COLLISION RESPONSE FOR VIRTUAL LAPARASCOPIC SURGERY

by

JITESH BUTALA

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2005

ACKNOWLEDGEMENTS

I would like to thank my research supervisor, Dr. Venkat Devarajan, for his support and encouragement during the period of my study at University of Texas at Arlington. I also extend my sincere thanks to my thesis committee members, Dr. Kamisetty Rao and Dr. Michael Manry.

I would like to thank all my colleagues at the Virtual Environment Laboratory, especially Yunhe Shen and Xiuzhong Wang who have looked out for me at every step of my research.

I would like to thank God for giving me the belief that I would succeed in the times when I thought that everything was in vain.

In addition, I would like to thank my roommates, my friends and mentors for their undying support and understanding.

I cannot end my acknowledgement without expressing my gratitude to my parents, Devi and Deepa, my sister, Harshala, my aunt, Maya, and my dogs, Marc and Scooby, without whom I would not have been able to achieve what I have achieved so far. I love you all. Thanks for being there for me whenever I needed you.

July 25, 2005

ABSTRACT

COLLISION RESPONSE FOR VIRTUAL

LAPARASCOPIC SURGERY

Publication No. _____

Jitesh Butala, M.S.

The University of Texas at Arlington, 2005

Supervising Professor: Venkat Devrajan

During laparoscopic surgery the surgeon operates inside the abdomen guided by the 2D imagery of the operating area. This image is obtained via a camera inserted through a small incision. Laparoscopic surgery, in contrast to open surgery, results in low pain and recovery time for the patient. However, the surgeons have some difficulty, as the training required for this type of surgery is more intense and rigorous than that for conventional open surgery. Hence, surgical trainers for laparoscopic surgery are important. The main difference in the different types of surgical trainers is the degree to which the actual environment of the surgical procedure is depicted. This thesis describes a virtual reality based surgical trainer and examines the problem of collision

response and improvements to the current algorithms for applying physical and visual response in real-time. Mass-spring models are used to simulate the behavior of the various objects, which require extensive numerical methods for implementation. Hence, the computations required to calculate the new parameters of the mass-spring system must be inexpensive. The methods that we describe and have implemented make a few approximations so as to make the system real-time. We classify the response according to the type of collisions so as to get the best possible response after consideration to the various constraints. We also investigate the problem of inter-penetrations of objects due to the lag in discrete-time sampling for collision detection and propose an algorithm to overcome inter-penetrations. We have implemented this fast and stable collision response algorithm.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Virtual Reality

Virtual Reality (VR) is a term coined by Jaron Lanier, founder of Visual Programming Language Research in 1989. It refers to the computer generated simulation of a real or imagined environment that may provide tactile feedback and that can be experienced visually in 3D or that may provide interactive experience visually in real-time motion with sound. Initially, VR was used for flight simulators and virtual tours of buildings and environments.

1.2 Minimally Invasive Surgery (MIS)

Minimally Invasive Surgery is also known as Laparoscopic Surgery. It is a surgical procedure that does not require large incisions on the body like open surgery. Only tiny holes are required in laparoscopic surgery. A small gas-tight pipe like structure with a sharp tip called a trocar is passed through the incision, is sealed to the tissue with pressure, expanding the peritoneal space. A tiny camera called the endoscope is inserted through the port and captures high-resolution images of the inside of the abdomen, which are then viewed through a television monitor. The surgical instruments are similarly inserted through other trocar ports.

1.2.1 MIS: Advantages and Disadvantages

The incisions are only 3-6 mm in length. This smaller area of tissue injury results in lesser pain and faster healing. However, the smaller incisions cost the

surgeons. The surgeons lose direct contact with the operation site. Also, due to the trocar ports, the surgeons lose some of the manipulative freedom, which is available in open surgery. The surgeons need to get used to the loss of depth perception as the operation site is now seen on a 2D television monitor. Performing operations under such demanding conditions requires special training for the surgeons.

1.2.2 MIS: Current Training Procedures

There are more than tens of thousands of deaths and disabilities in the United States alone each year, which are attributed to "surgical errors". One of the ways to reduce this is through more extensive training of surgeons. Currently, basic visual and manipulative skills can be learned using inexpensive training devices, which allow the trainee to learn navigation and acquire basic manual competence with the help of a simple visual feedback system. However, these non-interactive models lack real-life effects like the blood flow and tissue deformation. Training on actual patients under the supervision of an expert surgeon is quite common, but this is very expensive and might lead to a surgical error. Training on live animals is an ethical issue. It also has the same problem as those associated with other inflexible training devices. The use of cadavers to practice lacks the real-life qualities of blood flow and tissue reaction. Although, these procedures may be helpful in specific circumstances, they are not flexible as those based on virtual reality. VR-based trainers have the ability to run different procedures and present different complications so that the surgeons become adept in the procedure. Also, no supervision is required, so there is no need for the expert surgeon to expend his or her time.

1.2.3 VR-based Training

VR-based trainers are very similar to flight simulators. They both require a virtual environment. Both are used to provide rigorous training and they are critical as they provide the trainees with useful insight on what they are going to face when they go out in the real world and in emergency situations. The systems need to be fast enough to be interactive. This poses strict real-time requirements for the designs. The amount of data to be rendered is humongous for both trainers. The surgical simulators need to render complex anatomical models while the flight simulators need to render country-sized image databases. The main difference between flight simulators and surgical simulators is that surgical simulators need haptics - the sense of touch. This is an important cue to a surgeon in case of MIS. It turns out the haptic cues can be built into a VR-based trainer. Thus, we can conclude that VR-based technology has the potential to develop next generation surgical training systems.

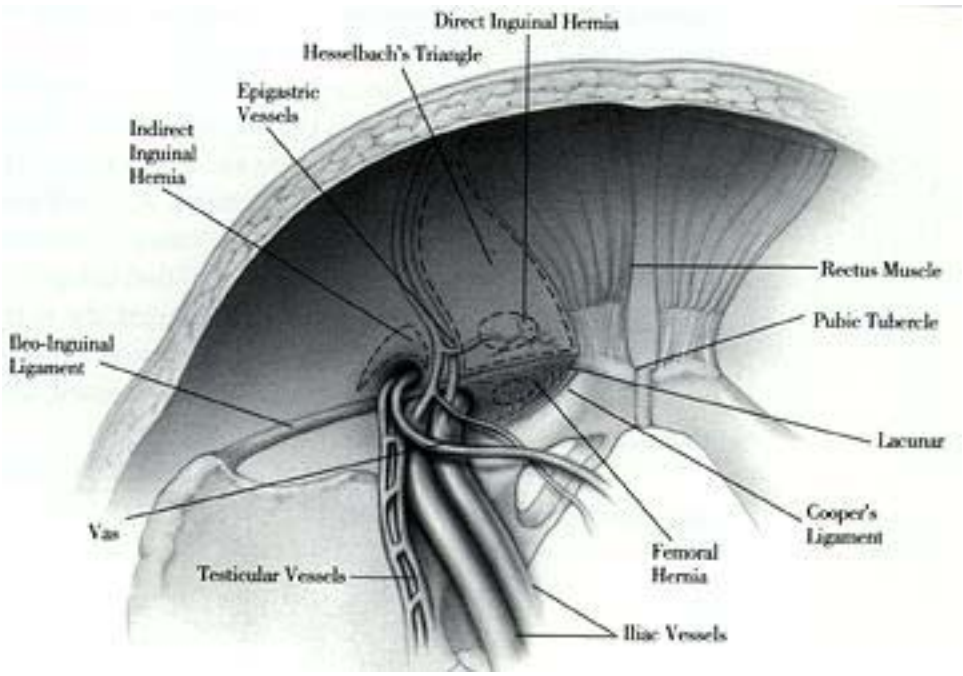
1.3 Laparoscopic Hernia Surgery

Hernia is defined as the condition in which a part of the intestine bulges through a weak area in the abdominal muscle [20]. An inguinal hernia occurs between the abdomen and the thigh i.e. in the groin and is called so because the intestine pushes through a weak spot in the inguinal canal. The inguinal canal is a triangular opening between the layers of abdominal muscle near the groin. The hernia repair, called herniorrhaphy, consists of first pulling the sac back into the abdominal cavity and then the weakened hole is exposed which is covered with a polymer mesh patch [21]. A scene from an actual laparoscopic surgery is shown in Figure 1.1.

Source: Videoscopic institute of Atlanta, LLC

Figure 1.1 Laparoscopic inguinal hernia repair surgery

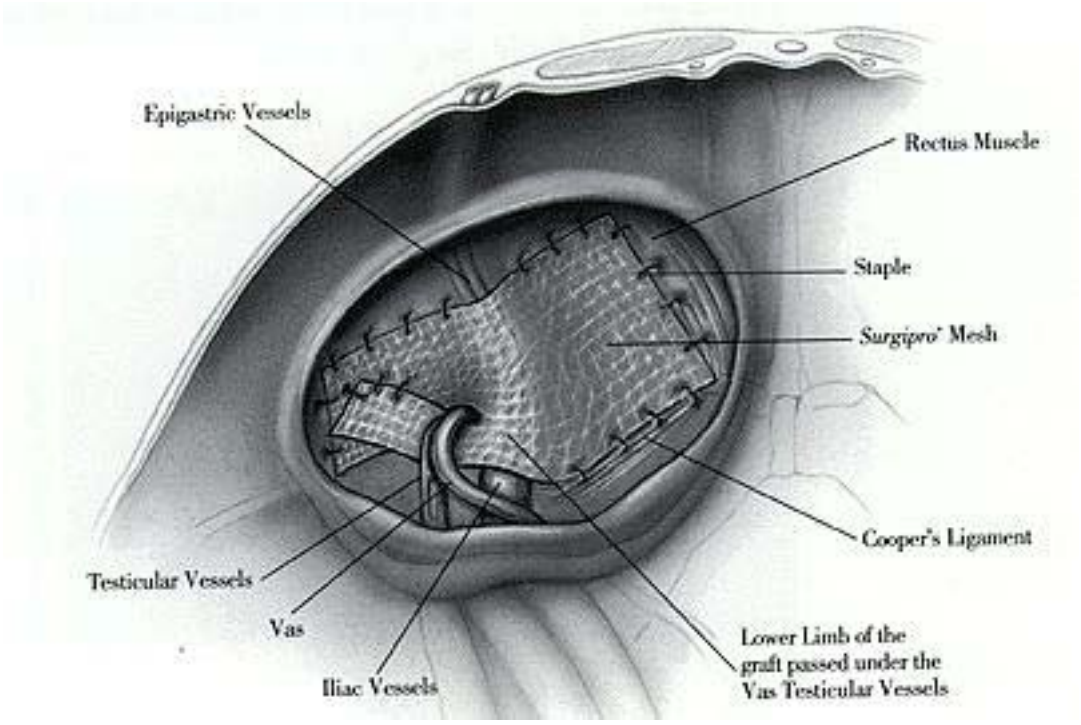
Figure 1.2 shows the anatomy of the abdomen with the different parts.



