

The simulation loop of the dynamic thread is executed as procedures shown in Algorithm 4. In the dynamic thread, the simulation starts with the haptic proxy tool. CCD for each proxy particle is performed as the same as the method introduced in 4.2.2 and 4.2.4. Next, the device configuration, i.e., position and orientation, are retrieved and used to compute the goal position of haptic constraint C_{haptic}. After that, an additional traditional proxy collision detection is performed from the proxy's current position $\mathbf{p}$ to the goal position $\mathbf{g}$. Then, this position is recorded as $\mathbf{p}_{start}$. After the computation of the position-based constraints, e.g., deformation constraint, an additional traditional proxy collision detection is performed from $\mathbf{p}_{start}$ to $\mathbf{p}$ to revise the proxy position. Finally, the haptic contact forces are calculated, and the other virtual objects are simulated with these forces applied.

Algorithm 4: Simulation Procedure of a 6-DoF Haptic Proxy Simulation

```
1  Initialization: Create OP model and MSP model on the mesh model;
2  Dynamic thread start:
3  Simulate the haptic proxy tool:
4  foreach proxy particle i do
5      do p_i ← SolveCCD(); (Section 4.2.2)
6      do p_i ← SolveProxyPopOut(); (Section 4.2.4)
7      do g_i ← C_haptic; (Equation 5.1)
8      do p_i ← SolveVirtualProxy() from p_i to g_i ;
9  end
10 foreach proxy particle i do
11     do p_i ← p_start;
12 end
13 repeat
14     projectConstraints(C_op, ··· , p_i, p_{i+1}, ··· );
15     j = j + 1;
16 until j ≥ solverIterationsNum;
17 foreach proxy particle i do
18     do p_i ← SolveVirtualProxy() from p_start to p_i;(Section 5.3.2)
19 end
20 CalculateHapticContactForces(); (Section 5.2.3)
21 Simulate the other virtual objects.
```

Even though PBD is used for the simulation, the mass and velocity of the haptic proxy are no longer integrated. This helps us to create a haptic proxy with ZoD that eliminates the force artifacts of the tool inertia.

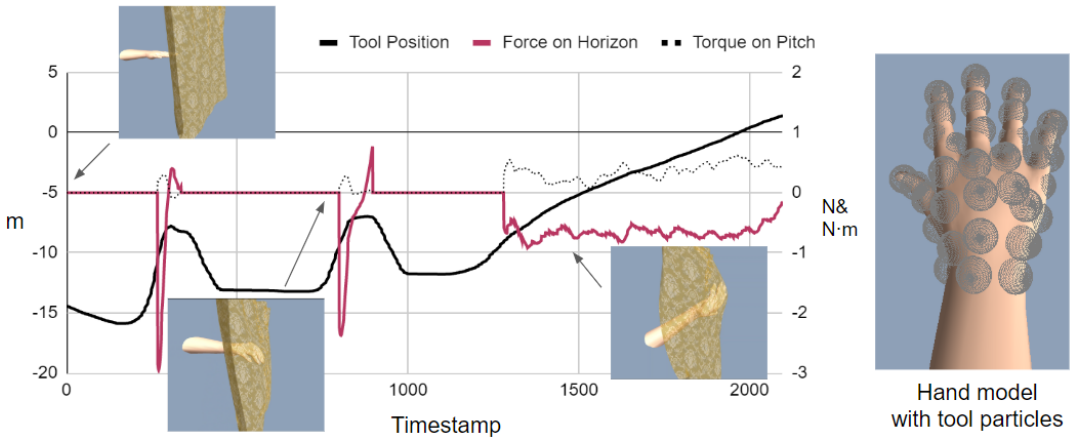

FIGURE 5.5: Evaluation of Interactions. Left: the recorded tool position, feedback force, and torque are presented with screenshots of a deformable hand interacting with a yellow curtain. Right: the hand model and its proxy particles with the proxy radius.

5.5 Evaluations and Result

In this section, two evaluations of the proposed method are introduced. The evaluations were performed on an Intel i5-7300 4-core CPU PC without GPU. A Spidar-G6 was used as the haptic device.

5.5.1 The Evaluation of Interactions and Efficiency

In this evaluation, a deformable 1kg hand model [got18] was used as the tool to interact with a double-faced, 8-meter, 200g curtain model. The curtain model contained 1089 vertices and 4096 triangles, while each vertex was modeled with one OP particle and connected with another eight nearby particles. The hand model had 3919 vertices and 3906 triangles with 37 OP particles (the placement of the proxy particles can be further optimized). Only SMM and distance constraints were simulated for the two models. The PBD iteration counts were two and four for the curtain and the hand models, respectively. The feedback springs were both set to 20N/m.

The result is displayed in Figure 5.5. A series of interactions were performed, including tapping and pressing, on the curtain model. From Figure 5.5, we can see that the tool moved freely in the air without the force caused by the inertia. The system was stable even fast motions were inputted with strong impacts while the tool and the device deformation were performed correctly. The whole simulation took 2.5 seconds. The data was collected from the haptic thread, which was running at 1kHz, while one time-stamp indicated one loop of the calculation. The simulation ran at an average rate from 45 to 59 FPS, depended on the collision number. The calculation of the PBD constraints took 4 ms. The CCD, including triangle-proxy CCD and Proxy Pop-out, required 3 ms, while the two runs of the proxy method required 8 to 12 ms depending on the collision count. Only a basic pruning method, which ignored far-off meshes (at least 1 meter from every proxy particle), was used.

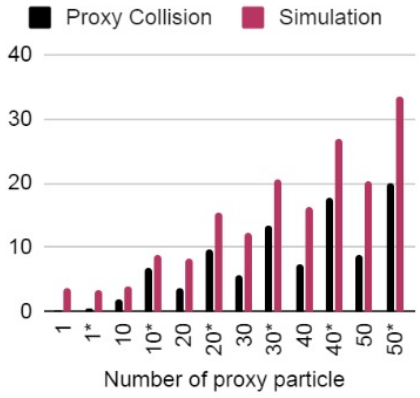

FIGURE 5.6: Evaluation of Efficiency. The efficiency evaluation utilized the average computation time for the simulation and proxy method using 1 to 50 particle particles. The label with "*" indicates collisions have occurred in the simulation while others are not.

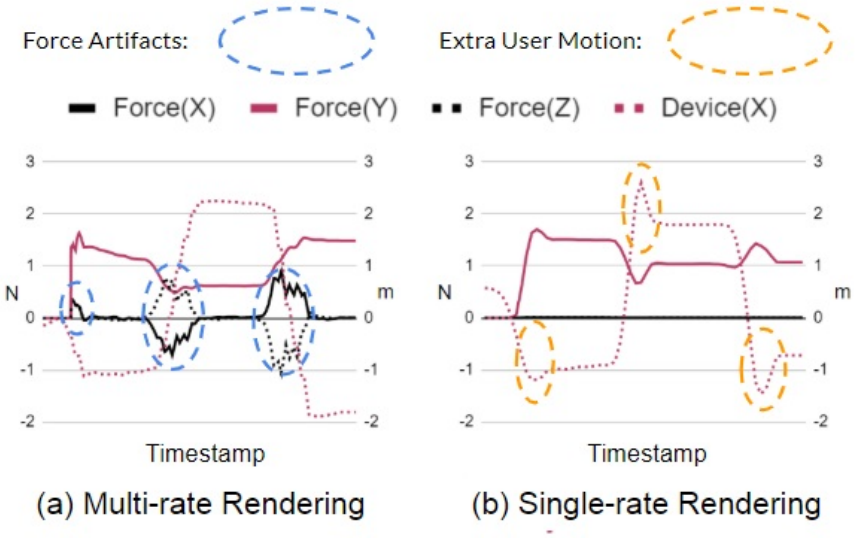

FIGURE 5.7: Comparison Between Multi-Rate (a) and Single-Rate (b) Haptic Rendering. The multi-rate approach shows "dragging" force artifacts while the contact occurred and the user started to move the tool. On the contrary, the single-rate approach is able to eliminate this force artifacts, however, the interaction stability was reduced, which can be indicated by the extra user motions compared to the multi-rate one.