Source: http://www.laparoscopy.net

Figure 1.2 Anatomy of the abdomen

Figure 1.3 shows the placement of the mesh after the laparoscopic surgery.



Source: http://www.laparoscopy.net

Figure 1.3 Laparoscopic mesh placement

1.4 Motivation behind Virtual Laparoscopic Surgery Simulator

About 600,000 hernia operations are performed annually in the United States according to a report from the Society of American Gastrointestinal Endoscopic Surgeons (SAGES). VR-based training reduces the learning curve and also it has other advantages, as discussed, over conventional open hernia repair procedures. Hence, a surgical simulator to simulate the hernia repair of commonly occurring inguinal hernia is being developed at the Virtual Environment Laboratory, University of Texas at Arlington (Figure 1.4).

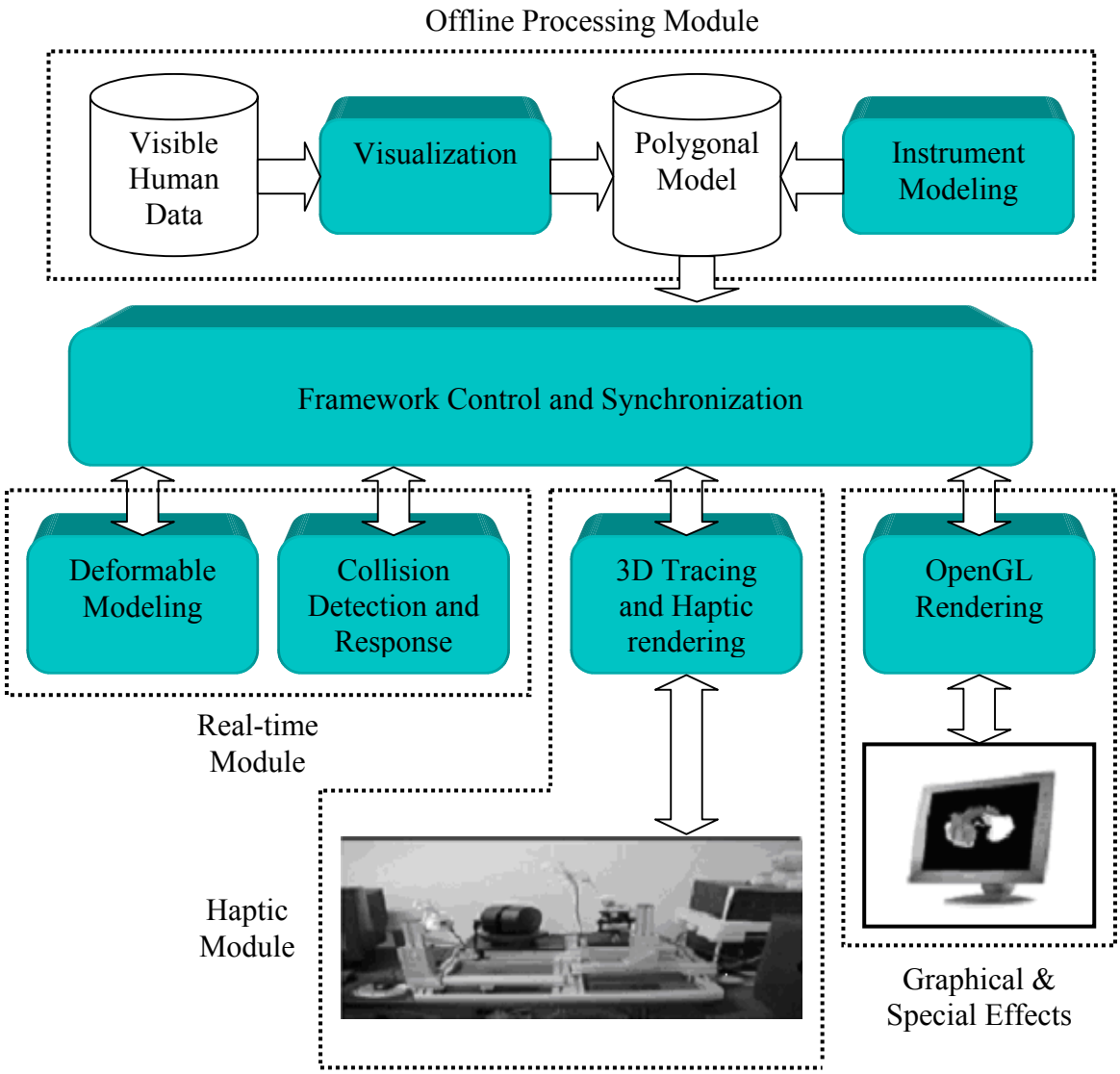


Figure 1.4 Block diagram of the virtual laparoscopic surgery simulator [39]

The system consists of four main modules: Offline Processing, Graphical and Special Effects, Real-time Module and Haptic block.

1.4.1 Offline Processing block

The offline processing block prepares the necessary input of polygonal geometry by converting the raw image data of the Visible Human Data (VHD) slices obtained

from The National Institute of Health (NIH). A VHD data slice obtained from NIH is shown in Figure 1.5.

Figure 1.5 Visual Human Data (VHD) slice

A realistic texture image [15] is added onto the models. A surface rendering technique implemented using the Marching cubes [26] algorithm is used to accomplish surface polygonization. Finally, a realistic inguinal hernia scene [18] is generated from the basic image files of the VHD. The 3D models of all instruments [32] needed for surgery simulation is generated using 3D Studio Max. An example of the iliopsoas muscle and the transverses abdominis muscle is given in Figure 1.6.

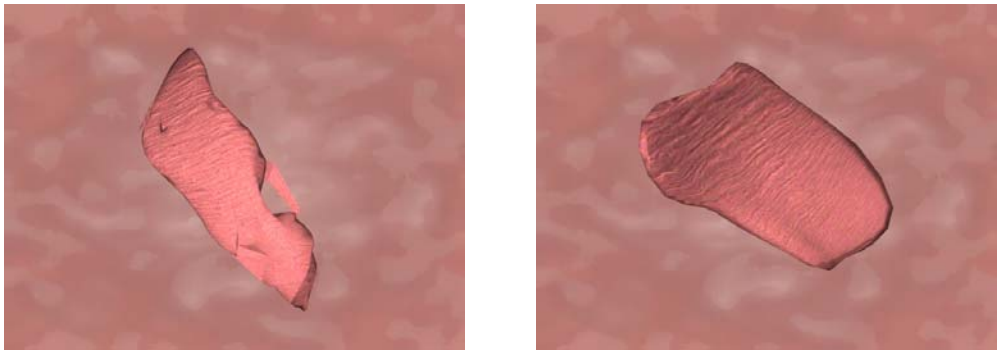


Figure 1.6 3D Studio Max models of (a) iliopsoas (b) transverses abdominis muscle

1.4.2 Graphical and Special Effects block

The graphical and special effect block provides a realistically rendered view of the geometrical information obtained from the above-discussed block. It renders the patient's inside at an interactive frame rate. It also consists of a *special FX module* [37] for simulating special visual effects that occur during virtual surgery such as bleeding, cauterization, irrigation, suction, suturing, stapling etc. A major task of stapling the mesh [14] in a virtual laparoscopic inguinal hernia surgery is also implemented with collision detection and primitive response methods.

The Instructor Station provides an interactive graphical interface for the trainee and provides a means to record the training exercises, evaluate them with various performance metrics and compare their simulations with reference to any other simulation runs. This does the work of the supervising surgeon [13].

1.4.3 Real-time Module

The real-time module has two sub-blocks named collision detection and deformation model. It is a core block that requires intensive computation and interaction with physical parameters needed for simulation. The collision module helps in interference detection among the different virtual models present in the scene and provides the surgeon accurate visual cues in real time. A real-time collision detection algorithm [39] for interactions between instruments and deformable bodies is implemented for haptic environments. An appropriate collision response force remained to be provided in order to avoid penetration of the models. This is the subject matter of this thesis. The deformation module provides realistic physical effects for the input

geometry in the scene. The physical models should incorporate accurate tissue properties for accurate deformations and interactive forces. Mass-spring models and finite element methods are the common computational models employed for adding physical properties. Mass-spring based deformable models are used in the simulation.

1.4.4 Haptic Module

This block provides force-feedback effect for the practicing surgeon as he interacts with the anatomy. An update rate of about 1 KHz is required for real-time force-feedback for a stable and realistic feel of the virtual objects. The module also consists of hardware with an arm as a support to provide the necessary tactile feedback. At Virtual Environment Laboratory, University of Texas at Arlington, PHANToM devices from Sensable Technologies Inc. are used for this purpose.

1.5 Need for Real-time Collision Detection and Response

Collision detection and response is placed under real-time module, as it is necessary to provide accurate intersection points at real-time rates and provide appropriate collision response. Collision detection module identifies the interaction between deformable bodies, between deformable bodies and instruments, between deformable bodies and rigid bodies like bones and between rigid bodies and instruments. The tactile force felt between the instrument and tissue by the trainee is an important cue. The instrument is not deformable and hence, graphics hardware can be exploited to find the intersection and the response between different instruments. But, the response between instrument and tissue cannot be computed directly on the

hardware as the geometry of the tissue changes over time requiring intensive calculations to calculate the collision points and the respective responses.

The collision detection algorithms are discrete. Currently, a few milliseconds for detecting and reporting hundreds of colliding pairs of primitives is the best ratio of *Collision Detection Time Vs Actual Output Size* ex: the number of detected collisions. Since, the real-time response is important to provide proper visual as well as tactile feedback, the purpose of the thesis is twofold:

- Develop a real-time collision response module for good visual and tactile feedback for the trainees.
- Develop an algorithm to achieve collision free state after the inter-penetrations due to the missed collisions by the discrete-time collision detection algorithms.

Towards this effort, a collision response algorithm for interaction between deformable and rigid bodies has been developed.

1.5.1 Problem Statement

To develop a real-time, accurate and efficient collision response algorithm to calculate the response between instrument and deformable bodies and to develop an algorithm to remove any unexpected inter-penetrations due to missed collisions by the discrete-time collision detection algorithm.

The top-level objective is to develop an algorithm to accurately model the responses of the tissue and the instruments on colliding in real life examples.

1.6 Organization of the Thesis

Chapter 1 gave a brief introduction about the application of VR-based training for surgeries and the need for efficient collision detection and response to provide realistic feedback. Chapter 2 covers the previous work done at various laboratories around the world. Chapter 3 describes the theoretical background, derivation and the design of the proposed collision response algorithm. Chapter 4 discusses the implementation of the collision response algorithm. Chapter 5 discusses the results of the algorithm and concludes the work with a brief report on potential future work.