The computational efficiency was highly related to the number of proxy particles; therefore, a test of this scene using proxy particles numbering from 1 to 50 was performed (illustrated in Figure 5.6).

5.5.2 The Evaluation of Multi-rate Haptic Rendering

A comparison was conducted between multi-rate and single-rate haptic rendering to evaluate the force artifacts caused by multi-rate simulation. In this evaluation, the device slid along a virtual horizontal plane using the same hand model of the first evaluation. The results in Figure 5.7 show that the horizontal "dragging" forces (forces on X and Z axis) arise only in the multi-rate case as the device moves. The single-rate case presents no force artifacts, while extra user motion shows a drop in the tool's controllability as a result of the decrease

of system stability. During our tests, the magnitude of the force artifacts was proportional to the computation time of the physics simulation.

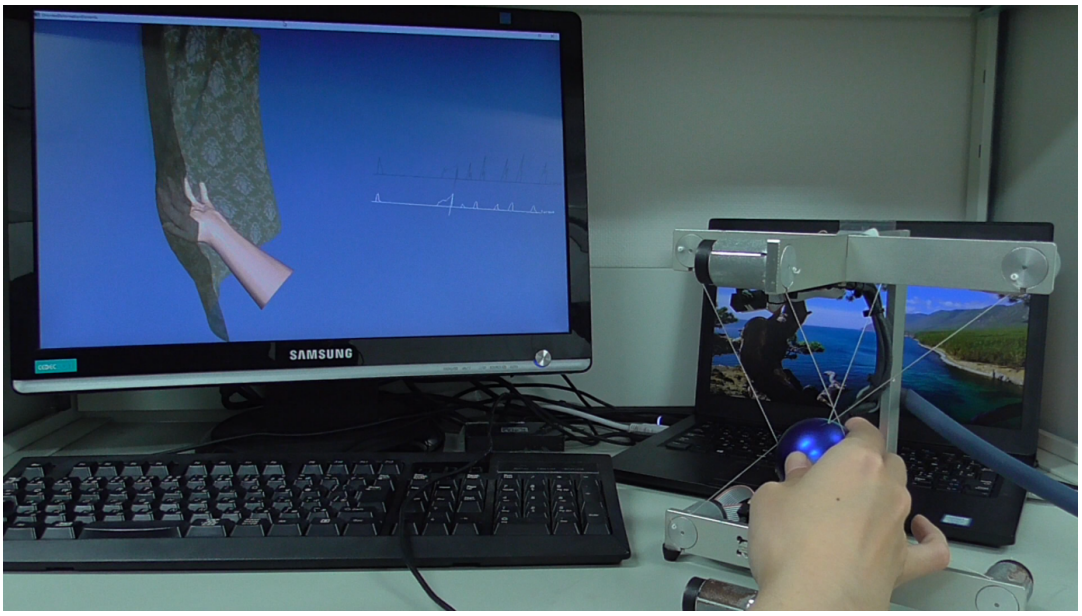

FIGURE 5.8: Screenshot of haptic interaction using our system.

5.5.3 Result

A scenario in the same virtual scene as Section 5.5.1 is presented, in which the user can interact with a complex deformable curtain using a deformable hand. Thanks to the CCD method and system efficiency, a large variety of interactions is supported in the proposed system, such as tapping, catching, pressing, and probing of stable haptic rendering (illustrated in Figure 5.8).

5.6 Summary

In this chapter, this study proposed a 6-DoF deformable haptic proxy that can interact with another deformable virtual object. The CCD is handled between spherical proxies and triangular meshes. The force artifacts caused by the tool's inertia were eliminated, and deformation was performed without the "tunneling" problem. The result simulation was stable even with fast motion input using a large feedback spring. An optional multi-rate haptic rendering is presented, which improves the stability but induces other force artifacts.

5.7 Discussion

This system only focuses on solving the CCD and force artifacts problem; therefore, it uses a less accurate contact tool model compared to the one in [Gar+11], in which the tool has 1441 triangular meshes for collision detection (not sure about the deformable object). The hand model used in the evaluations does not have a rigid-body or joint constraint; therefore, the hand model's physics behavior is less accurate than that used in [Gar+11].

Also, since the tool is represented with spherical particles, contact objects that are small and thin compared to the gap between particles may result in penetrations and "tunneling" problems (shown in Figure 5.9). In the future, it is an interesting topic to find solutions to cover this gap, such as using ellipsoids to achieve accurate contacts.

Other common limitations for both two proposed systems are introduced in Section 6.1.1.

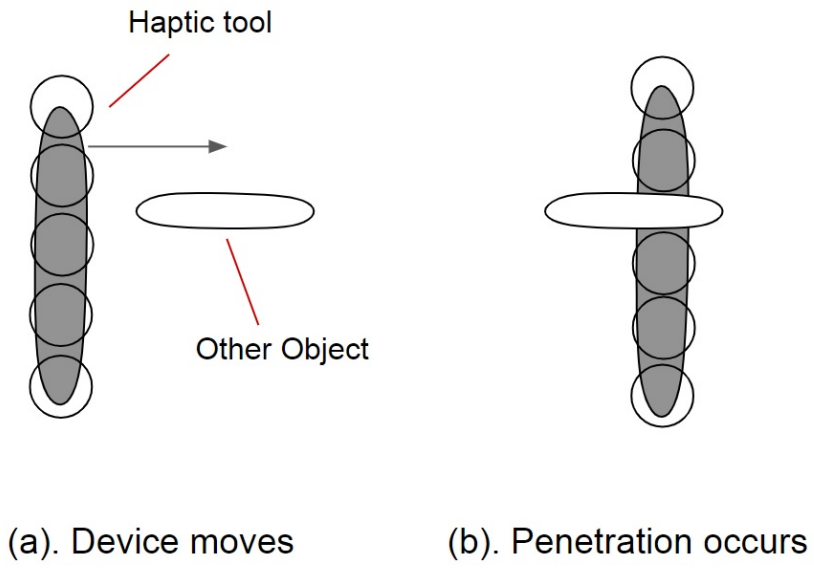

FIGURE 5.9: The limitation of interacting with small objects.

Chapter 6

Conclusion

In this dissertation, a haptic proxy, supporting dynamic objects and deformation on both the haptic tool and the other object, is proposed. This thesis starts with a 3-DHS and then extends to a 6-DHS.

The "tunneling" problem of collision detection, especially for thin and moving dynamic objects, is handled using a triangle-proxy CCD method and Proxy Pop-out. The force artifacts are eliminated using ZoD simulation of a position-based haptic constraint of the tool. The "tunneling" problem caused by constraint computation is also resolved efficiently. Therefore, the proposed method can provide only the contact forces of the deformed tool to the user. The systems can run in real-time, and the haptic interactions are stable thanks to the position-based simulation procedure and multi-rate architecture.

6.1 Discussion

Using this system, a user can play without the force artifacts. However, using spheres, contact accuracy is reduced. Even though there are other studies (such as [Wan+12b; Wan+12a; Wan+15]) also use sphere contact models, the computation cost of a massive number of particles will be problematic. Therefore, this system is more suitable for systems that require less contact accuracy compared to others. Fortunately, if the tool and object are deformable, the difference of the feedback force of inaccurate model compared to accurate one is not obvious.

The computation speed can be further boost using advanced pruning methods to reduce the collision detection cost of the proxy, which is the bottleneck of the efficiency.

6.1.1 Common Limitations

Multi-rate Haptic Rendering

While utilizing a multi-rate simulation, since the haptic thread does not consider the deformation constraint and only synchronizes one-way from the dynamic thread, high-rate haptic vibrations in the 3-DoF system occur during the probing if the rate of the dynamic thread drops to a low value. The same problem may arise in the 6-DoF system as the "dragging" forces (as discussed in Section 5.5.2). According to the review from the users, those force artifacts are not always a problem. The reason is also the haptic interaction with soft objects is somehow difficult to notice the moment of contact. At the same time, this force artifact provides an obvious clue of contact without damaging the system stability. Using compliant mechanisms [Pet+11] to synchronize the constraint information may alleviate the force artifacts.