CHAPTER 2

PREVIOUS WORK

The state-of-the-art collision detection algorithms work really well and they have achieved close to haptic rates. But if the collision response is not accurate, then the visual as well as the haptic feedback will not provide the operator a realistic feel. Hence, our focus on the collision response is to achieve good visual and haptic feedback. In the following sections, we will describe the collision response that have been used for virtual environments like surgery and cloth simulation.

2.1 Collision Response (CR)

This section will describe, in brief, a few systems in which collision response has been used.

2.1.1 Systems based on Conservation Laws

Uno et al [46] describe an experiment in which the sensitivity of subjective presence to varying collision response parameters is examined. They have tried to elicit the factors that contribute to the subjective experience of *presence* in an immersive virtual environment – the sense of being in the environment depicted by the computer generated displays. The subjective presence is gauged along three orthogonal dimensions: the extent to which the participant has a sense of *being there*, *reality* and *place*. They have described two approaches to solving the collision response problem:

- *Physical Equations* – three kinds of equations are used: conservative laws of momentum, conservative law of kinetic energy and the relative velocity at post-collision position.
- *Energy Conversation Method* – physical parameters, elasticity and friction are considered to determine an impulse force. This approach simplifies the handling of elasticity and friction to give the illusion of their correct operation, but without the computational expense of full simulation. Four impulses corresponding to special collision conditions are first calculated. The impulse response can be obtained by the linear interpolation of the special case impulse response.

C. O'Sullivan et al [41] attempt to refine the collision response determination much like collision detection so that the accuracy of the response would be dependent on the processing time allocated to it. This system allows for very accurate to inaccurate response depending upon the CPU times allocated to the response process. Guidelines such as laws of conservation of energy and momentum are always followed. The objects are treated as the union of spheres represented by their sphere trees. A simplified impulse based method is used to calculate response [3] given the point of collision and the collision direction. The interpenetrations, which are an integral part of the system due to discrete-time implementation, are solved by performing some sort of backtracking or interpolation to determine the states of the objects at the exact instant of collision. Hence, this system cannot be successfully modeled as a real-time system.

M. Moore et al [27] describe a collision response method based on the conservation of linear and angular momentum for the bodies involved in the collision. Here, the body is not considered to be made up of discrete particles. This approach is very simple and works only for rigid bodies.

P. Volino et al [47] propose a collision response algorithm, which is based on a combination of the elastic model and the mass-spring model. Immediate position and velocity corrections are performed in order to put the elements in an acceptable position and prevent their speeds from pushing further into each other. Force correction then enforces the maintaining collision distances between iterations of collision detection. Friction effects are simulated by a velocity correction and a force correction.

B. Geiger et al [12] propose an algorithm based on the concept of *compliant motion*. It is a physically based method. The concept of compliant motion is basically taken from the field of Robotics. A reaction force is defined at each sample point, which depends upon the surface normals of the two colliding surfaces and the depth of interpenetration of the surfaces. The global force is calculated by adding the forces at the sample points. Also, the rotational velocity is modified by using the law of conservation of momentum. The author claims that despite the simplicity and approximate character, this method can be used to simulate quite complex motions, like an insertion of a bolt into a nut. But the main limitation is that the method only works well for non-deformable objects. The forces calculated by this method cannot realistically simulate the deformable objects.

2.1.2 Systems based on Constraints

D. Baraff et al [3] propose an idea for collision response for animated cloth simulation, which couples a technique for enforcing constraints on individual cloth particles with an implicit integration method. This method takes the stretch, shear and bending forces into consideration. It also considers the damping and the constraint forces. The force on the particles is a summation of all the effects mentioned above. It takes $O(n^{1.5})$ calculations where n is the number of particles of the cloth. This method, though attractive, is unlikely to be implemented in real-time.

J. Mezger et al [29] proposed another technique where the coherence between the consecutive frames can be used to reduce the computational load. This method is applicable for cloth simulation like the draping of cloth over a table. The velocities of the cloth particles are constrained in a direction. This direction is not the normal direction of the close object, but in the direction of the closest point pair of the close faces. This approach leads to a more stable collision response for non-smooth surface and edge collisions, as there are no discontinuities in the constrained direction. The collision response module of this system distinguishes between collisions of two deformable faces and collisions between a deformable face and a face of the pre-computed rigid environment. In the first case, no velocity and hence no momentum is transferred between the faces while in the second case, the velocity of the rigid body in the constrained direction has to be added to the velocity of the deformable object. Since, only one direction of the velocity is constrained, the particle is free to move according to the forces acting on it externally.

2.1.3 Systems based on Impulse Dynamics

B. Mirtich et al [30] explains an impulse-based simulation of rigid bodies. The impulse-based method is one of the oldest and the simplest methods for collision response. This method is not very accurate. This approach is well suited to modeling physical systems with large number of collisions, or with contact modes that change frequently. All the different types of contact i.e. colliding, rolling, sliding and resting are modeled through a series of impulses between the objects in contact and hence, this method is faster that constraint-based methods. The main disadvantage of this method is that it cannot be used for deformable object collisions.

P. Sovis et al [40] describes an algorithm in which impulse dynamics is used. This is the only approach that has classified the collision based on the type of penetration i.e. vertex-face, vertex-vertex, vertex-edge, and edge-edge collisions. Classifications of this type are very helpful as the formulae that are used for each case need to represent only that particular case and hence, are generally simple. This system deals with 3D meshes and objects like tetrahedral meshes and hence, this approach is not applicable to our system directly.

F. Policarpo et al [34] describes one of the simplest collision response algorithms. This algorithm is almost the de facto standard of the industry. This treats the colliding objects as a particle and the surface that it is colliding with as a plane. The particle is reflected from the surface making an exit angle with respect to the normal equal to the incident angle. The behavior can be modified and made more convincing by using a bump and friction factor. These take values in the range 0 to 1. A value of 0

for the friction factor means infinite friction i.e. the box will stick to the collision plane and a friction factor of 1 means no friction. A bump factor of 0 means no bump i.e. the box will slide on the collision plane and bump factor of 1 means that normal component of velocity will be maintained i.e. the box will bounce with no damping. The main problem is that this does not consider deformable objects and the response to such objects is not very good.

P. Volino et al [48] [49] present a general geometrical correction method for enforcing collisions and other geometrical constraints between polygonal mesh surfaces. It is based on a global resolution scheme that takes advantage of an efficient use of the conjugate gradient algorithm to find the appropriate displacement of the mesh vertices that would satisfy all the constraints simultaneously and according to the momentum conservation laws. It describes a method in which the colliding primitives can be considered as points and the collision response calculated using impulse based method. Then the collision response force is distributed on the vertices in the collision using the geometrical property correction process.

T. Kurihara et al [19] describe an algorithm for the animation of hair. The collision response is based on impulse dynamics. The force on a particle is divided into two components, the unconstrained component and the constrained component. The unconstrained component has no relationship with the collision and is solely based on the external forces acting on the particle. The constrained component of force is adjusted so that the collision is avoided. Inelastic collision can be simulated if the constraint is applied only if the input force is not lifting the colliding point away from

the surface. This approach is good, but they appear to have not considered the various cases of the collisions and hence, the response is unlikely to be accurate.

2.1.4 Systems based on other ideas

V. Vuskovic et al [50] takes a different approach to the modeling of the 3D objects. They use the finite element method. They describe the main components necessary for haptic feedback in virtual surgery. They also specify a method to measure in-vivo the material parameters figuring in the developed elasto-mechanic models of living tissues. This method is accurate and can be used for real-time applications.

C. Basdogan et al [5] uses a hybrid approach in which the finite element model and a particle model are used to simulate flexible dynamics of the duct and the catheter respectively. The forceps were modeled as connected line segments. The interactions between the particles of the catheter and duct were simulated using point-based haptic techniques. The interactions between the forceps and the duct as well as the catheter were simulated using ray-based haptic interaction techniques. The main challenge in this method is to link the low fidelity particle system with the high fidelity finite element system while satisfying the continuity requirements.

O. Etzmuß et al [11] describe a collision response algorithm which adds and subtracts virtual particles as needed. The collisions are classified as face-particle and edge-edge intersections. Hence, when coarse meshes collide with an object, they are automatically refined. Hence, this model allows physically accurate simulations that require a much smaller number of particles than regular particle systems, which in turn allows faster simulations.

C. O'Sullivan [42] explains a model of the human perception of collisions, which is based upon the 2D measurements of eccentricity and separation. An eye-tracker is used to locate the user's point of fixation and by using a priority queue scheduling algorithm, the perceived collision inaccuracy was approximately halved. This system basically uses the eye-tracker output to determine which part of the scene has to have accurate collision response and the remaining scene has approximate collision response.

B. Lee [23] describe a method for contact modeling for deformable models. The algorithm is used to simulate contact between rigid instrument and deformable tissue. A divide and conquer strategy is used to redistribute an arbitrary field of displacements on organ surfaces into an equivalent field of displacements at the nodes. This gives accurate modeling and can be adopted in real-time systems.

The summary of the different methods is given in Table 2.1.

Table 2.1 Comparison of previous systems

Authors	**Real-time**	**Applicable to deformable objects**	**Applicable to the mass-spring system**	**Basis of algorithm**
Uno et al	N	Y	Y	Conservation laws
C.O'Sullivan et al	N	N	N	Conservation of momentum and energy
M. Moore et al	Y	N	N	Conservation of linear and angular momentum
P. Volino et al	Y	Y	Y	Combination of elastic model and the mass-spring model
B. Geiger et al	Y	N	N	Complaint Motion
D. Baraff et al	N	Y	Y	Constraint
J. Mezger et al	N	Y	Y	Coherence between consecutive frames
B. Mirtich et al	Y	N	N	Impulse based
P. Sovis et al	Y	Y	N	Impulse dynamics
F. Policarpo et al	Y	N	N	Impulse dynamics
P. Volino et al	Y	N	Y	Impulse based
T. Kurihara et al	N	Y	N	Impulse dynamics
V. Vuskovic et al	N	Y	N	Finite element modeling
C. Basdogan et al	Y	Y	Partially	Hybrid of finite element model and particle system
O. Etzmuß et al	Y	Y	N	Add/subtract virtual particles
C.O'Sullivan et al	Y	Y	N	Human perception of collision
B. Lee et al	Y	Y	N	Fields of displacement

2.2 CR: Overcoming Interpenetrations

The above-described algorithms give us the different collision response schemes tried by the researchers. But all the collision detection schemes are discrete-time and hence, the algorithm may miss the exact moment of collision i.e. the first contact point. The collision response algorithm has to determine that the moment of collision is missed and take measures to resolve the interpenetrations. The following algorithms are aimed at resolving the interpenetrations.

The main problem in getting the collision response is that the penetrating depth is unknown. Consistent penetration depth can be calculated by the algorithm suggested by B. Heidelberg [16]. The main drawback of this algorithm is that we require 3D tetrahedral modeling of the objects so as to get the penetrating depth. In our system, we have decided to stay with 2D surface meshing for objects. Hence, this algorithm will not work for our system.