Deformation Simulation

As a common limitation of SMM, the deformation does not have a clear relation to physics, making it difficult to adjust to represent a realistic behavior. The object stiffness in the proposed system is affected by the shape-matching constraint stiffness, OP group configurations, and the PBD iteration count. Future works on that could be establishing an approach to modeling OP with physics-based material properties, such as Young's modulus and Poissons's ratio. Fortunately, Tian et al.[Tia+15] show that OP can simulate detailed and complex simulation, and Pan et al.[Pan+15] proves that using geometric constraints can obtain similar results but much less computation cost compared to FEM.

6.2 Perspective for Future Research

Future researches about this thesis may focus on the adaption of a specific application. For example, by optimizing the tool's physics model, the proposed method can be used to achieve haptic interaction of a deformable virtual hand, similar to the model in [Gar+11], with no force artifacts. The User can use a haptic glove to input the finger motion without feeling the "dragging" force of the tool. Such a system can be used in surgery training such as palpation, rectal examination, or other operation with fingers. Other haptic devices, such as a haptic glove or even more DoF device, can be used in our system.

The number of the proxy particle was equal in our proposal. However, it is possible to use only part of the OP particles as the proxy particle to achieve a different state of grabbing, for example, holding a deformable object with different places and sizes of the object partial.

Since our system uses PBD for the dynamics simulation, a desirable future extension is to integrate with the most advanced PBD system, such as XPBD[MMC16], which supports solids, fluids, and other various physics properties. As a promising solution, the new XPBD with sub-steps [Mac+19] overcomes the convergence and efficiency trade-off problem and also reduces the constraint error and damping effect occurred in traditional implicit simulator even with large time-steps. Our haptic solution can be employed to achieve haptic interactions with objects that possess various physics properties to construct haptic systems of numerous physics phenomena for medical, scientific applications, and entertainment systems.

Bibliography

[AKO95] Yoshitaka Adachi, Takahiro Kumano, and Kouichi Ogino. “Intermediate representation for stiff virtual objects”. In: *Proceedings Virtual Reality Annual International Symposium'95*. IEEE. 1995, pp. 203–210.

[Ber+99] Jeffrey Berkley et al. “Real-time finite element modeling with haptic support”. In: *Proceedings of the ASME Design Engineering Technical Conferences*. 1999.

[BFA02] Robert Bridson, Ronald Fedkiw, and John Anderson. “Robust treatment of collisions, contact and friction for cloth animation”. In: *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*. 2002, pp. 594–603.

[BHB99] Peter J Berkelman, Ralph L Hollis, and David Baraff. “Interaction with a real time dynamic environment simulation using a magnetic levitation haptic interface device”. In: *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*. Vol. 4. IEEE. 1999, pp. 3261–3266.

[BJ08] Jernej Barbič and Doug L James. “Six-dof haptic rendering of contact between geometrically complex reduced deformable models”. In: *IEEE Transactions on Haptics* 1.1 (2008), pp. 39–52.

[BJ+88] Stanley J Bolanowski Jr et al. “Four channels mediate the mechanical aspects of touch”. In: *The Journal of the Acoustical society of America* 84.5 (1988), pp. 1680–1694.

[BMM15] Jan Bender, Matthias Müller, and Miles Macklin. “Position-Based Simulation Methods in Computer Graphics.” In: *Eurographics (tutorials)*. 2015, p. 8.

[BNC96] Morten Bro-Nielsen and Stephane Cotin. “Real-time volumetric deformable models for surgery simulation using finite elements and condensation”. In: *Computer graphics forum*. Vol. 15. 3. Wiley Online Library. 1996, pp. 57–66.

[BWK03] David Baraff, Andrew Witkin, and Michael Kass. “Untangling cloth”. In: *ACM Transactions on Graphics (TOG)* 22.3 (2003), pp. 862–870.

[CCO02] Jason J Corso, Jatin Chhugani, and Allison M Okamura. “Interactive haptic rendering of deformable surfaces based on the medial axis transform”. In: *Proceedings of EUROHAPTICS*. 2002, pp. 92–98.

[CDA99] Stéphane Cotin, Hervé Delingette, and Nicholas Ayache. “Real-time elastic deformations of soft tissues for surgery simulation”. In: *IEEE transactions on Visualization and Computer Graphics* 5.1 (1999), pp. 62–73.

[Cir+13] Gabriel Cirio et al. “Six-oof haptic interaction with fluids, solids, and their transitions”. In: *2013 World Haptics Conference (WHC)*. IEEE. 2013, pp. 157–162.

[CSB95] J Edward Colgate, Michael C Stanley, and J Michael Brown. “Issues in the haptic display of tool use”. In: *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*. Vol. 3. IEEE. 1995, pp. 140–145.

[CSC04] Daniela Constantinescu, Septimiu E Salcudean, and Elizabeth A Croft. "Haptic rendering of rigid body collisions". In: *12th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2004. HAPTICS'04. Proceedings.* IEEE. 2004, pp. 2–8.

[cT00] Murat Cenk Çavuşoğlu and Frank Tendick. "Multirate simulation for high fidelity haptic interaction with deformable objects in virtual environments". In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*. Vol. 3. IEEE. 2000, pp. 2458–2465.

[DAK04] Christian Duriez, Claude Andriot, and Abderrahmane Kheddar. "A multi-threaded approach for deformable/rigid contacts with haptic feedback". In: *12th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2004. HAPTICS'04. Proceedings.* IEEE. 2004, pp. 272–279.

[dCL99] Diego d'Aulignac, Murat Cenk Cavusoglu, and Christian Laugier. "Modeling the dynamics of the human thigh for a realistic echographic simulator with force feedback". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 1999, pp. 1191–1198.

[Dur+05] Christian Duriez et al. "Realistic haptic rendering of interacting deformable objects in virtual environments". In: *IEEE transactions on visualization and computer graphics* 12.1 (2005), pp. 36–47.

[DZ93] Paul Dworkin and David Zeltzer. "A New Model for E cient Dynamic Simulation". In: *4th Eurographics workshop on Animation & Simulation*. 1993.

[EC+16] David Escobar-Castillejos et al. "A review of simulators with haptic devices for medical training". In: *Journal of medical systems* 40.4 (2016), p. 104.

[Fis] *Fish Model*.

[Gar+11] Carlos Garre et al. "Interactive simulation of a deformable hand for haptic rendering". In: *2011 IEEE World Haptics Conference*. IEEE. 2011, pp. 239–244.

[GO10] Carlos Garre and Miguel A Otaduy. "Haptic rendering of objects with rigid and deformable parts". In: *Computers & Graphics* 34.6 (2010), pp. 689–697.

[got18] gotferdom. *Free hand 3D model*. 2018 (accessed Janurary 25, 2020). URL: https://www.turbosquid.com/3d-models/hand-hdri-shader-3d-model-1311775.

[Gre+00] Arthur Gregory et al. "Six degree-of-freedom haptic display of polygonal models". In: *Proceedings Visualization 2000. VIS 2000 (Cat. No. 00CH37145)*. IEEE. 2000, pp. 139–146.

[Has+04] Shoichi Hasegawa et al. "Real-time rigid body simulation for haptic interactions based on contact volume of polygonal objects". In: *Transactions of the Society of Instrument and Control Engineers* 40.2 (2004), pp. 122–131.

[JCC05] Seongki Jun, Jinbok Choi, and Maenghyo Cho. "Physics-based s-adaptive haptic simulation for deformable object". In: *2006 14th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*. IEEE. 2005, pp. 477–483.

[JP00] Doug L James and Dinesh K Pai. "Pressure masks for point-like contact with elastic models". In: *Proc. Fifth Phantom User Group Workshop*. 2000.

[JP03] Doug L James and Dinesh K Pai. "Multiresolution green's function methods for interactive simulation of large-scale elastostatic objects". In: *ACM Transactions on Graphics (TOG)* 22.1 (2003), pp. 47–82.