Another approach suggested by D. Baraff [4] is used to untangle cloth during animation. The Global Intersection Analysis finds the portions of the two objects that are interpenetrating each other and then applies attractive forces between particles of different objects so that the interpenetration is resolved. The intersection areas are found by using path following and flood fill algorithms which are computationally expensive. This approach was thought to be computationally too expensive at the moment and hence was not implemented in our collision response module.

CHAPTER 3

THEORETICAL BACKGROUND FOR COLLISION DETECTION AND COLLISION RESPONSE

The Collision Detection algorithms are prominent in a large number of applications like robotics, computer graphics, automation, computer-aided design, surgical simulation and other simulated virtual environments. In the following sections, we provide a theoretical background for collision detection and collision response.

3.1 Model Representations

There are many ways in which models can be represented in 3D graphics. One possible classification is shown in Figure 3.1.

Broadly, the 3D models are classified [25] as:

- Non-polygonal Models
- Polygonal Models.

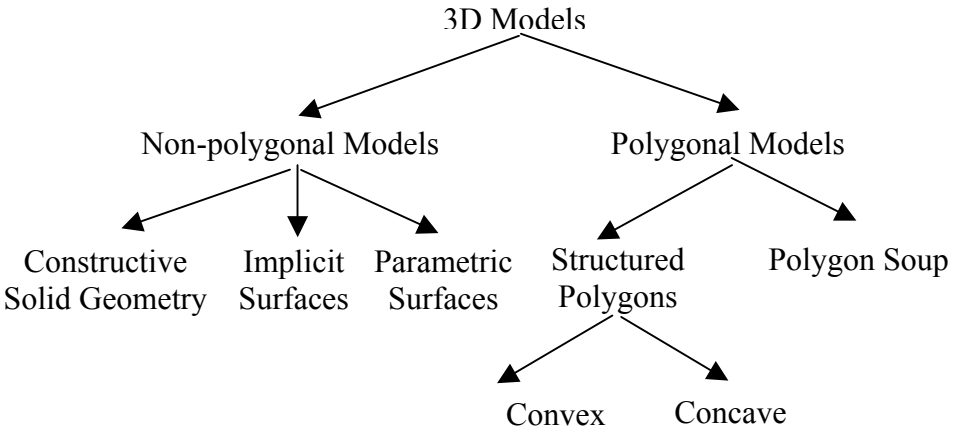


Figure 3.1 Classification of 3D model representations [25]

3.1.1 Non-polygonal Models

Constructive Solid Geometry (CGS) representations are used in Computer Aided Design (CAD) models. These models are formed by using set operations like union and intersection. The main difficulty is that an accurate boundary or surface representation, which is useful for rendering, can be hard to compute from these representations.

Implicit surfaces are those that are defined by mathematical functions. It is difficult to represent objects even using simple primitives like spheres or spline surfaces. A special case of implicit surfaces are *quadrics*, which are second-degree polynomials in x, y and z, which are widely used in OpenGL (a registered trademark of Silicon Graphics Inc.).

Parametric surfaces are defined by parametric functions. Non-uniform rational B-splines (NURBS) are the most commonly used parametric surfaces.

3.1.2 Polygonal Models

Polygonal models are the most commonly used representation in computer graphics. They are simple to implement. Polygon soup is a general collection of polygons that are not necessarily connected and has no topological information available. Structured polygons consist of polygons having specific relationship to one another. A convex polygon contains all the line segments connecting any pair of its points, while a concave polygon does not. A couple of 3D triangular meshes are shown in Figure 3.2.

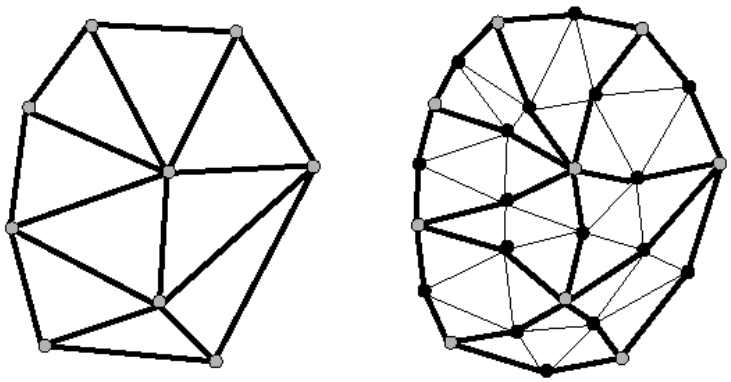


Figure 3.2 3D triangular mesh

3.2 Mass-Spring Model

The notion of mass-spring model has been borrowed from the field of mechanics, which has been used in modeling of thin membrane such as shell structure and dynamic analysis. In the last ten years, this approach has also been used in animation field to simulate the behavior of cloth and other thin objects [35]. This notion has been applied to haptic rendering recently [8]. The surface boundaries are divided into triangular meshes. Each vertex of the triangular mesh is assigned a mass while each

edge acts as a spring. The main reasons for which we have chosen a mass-spring damper (MSD) model are:

- Simplistic approach and well defined dynamics
- Easy to construct the continuous medium as discrete medium, for the modeling of deformable bodies
- Enables achieving haptic feedback rates
- Relatively low computational costs

The mass-spring model consists of a series of particles connected by simulated springs. An example of hexahedral 3D mesh with mass-spring model is given in Figure 3.3.

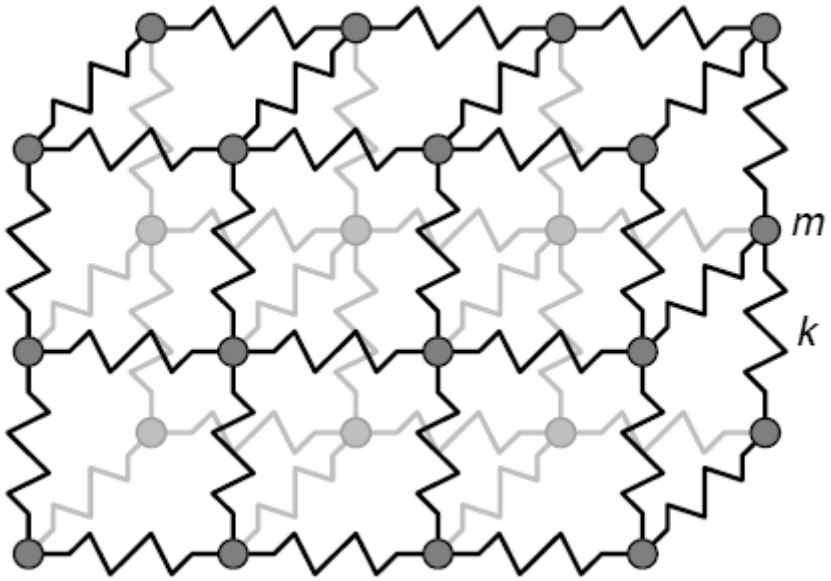


Figure 3.3 Mass-spring model [51]

The springs connecting point masses exert forces on neighboring masses [51]. The spring forces are often linear but nonlinear springs can be used to model tissues

such as human organs that exhibit inelastic behavior. In a dynamic system, Newton's Second Law governs the motion of each mass point in the lattice:

$$m_i \ddot{x}_i = -\gamma_i \dot{x}_i + \sum_j f_{ij} + F_e \tag{3.1}$$

where m_i is the nodal mass, $x_i \in R^3$ is its position, f_{ij} is the force exerted on mass i by the spring between masses i and j and F_e is the sum of external forces (e.g. gravity or user applied forces) acting on mass i. Note that if we let the left side of the equation be zero, the dynamic equation becomes a static equation and the corresponding system becomes a static system.

Based on the motion equation of each point, we can obtain the motion equation for the entire lattice system:

$$M\ddot{\vec{X}} + C\dot{\vec{X}} + K\vec{X} = \vec{F}_e \tag{3.2}$$

where M, C and K are 3N by 3N mass matrix (N is the number of point masses), damping matrix and stiffness matrix respectively. Note that M and C are diagonal and K is banded.

The above second-order equations can be converted to first-order equations for the convenience of analysis or integration.

$$\begin{aligned} \dot{\vec{U}} &= -M^{-1}C\vec{U} - M^{-1}KX + M^{-1}\vec{F}_e \\ \dot{\vec{X}} &= U \end{aligned} \tag{3.3}$$

The mass-spring model is a simple model with a well-understood dynamics. It has a small computation burden and is suitable for real-time applications. Since mass-spring system has a simple structure, all kinds of operations including cut in surgery can be handled easily.

The drawbacks of the mass-spring model are listed below.

- The mass-spring model is a discrete approximation of the true organs
- Proper values for the constants of the mass-spring model are not easy to specify
- Certain constraints are not easily expressed in the model
- Numerical instability phenomena often occur [2]
- Physical accuracy is often lacking

The mass-spring model has been widely used in facial animation (both static and dynamic, two dimensional and simple three dimensional [33] [44] [54]). It also has been used for cloth simulation in game development, and several methods have been suggested to avoid numerical instability [3] [10].

To improve the mass-spring model, much research work has been done on various aspects, i.e. to refine the mass-spring system adaptively [17], to update the constants after refinement [56] and to control the isotropy or anisotropy of the material [6], etc.

J. Brown and S. Sorkin et al [7] developed a simple but efficient algorithm based on the mass-spring model. This algorithm took advantage of the local nature of the deformations to reduce calculations by using a "wave-propagation" technique that has automatic computation cutout when deformations become insignificant. Using this algorithm, they achieved an update frequency of 30Hz for the deformations in suturing vessel surgery, which is compatible with real-time graphic animation. But to provide realistic force feedback, we need to update the forces at a frequency of 1000Hz. This

problem also exists for other methods. Although we can use a certain interpolation method to make up the gap, we are not clear about how serious the side effect of interpolation is. In fact, some research work has been done to study human haptics [43]. It shows that human somatosensory system can perceive vibrotactile stimuli up to 1000Hz. Although the force control bandwidth for human is only 20-30Hz, it is still important for the simulator to provide high frequency force feedback to make the simulation realistic. If the simulator cannot provide the high frequency force, people will feel that the simulation somewhat dull.

Recently, Wang et al [53] have proven that triangular meshes are better for mass-spring model than rectangular meshes as the triangular mesh gives a better response in the case where there is bending force. He also showed that the results with preload are much better than the one without preloaded springs.

3.3 Collision Detection - OHC Algorithm

All the objects in the virtual surgery environment are represented using polygonal models. The lowest level primitive is a triangle. Based on these, the requirements of the collision detection algorithm are:

- *Robustness*: The algorithm should always result in accurate collision points and should not go into infinite loops or crash.
- *Scalability:* As the number of primitives in the environment increase, the performance of most algorithms degrades. The aim is to decrease the number of intersection tests in each time step to maintain a constant query time (total time taken by the collision detection algorithm).