[JP05] Doug L James and Dinesh K Pai. "A unified treatment of elastostatic contact simulation for real time haptics". In: *ACM SIGGRAPH 2005 Courses*. 2005, 141–es.

[Kim+02] Young J Kim et al. "Six-degree-of-freedom haptic display using localized contact computations". In: *Proceedings 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. HAPTICS 2002*. IEEE. 2002, pp. 209–216.

[Mac+19] Miles Macklin et al. "Small Steps in Physics Simulation". In: *Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '19. Los Angeles, California: Association for Computing Machinery, 2019. ISBN: 9781450366779. DOI: 10.1145/3309486.3340247. URL: https://doi.org/10.1145/3309486.3340247.

[MC11] Matthias Müller and Nuttapong Chentanez. "Solid Simulation with Oriented Particles". In: *ACM Trans. Graph.* 30 (July 2011), p. 92.

[MMC16] Miles Macklin, Matthias Müller, and Nuttapong Chentanez. "XPBD: position-based simulation of compliant constrained dynamics". In: *Proceedings of the 9th International Conference on Motion in Games*. 2016, pp. 49–54.

[MML02] Frédéric Mazzella, Kevin Montgomery, and J-C Latombe. "The forcegrid: a buffer structure for haptic interaction with virtual elastic objects". In: *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*. Vol. 1. IEEE. 2002, pp. 939–946.

[MN04] Probal Mitra and Günter Niemeyer. "Dynamic proxy objects in haptic simulations". In: *IEEE Conference on Robotics, Automation and Mechatronics, 2004.* Vol. 2. IEEE. 2004, pp. 1054–1059.

[MN07] Probal Mitra and Günter Niemeyer. "Haptic simulation of manipulator collisions using dynamic proxies". In: *Presence: Teleoperators and Virtual Environments* 16.4 (2007), pp. 367–384.

[MS+94] Thomas H Massie, J Kenneth Salisbury, et al. "The phantom haptic interface: A device for probing virtual objects". In: *Proceedings of the ASME winter annual meeting, symposium on haptic interfaces for virtual environment and teleoperator systems*. Vol. 55. Chicago, IL. 1994, pp. 295–300.

[Mül+05] Matthias Müller et al. "Meshless deformations based on shape matching". In: *ACM transactions on graphics (TOG)* 24.3 (2005), pp. 471–478.

[Mül+07] Matthias Müller et al. "Position based dynamics". In: *Journal of Visual Communication and Image Representation* 18.2 (2007), pp. 109–118.

[OG07] Miguel A Otaduy and Markus Gross. "Transparent rendering of tool contact with compliant environments". In: *Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC'07)*. IEEE. 2007, pp. 225–230.

[OL05a] Miguel A Otaduy and Ming C Lin. "Sensation preserving simplification for haptic rendering". In: *ACM SIGGRAPH 2005 Courses*. 2005, 72–es.

[OL05b] Miguel A Otaduy and Ming C Lin. "Stable and responsive six-degree-of-freedom haptic manipulation using implicit integration". In: *First Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics Conference*. IEEE. 2005, pp. 247–256.

[ORC06] Michael Ortega, Stephane Redon, and Sabine Coquillart. “A six degree-of-freedom god-object method for haptic display of rigid bodies”. In: *IEEE Virtual Reality Conference (VR 2006)*. IEEE. 2006, pp. 191–198.

[Ota+09] Miguel A Otaduy et al. “Implicit contact handling for deformable objects”. In: *Computer Graphics Forum*. Vol. 28. 2. Wiley Online Library. 2009, pp. 559–568.

[Pan+15] Junjun Pan et al. “Real-time haptic manipulation and cutting of hybrid soft tissue models by extended position-based dynamics”. In: *Computer Animation and Virtual Worlds* 26.3-4 (2015), pp. 321–335.

[Pet+11] Igor Peterlik et al. “Constraint-based haptic rendering of multirate compliant mechanisms”. In: *IEEE Transactions on Haptics* 4.3 (2011), pp. 175–187.