- *Time-critical:* This requirement guarantees an upper bound for the query time of the algorithm.
- *Independence from Geometry:* The algorithm should work for both convex and concave objects.

The *Occupancy* test based algorithm, with a new *Hashing* and *Cascaded* structure (OHC) proposed and implemented by Shen [39] which deals with the above complex factors in the detection context without introducing excessive complexity to the detection scenario itself, is the algorithm chosen for our surgical simulator. This algorithm is a conservative algorithm, which has very low query time and is discrete in nature. Currently, the algorithm gives one of the best ratios for *Query time Vs Actual Output Size.* It takes a few milliseconds for detecting and reporting hundreds of colliding pairs. This narrow phase of the algorithm is based on the subdivision of the 3D space into cells. The cells are of equal size and shape. Then the objects or their parts in each cell are identified. Then the algorithm traverses each cell and checks for collisions in the cell. The cell size is a very important parameter. If the cell size is too big, then the number of primitives needed to be checked for intersection increases and if the cell size is too small, then the memory used for storing the same information is too large and parts of the primitives are present in a large number of cells. The comparison of different collision detection methods is given in Table 3.1 [39].

Table 3.1 Comparison of various Collision Detection Algorithms [39]

	Bounding Volume Hierarchy Method	*Common Spatial Tessellation Method*	*OHC Method*
Broad phase detection at object level	Pairwise	Non-pairwise	Non-pairwise
Broad phase detection at sub-object level	Pairwise	Pairwise	Non- pairwise
Hierarchy complexity	Multi-level	1 level	3 level
Data storage and access	Efficient/Inefficient	Inefficient	Efficient
Topological complexity	Dependent	Independent	Independent
Applicable to large number of moving objects	No	Yes	Yes
Applicable to large scale environment	Yes	No	Yes

3.4. Collision Response

Once the collisions are detected, then the remaining job is to get the appropriate properties like forces, velocities, positions of the colliding surface vertices and then to apply the properties to the deformable and geometric model. The response of different objects in the same scenario having the same collisions will be different because of the different nature of the objects. Hence, we need to classify the collision response into various cases. In our system, we have the following classification of collisions [52]:

- Instrument to Instrument
- Instrument to Rigid Body
- Instrument to Tissue

- Tissue to Rigid Body
- Tissue to Tissue

The first two come under the category of *Non-deformable Object vs. Non-deformable Object* collision. The third and the fourth come under the category of *Non-deformable Object vs. Deformable Object* while the last one comes under the category of *Deformable Object vs. Deformable Object*.

3.4.1 Instrument to Instrument

For the collision between instruments, we do triangle-to-triangle intersection test.

If one projected interval includes the other i.e. the second triangle penetrates into the first triangle as shown in Figure 3.3 (a), we check the depths of the three vertices of the second triangle relative to the first triangle. One of the depths should be positive. Let the vertex be *A2* and the penetration depth be *d*. For the first triangle, we check the distance between the three vertices and *A2*. Let the closest vertex be *A1*. Then we retrieve the velocity $\vec{V}_2$ of vertex *A2* and the velocity $\vec{V}_1$ of vertex *A1*. Assume the unit outer normal of the first triangle is $\vec{n}$. The reaction force on *A2* is calculated as follows:

$$\vec{F} = (kd - \mu(\vec{V}_2 - \vec{V}_1) \bullet \vec{n})\vec{n} \tag{3.4}$$

For the first triangle, this force is exerted in the opposite direction on vertex *A1*.

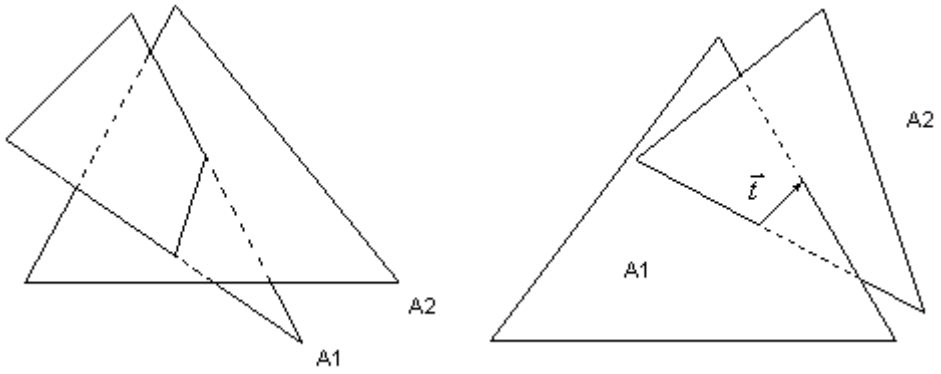


Figure 3.4 Type of intersection of triangles for instrument to instrument case

If neither of the two projected intervals includes another i.e. the two instruments intersect on some edge as shown in Figure 3.3 (b), we need to find the closest vertex pair *(A1, A2)* of the two triangles. Let the length of the overlapped interval be *d*. Assume the unit direction vector for the projected interval of the second triangle on the intersection line to be $\vec{t}$. Then the reaction force on *A2* is calculated by

$$\vec{F} = kd\vec{t} \tag{3.5}$$

For the first triangle, this force is exerted in the opposite direction on vertex *A1*.

Finally all the force on the vertices of the instrument should be converted into a resultant force on the stylus of the PHANToM device. The elastic constant should be large enough to make the instrument rigid.

3.4.2 Instrument to Rigid Body

The instrument to rigid body collision is very similar to the one described above. The main difference is that we do not apply any force to the rigid body, as it is static. The GHOST SDK is a software development kit associated with the Sensable

Technology haptic devices. It gives us the response for the tip of the instrument by using the graphics hardware. Hence, we have not modeled the instrument tip separately.

3.4.3 Instrument to Tissue

For the collision between an instrument and tissue (a deformable object) it is necessary to give a more accurate force or displacement adjustment to both the instrument and the deformable object.

If the projected interval of the deformable object includes the projected interval of the instrument i.e. the triangle of the instrument penetrates into the triangle of the deformable object as seen in Figure 3.4 (a), we check the depth of the vertices of the instrument triangle relative to the triangle of the deformable object. One of them should be positive. Assume that the positive depth is *d* and the corresponding vertex is *A1*. For the triangle of the deformable object, we check the distance between the three vertices and *A1*. Let the closest vertex be *A2*. We retrieve the velocity $\vec{V}_1$ of vertex *A1*, the velocity $\vec{V}_2$ of the vertex *A2* and the unit normal $\vec{n}$ of the triangle of the deformable object. The reaction force on vertex *A1* of the instrument is calculated as follows

$$\vec{F} = (kd - \mu(\vec{V}_2 - \vec{V}_1) \bullet \vec{n})\vec{n} \tag{3.6}$$

where *k* is the average elastic constant of the springs connected to *A2*. In this way, the user can feel the surface of the deformable object with the instrument.

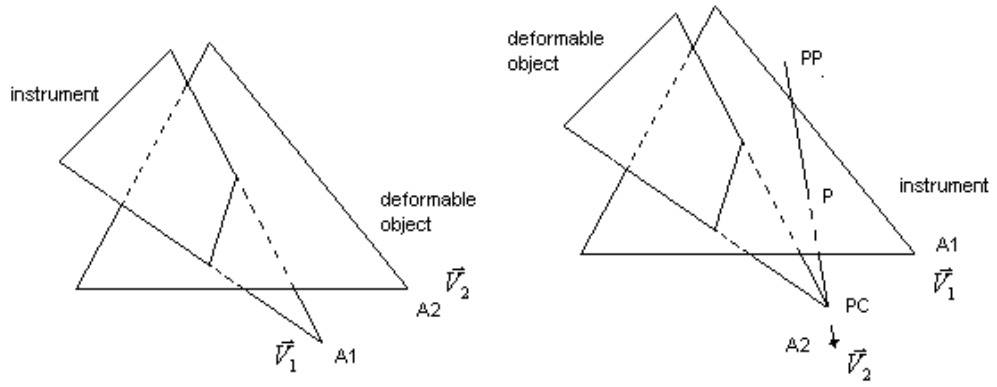


Figure 3.5 Type of intersection of triangles for instrument to tissue case

This force is exerted in the opposite direction on vertex A2 of the triangle on the deformable object.

If the projected interval of the instrument includes the projected interval of the deformable object i.e. the triangle of the deformable object penetrates into the triangle of the instrument as shown in Figure 3.4 (b), we check the depth of the vertices of the deformable object triangle relative to the triangle of the instrument. One of them should be the positive. Let the vertex be *A2*. Assume the previous position of vertex *A2* to be *PP*, the current position to be *PC*, and then the contact position should be the intersection point $\vec{P}$ of the line *PP-PC* with the triangle of the instrument. We force the current position of vertex *A2* of the deformable object to be $\vec{P}$. Let the closest vertex on the triangle of the instrument be *A1*. We retrieve the velocity $\vec{V}_1$ of vertex *A1*, $\vec{V}_2$ of vertex *A2* and the outer unit normal $\vec{n}$ of the triangle of the instrument. Now, if the relative velocity of the two triangles is negative i.e. they are trying to penetrate more into each other, we should give them a velocity in the opposite direction. Hence, if,

$$v_R = \vec{V}_2 \bullet \vec{n} - \vec{V}_1 \bullet \vec{n} < 0 \tag{3.7}$$

the velocity of *A2* is changed to,

$$\vec{V}_2' = \vec{V}_2 - v_R \vec{n} \tag{3.8}$$

In this way, the mass of the deformable model will not enter the instrument. The reaction force to the instrument is calculated through the equilibrium condition of the vertex *A2* and exerted on *A1*.

If neither of the projected intervals includes the other, and they just overlap, the deformable object and the instrument intersect each other on some edge, similarly as in Figure 3.3 (b). We need to find the closest vertex pair *(A1, A2)* of the two triangles. Let the length of the overlapped interval be *d*. Assume the unit direction vector for the projected interval of the second triangle on the intersection line to be $\vec{t}$. Then the reaction force on *A2* is calculated by

$$\vec{F} = kd\vec{t} \tag{3.9}$$

where *k* is the average elastic constant of the triangle of the deformable model. The force exerted on vertex *A1* of the deformable object is $-\vec{F}$.

3.4.4 Rigid Body to Tissue

For the collision between a bone and a deformable object, we can also do a triangle-to-triangle test.

If the projected interval of the bone includes the projected interval of the deformable object as shown in Figure 3.5, we check the depth of the three vertices of triangle on the deformable object relative to the triangle on the bone. We choose the mass with positive depth. Then we retrieve the velocity $\vec{V}$ of the mass of the

deformable object, the outer unit normal $\vec{n}$ of the colliding triangle of the bone. Let the previous position of the mass be *PP* and the current position be *PC*. The collision position *P* on the triangle of the bone is the intersection point of the line *PP*-*PC* with the triangle on the bone. If the relative velocity is negative i.e. the two objects are trying to penetrate into each other, we reverse the velocity direction. The relative velocity is given as,

$$v_R = \vec{V} \bullet \vec{n} < 0 \tag{3.10}$$

we make the velocity of the mass be

$$\vec{V}' = \vec{V} - v_R \vec{n} \tag{3.11}$$

and the position of the mass be *P*. In this way, the mass of the deformable model will not enter the bone.

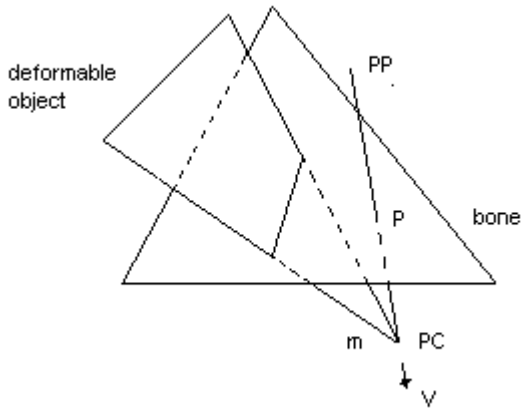


Figure 3.6 Type of intersection of triangles for rigid body to tissue case

We do not consider the cases in which the rigid body triangle penetrates into the tissue triangle, or the two triangles intersect on the edge, as the case considered above is the most important.

3.4.5 Tissue to Tissue

When there is a collision detected between deformable objects, we simplify the interactions between the bodies as two masses.

For each of the intersection pairs of triangles, we choose the closest vertex pair of the two triangles to represent the two objects. Then for each pair of the vertex, we retrieve the velocity $\vec{V}_1$ of the mass m_1 of the first deformable object that corresponds to the collision, the velocity $\vec{V}_2$ of the mass m_2 of the second deformable object.

The relative velocity of mass m_2 to mass m_1 as shown in Figure 3.6 is,

$$\vec{V}_R = \vec{V}_2 - \vec{V}_1 \tag{3.12}$$

Assume there is no energy loss during the collision and the collision direction is in the direction of $\vec{V}_R$.

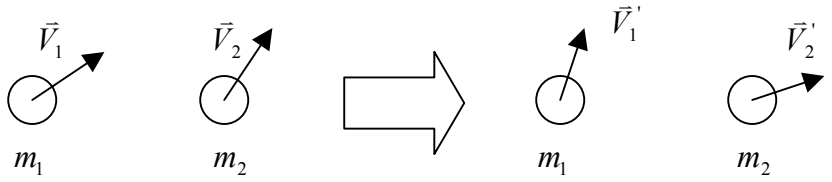


Figure 3.7 Collision of two elastic spheres

Let

$$\vec{n}_R = \frac{\vec{V}_R}{\| \vec{V}_R \|} \tag{3.13}$$

then

$$V_{1R} = \vec{V}_1 \bullet \vec{n}_R \tag{3.14}$$
$$V_{1R} = \vec{V}_1 \bullet \vec{n}_R$$

We have

$$m_1 V_{1R} + m_2 V_{2R} = m_1 V_{1R}^{'} + m_2 V_{2R}^{'} \tag{3.15}$$

$$m_1 V_{1R}^2 + m_2 V_{2R}^2 = m_1 V_{1R}^{' 2} + m_2 V_{2R}^{' 2}$$

From the two equations above we can obtain

$$V_{1R}^{'} = \frac{(m_1 - m_2)V_{1R} + 2m_2 V_{2R}}{m_1 + m_2} \tag{3.16}$$

$$V_{2R}^{'} = \frac{(m_2 - m_1)V_{2R} + 2m_1 V_{1R}}{m_1 + m_2}$$

Then

$$\vec{V}_1^{'} = \vec{V}_1 - V_{1R}\vec{n}_R + V_{1R}^{'}\vec{n}_R \tag{3.17}$$

$$\vec{V}_2^{'} = \vec{V}_2 - V_{2R}\vec{n}_R + V_{2R}^{'}\vec{n}_R$$

Amongst all these cases, the first three are more significant. Collisions of the last two types are quite few and also the reactions are not very significant and hence, they are ignored. We will go into the details of all the type of collision response, but we have implemented only the first three types.

The collision detection algorithm is a discrete-time algorithm due to which the first point of interpenetration of the objects may be missed, but we will know that there is interpenetration or collision. Hence, the collision response algorithm must correctly change parameters so that the interpenetrations are resolved. This is mostly evident in the *instrument to tissue* case. We do this by projecting the velocity of the instrument in the opposite direction and finding out the vertices through which the instrument has passed. Now, attractive forces are applied depending on the *penetrating depth* of the

instrument vertex with respect to the tissue vertex through which it has passed. An equal and opposite force is applied to the instrument which would simulate the reaction force on the instrument.

CHAPTER 4

IMPLEMENTATION DETAILS

In this chapter, we describe the various algorithms implemented for Collision Response. We will start with an overview of the existing system in the Virtual Environment Laboratory (VEL).

4.1 Framework of the Surgical Simulator

The framework of the surgical simulator is a real-time application that is capable of simulating complete surgery. It communicates intensively with the operating system for better performance. The students work towards improving portions of the existing software of the simulator, which have to be integrated into the framework. This integration is done with the help of Yunhe Shen, who designed and implemented the framework.

The components of the framework can be classified as:

- *CGeoModel*
- *CHaptic*
- *CDeformation*
- *CCollision*
- *CGraphics*
- *CDeformable*
- *CSetDialog*
- *CPerformanceMonitor*

- *CMI*
- *CShare*
- *CColliResp*

The *CGeoModel* structure uses the polygonal data, the triangles representing the tissue surface. This structure has the functions required to load data from the VRML files of the instruments and tissues and store in the corresponding *CGeoModel* instances. The structure has all the connectivity required for defining deformable data in space. It has the information about the texture of tissues obtained from offline processing.

The *CHaptic* structure can be used to create an instance of the PHANToM haptic device. This structure can be used to change the current position, orientation and rotation of the haptic device, which results in a change in the force on the instrument.

The *CDeformation* defines a separate thread for deformable modeling of the tissue.

The *CCollision* defines a separate thread for collision detection. The collision detection and response can be classified into:

- Instrument to instrument collision
- Instrument to tissue collision
- Instrument to rigid body collision
- Tissue to rigid body collision
- Tissue to tissue collision

The *CGraphic* structure has calls to OpenGL, which is a standard graphics library that implements real-time texture rendering and the special visual effects.

The *CDeformable* class encapsulates the deformable modeling algorithms. This implements the physical model of the mass-spring system.

The *CSetDialog* structure gives the various settings for the simulator, which varies the parameters ranging from force constant to the tissue or the instrument to be loaded into the scene.

The *CPerformanceMonitor* helps to monitor the efficiency and the speed of simulation.

The *CShare* and CMI structures synchronize and interface, respectively, the many classes and data, which are widely distributed in the simulation.

The *CColliResp* class defines the variable and the functions required to achieve collision response. We will talk about this in detail later on.

Different modules or classes require different amount of memory and processing power. To achieve this, different threads, which contain different classes are given different priorities. There are four main threads running in the system:

- *CHaptic* – Haptic device thread.
- *CDeformation* – Deformation thread.
- *CCollision* – Collision Detection and Response thread.
- *CGraphics* – Real-time graphics thread.

The Graphics thread has a priority lower than the Operating System as it requires lower update rate, while the remaining three threads have priority equal to the Operating System, as they are time critical.

4.2 Collision Detection

The collision detection algorithm along with the processor and its peripherals that we choose has a very big impact on the type of response that we must provide. If the collision detection algorithm is conservative and the time between consecutive time stamps is sufficiently small, then the collision response may just provide a change in the force, velocity or the position as required. But if the time stamps are too far apart, then the collision response must handle situations in which there are severe interpenetrations. In order to have a very good response, even though we have a conservative collision detection algorithm, we design a collision response algorithm that can handle severe interpenetrations.

4.2.1 OHC: Collision Detection Algorithm

The OHC algorithm implemented in our system by Yunhe Shen [39] is a conservative algorithm, which gives comparable results of *Query time Vs Actual Output Size* ratio to the best algorithms in the literature. The OHC algorithm consists of a broad phase collision detection followed by the narrow phase collision detection. In the broad phase, the objects, which are not colliding are identified and only those that are colliding are passed on to the narrow phase. The narrow phase finds the pair of primitives of the colliding objects that intersect i.e. the collision points are identified. These are then passed to the collision response module.

4.2.2 OHC: Narrow Phase

First, the *Cell Map* is loaded, which is traversed cell by cell in turn. Now, the *Object Map* in the concerned cell is loaded. After this, the *Primitive Map* of the objects in the present Object Map is loaded. This structure is further explained below:

The entire surgery simulator 3D space is subdivided into cells of uniform size and shape. The Cell Map is a structure that stores the arrangement of the cells. The objects are then rasterised into the cells i.e. the part of an object or whole object may lie in one of the cells. This information is stored in the Object Map. Now, we need to know which primitives are in a particular cell. Only identifying the objects in a cell will not suffice as some part of the primitive may be outside of the cell. Hence, primitives of a particular object, which are present in the particular cell is stored in the Primitive Map.

Then, the primitives are checked against one another for intersections using the triangle-triangle intersection test [31]. The primitives that are found to be intersecting are passed on to the *Collision Response* module, which further processes the primitive properties and applies the appropriate force or changes the velocity or the position as desired. This modeling is explained in detail in Section 4.3.

4.3 Collision Response

This module is the result of this thesis work. As we had discussed in Section 3.3, the collision response is classified according to the type of objects that are colliding. Hence, according to the classification we separate the code into the following collision types:

- Instrument to instrument

- Instrument to rigid body
- Instrument to tissue

The flowchart is given in Figure 4.1.

4.3.1 Instrument to instrument collision response

The velocities of the vertices of both triangles are obtained by using the geometric model parameters. This requires locking and unlocking the geometric model, as we need the previous positions and the current positions of the vertices so that we can know the velocities. This synchronization procedure slows the system down a bit as the graphics and haptics cannot access this data while it is locked and hence, they have to wait. This is one area that can be improved upon.

Then, the type of intersection i.e. triangle one inside triangle two, triangle two inside triangle one or edge intersection along with the intersection segment and the depths of the vertices of both triangles from the plane formed by the other triangle are calculated. Depending upon the type of intersection and the depths, the new forces on the vertices, which are penetrating inside, are calculated. Then, this force is distributed using the algorithm described by Volino et al. [48]. This force is then translated into an equivalent force on the tip of the instrument as the PHANToM devices that we use have only three degrees of freedom (DOF). The flowchart is given in Figure 4.2.