[Pic+00] Guillaume Picinbono et al. “Anisotropic elasticity and force extrapolation to improve realism of surgery simulation”. In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*. Vol. 1. IEEE. 2000, pp. 596–602.

[Pro97] Xavier Provot. “Collision and self-collision handling in cloth model dedicated to design garments”. In: *Computer Animation and Simulation’97*. Springer, 1997, pp. 177–189.

[RKC02] Stéphane Redon, Abderrahmane Kheddar, and Sabine Coquillart. “Fast continuous collision detection between rigid bodies”. In: *Computer graphics forum*. Vol. 21. 3. Wiley Online Library. 2002, pp. 279–287.

[RKK97] Diego C Ruspini, Krasimir Kolarov, and Oussama Khatib. “The haptic display of complex graphical environments”. In: *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 1997, pp. 345–352.

[RM00] Paul Read and Mark-Paul Meyer. *Restoration of motion picture film*. Elsevier, 2000.

[SDC08] Guillaume Saupin, Christian Duriez, and Stephane Cotin. “Contact model for haptic medical simulations”. In: *International Symposium on Biomedical Simulation*. Springer. 2008, pp. 157–165.

[SML02] Kenneth Sundaraj, César Mendoza, and Christian Laugier. “A fast method to simulate virtual deformable objects with force feedback”. In: *7th International Conference on Control, Automation, Robotics and Vision, 2002. ICARCV 2002.* Vol. 1. IEEE. 2002, pp. 413–418.

[Tan+09] Min Tang et al. “ICCD: Interactive continuous collision detection between deformable models using connectivity-based culling”. In: *IEEE Transactions on Visualization and Computer Graphics* 15.4 (2009), pp. 544–557.

[Tia+13] Yuan Tian et al. “Haptic-enabled interactive rendering of deformable objects based on shape matching”. In: *2013 IEEE International Symposium on Haptic Audio Visual Environments and Games (HAVE)*. IEEE. 2013, pp. 75–80.

[Tia+15] Yuan Tian et al. “Stable haptic interaction based on adaptive hierarchical shape matching”. In: *Computational Visual Media* 1.3 (2015), pp. 253–265.

[TOT13] Kazuyoshi Tagawa, Tatsuya Oishi, and Hiromi T Tanaka. “Adaptive and embedded deformation model: An approach to haptic interaction with complex inhomogeneous elastic objects”. In: *2013 World Haptics Conference (WHC)*. IEEE. 2013, pp. 169–174.

[VPDL12] Emmanuel Vander Poorten, Eric Demeester, and Piet Lammertse. "Haptic feedback for medical applications, a survey". In: *Proceedings Actuator 2012* (2012).

[Wan+12a] Dangxiao Wang et al. "Configuration-based optimization for six degree-of-freedom haptic rendering for fine manipulation". In: *IEEE transactions on haptics* 6.2 (2012), pp. 167–180.

[Wan+12b] Dangxiao Wang et al. "Six-degree-of-freedom haptic simulation of organ deformation in dental operations". In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. 2012, pp. 1050–1056.

[Wan+15] Dangxiao Wang et al. "Interactive haptic simulation of tooth extraction by a constraint-based haptic rendering approach". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 278–284.

[XB16] Hongyi Xu and Jernej Barbič. "6-DoF haptic rendering using continuous collision detection between points and signed distance fields". In: *IEEE transactions on haptics* 10.2 (2016), pp. 151–161.

[Xia18] Pingjun Xia. "New advances for haptic rendering: state of the art". In: *The Visual Computer* 34.2 (2018), pp. 271–287.

[Ye+17] Juntao Ye et al. "A unified cloth untangling framework through discrete collision detection". In: *Computer Graphics Forum*. Vol. 36. 7. Wiley Online Library. 2017, pp. 217–228.

[ZS95] Craig B Zilles and J Kenneth Salisbury. "A constraint-based god-object method for haptic display". In: *Proceedings 1995 ieee/rsj international conference on intelligent robots and systems. Human robot interaction and cooperative robots*. Vol. 3. IEEE. 1995, pp. 146–151.