4.3.2 Instrument to rigid body collision response

The instrument to rigid body collision is very much similar to the previous case. But in this case, we cannot change the parameters of the rigid body primitives as they

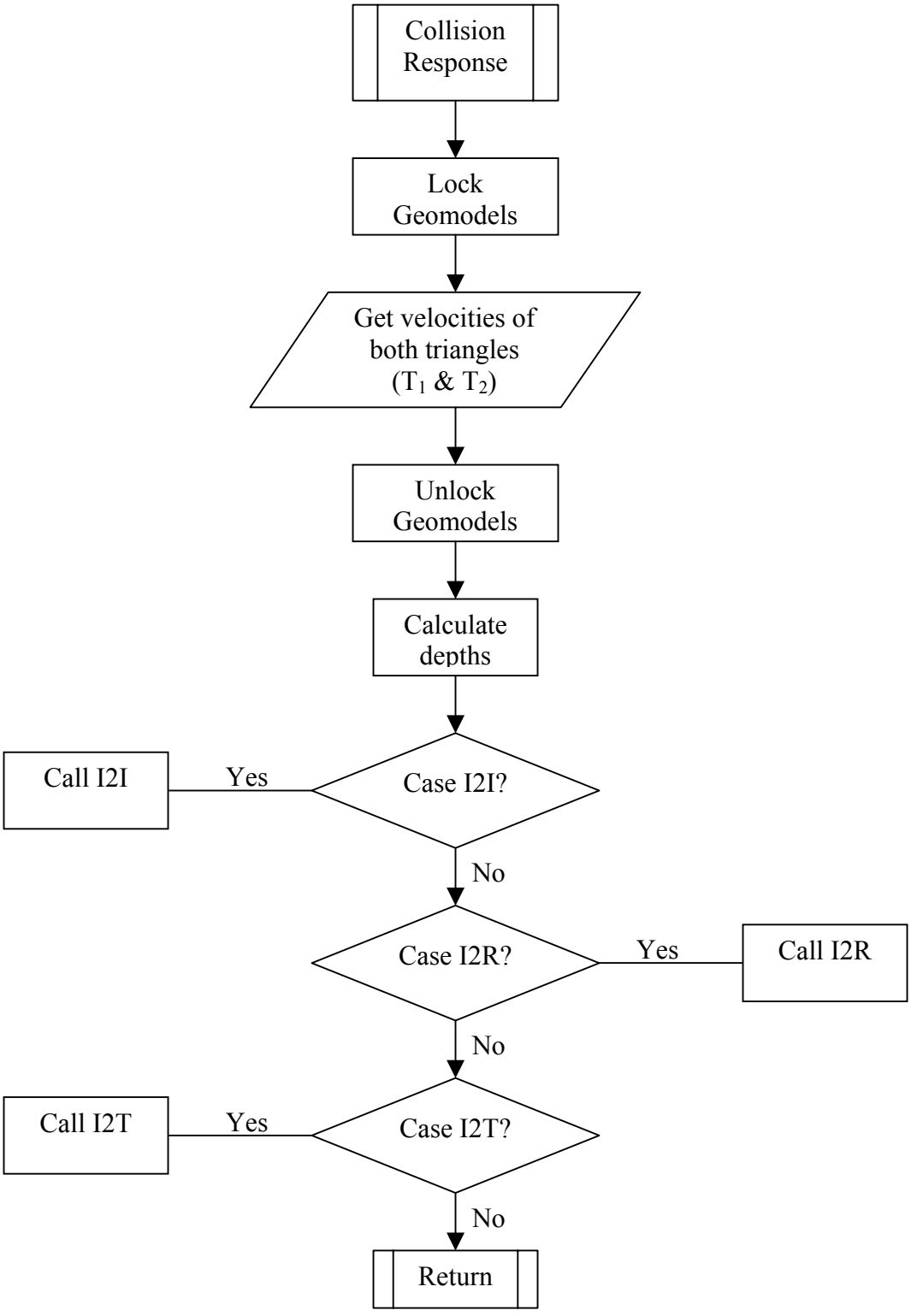


Figure 4.1 Main flowchart for collision response

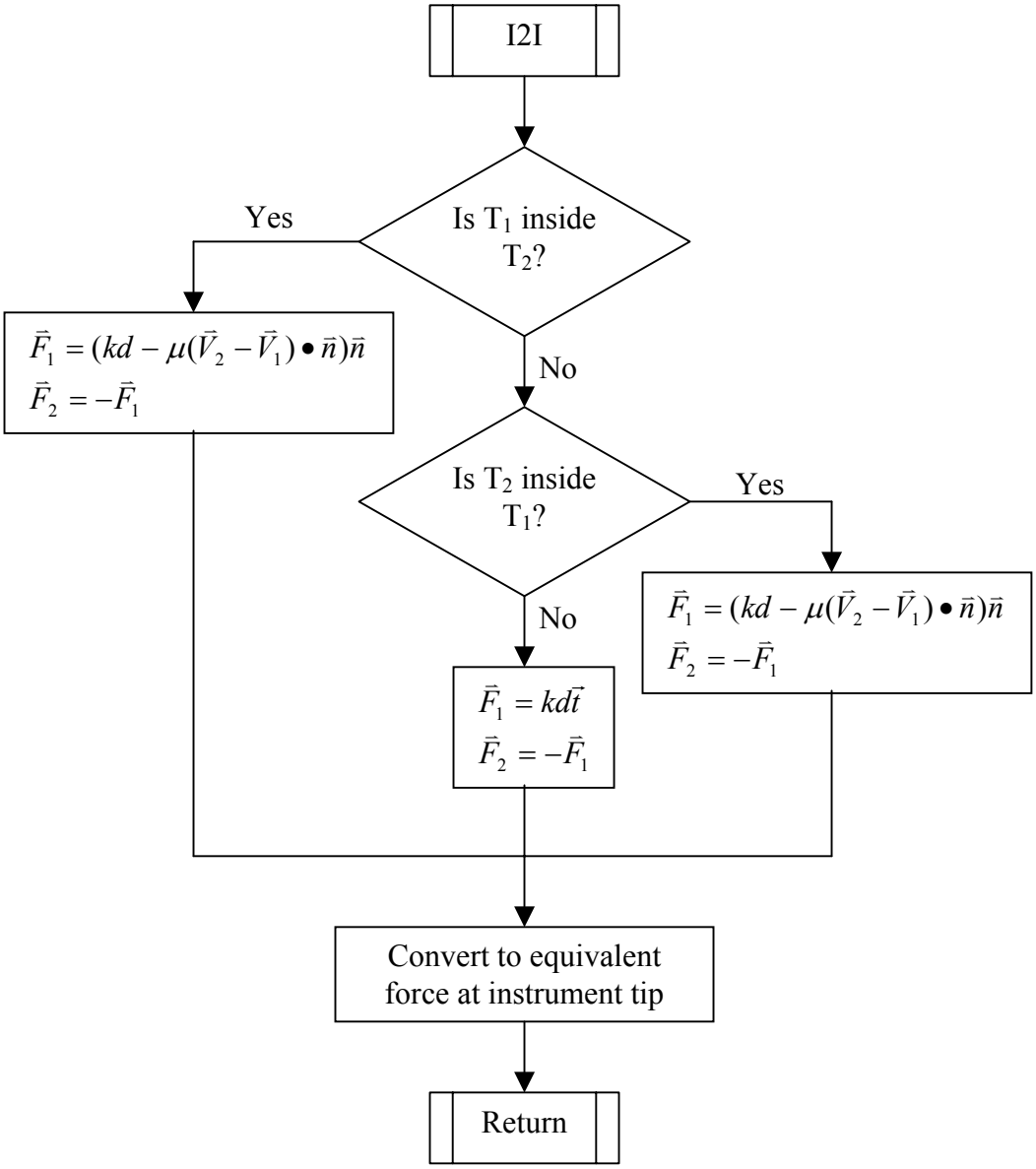


Figure 4.2 Flowchart for I2I case

have definite shape and cannot move. The instrument tip collision detection and response is calculated and the force applied by the GHOST SDK. The flowchart is given in Figure 4.3.

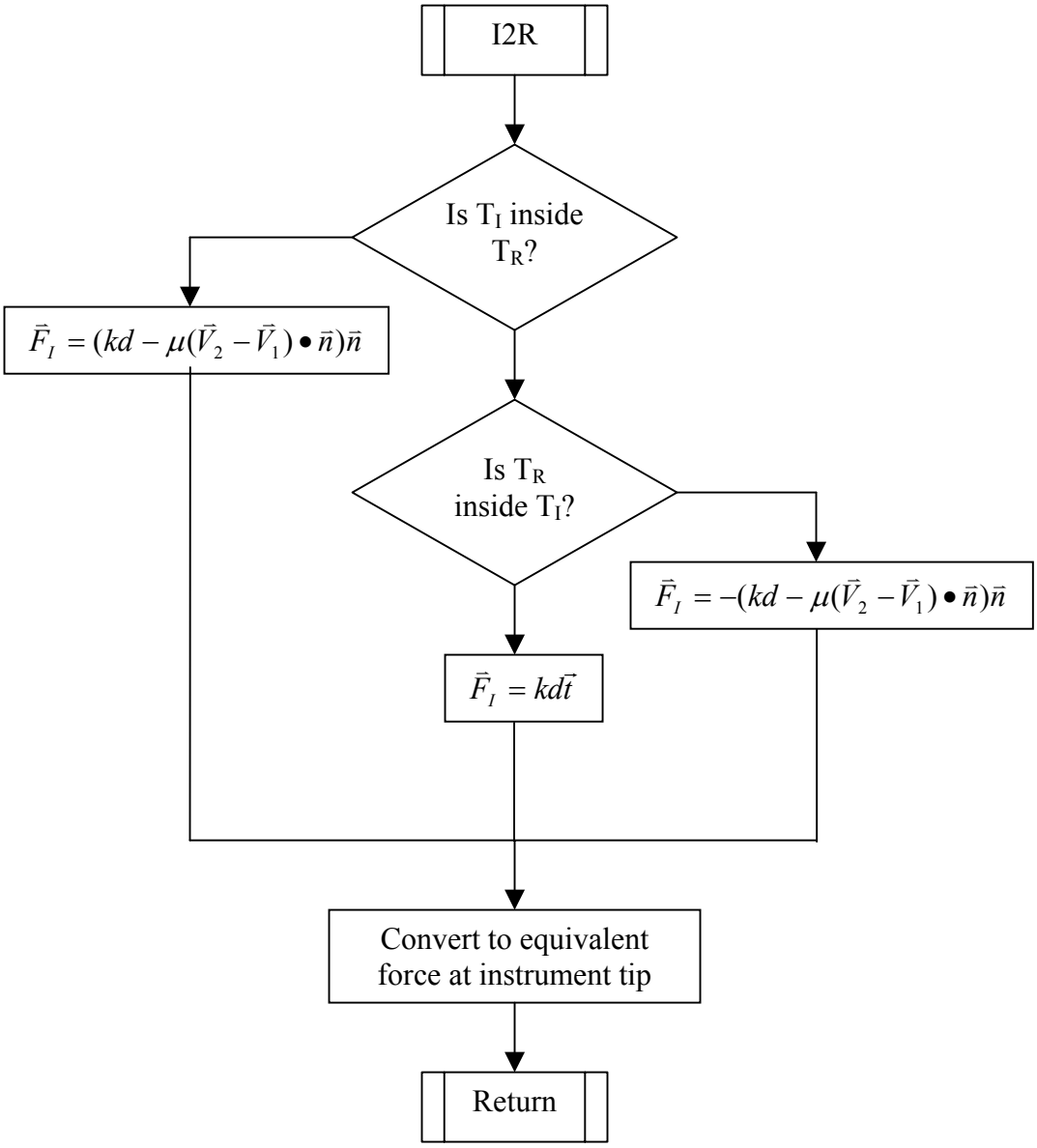


Figure 4.3 Flowchart for I2R case

4.3.3 Instrument to tissue collision response

The instrument to tissue collision response is more complicated than other two cases as this collision involves a deformable tissue and hence special considerations have to be made for it. Also, the primitives of the tissue have to be modified very

carefully as excess change in the parameters could mean that the tissue is stretched beyond what is physically possible. Also, the instrument can grasp the tissue and pull it. This effect also has to be simulated.

First, we get all the parameters of the instrument and the tissue triangles. We get the instrument vertex velocities similar to that in section 4.3.1. Now, we check if the instrument grasps the tissue. If it does, then we make the tissue follow the instrument. If there is no grasping, then the parameters of the instrument and the tissue parameters is modified in a similar way to that in section 4.3.1. In this case also, we use the distribution function to distribute the collision response forces calculated above over all the vertices. The equivalent forces are calculated for the instrument tip and applied in a similar manner to that in section 4.3.1. The flowchart is given in Figure 4.3.

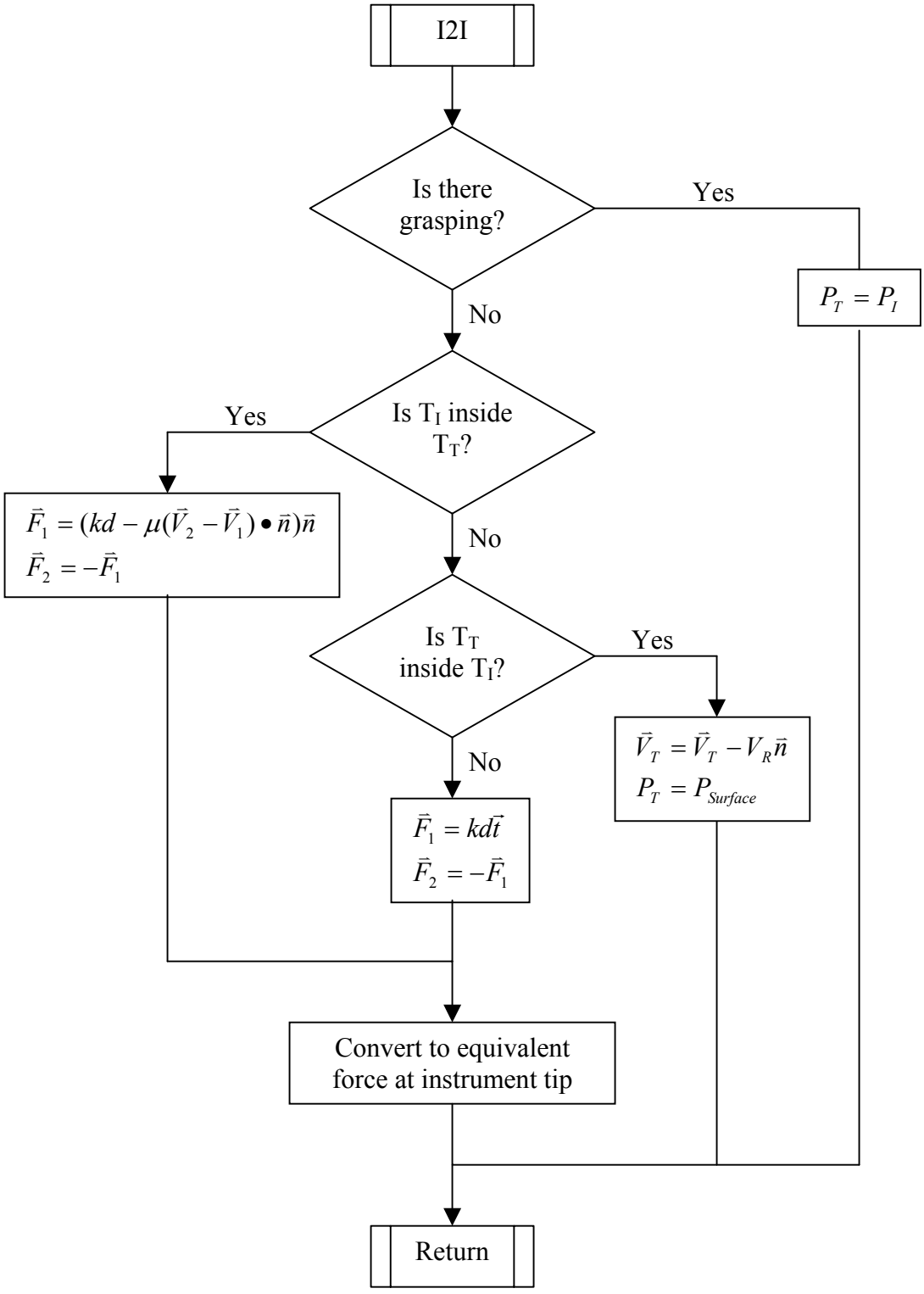


Figure 4.4 Flowchart for I2T case

CHAPTER 5

RESULTS AND FUTURE WORK

In this chapter we present the results of our simulation. We begin with a description of our simulation methods and then follow with statistics like the time taken by the algorithm. After that, we give a few snapshots of the simulation.

5.1 Simulation Details

Our simulator is implemented on a Windows platform, written in VC++.NET with OpenGL graphics, Glut, Glu libraries and MFC user interface. Multi-threading has been used to run simulations in an efficient manner. There are four threads running simultaneously i.e. Main Windows, Collision Detection, Deformation and Haptic threads. Our Collision response algorithm runs in the Collision Detection thread.

The machine that we use for simulation runs has the following specifications and all the results that we have shown are for the same configuration:

- Intel ® Xeon™ dual CPU 2.80 GHz.
- 1 GB RAM.
- Microsoft Windows 2000 SP4.
- Radeon 9700 Graphics Cards – 2.

5.2 Results

This section deals with the results we obtained for the various cases of collision response. The timing diagrams for the three cases of collision response are shown in the following sub-sections. We have provided results for the collision response with and without the triangle-triangle intersection (TTI) test. This is because both the collision detection and collision response modules use the TTI test. A common test is done for both the modules to reduce the computational complexity of the system.

5.2.1 Instrument to Instrument Collision Response

The instrument to instrument (I2I) case has very few collisions as compared to the other cases as the instruments have less number of triangles in the surface mesh. The graphs are given below.

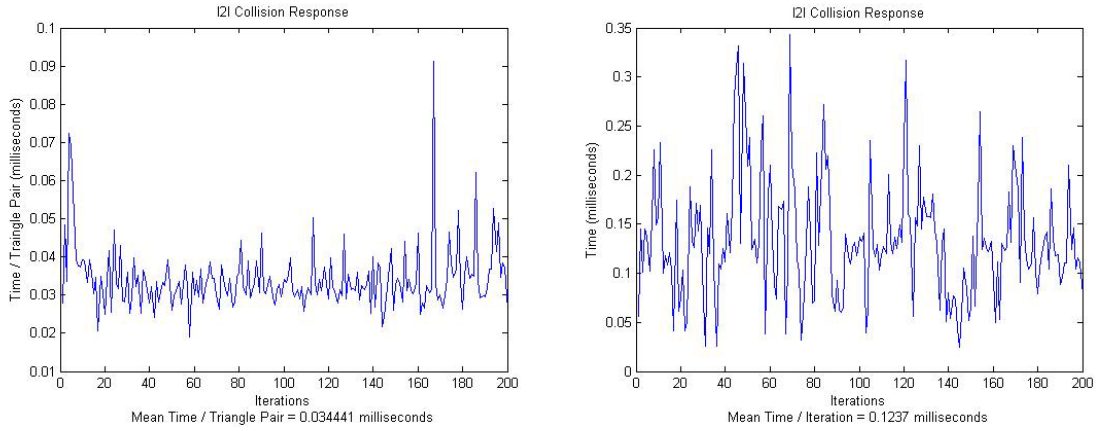


Figure 5.1 I2I collision response without TTI test

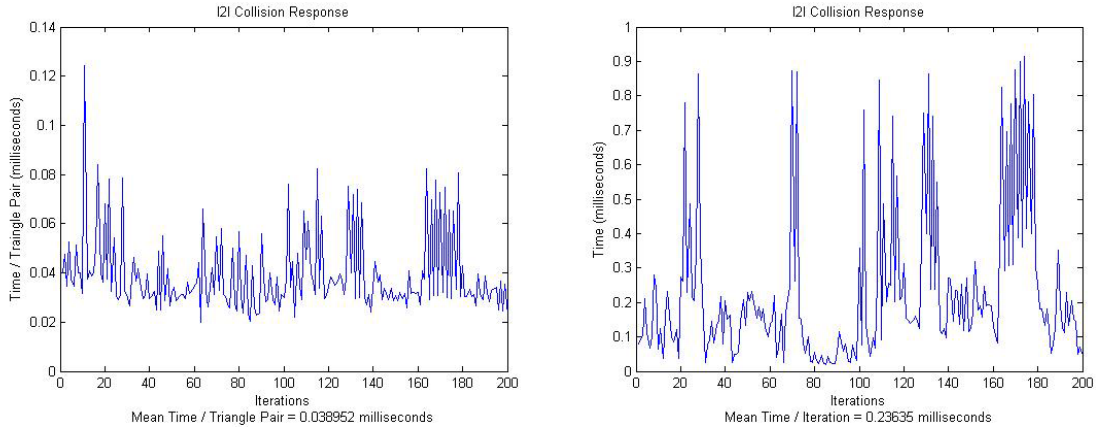


Figure 5.2 I2I collision response with TTI test

The above figures show that the collision response time for I2I without TTI test is about 0.034 milliseconds per triangle pair and 0.124 milliseconds per iteration. The collision response time inclusive of TTI test rises to 0.039 milliseconds and 0.236 milliseconds respectively.

5.2.2 Instrument to Rigid Body Collision Response

The figures below give the collision response times for an instrument to rigid body (I2R) collision response case.

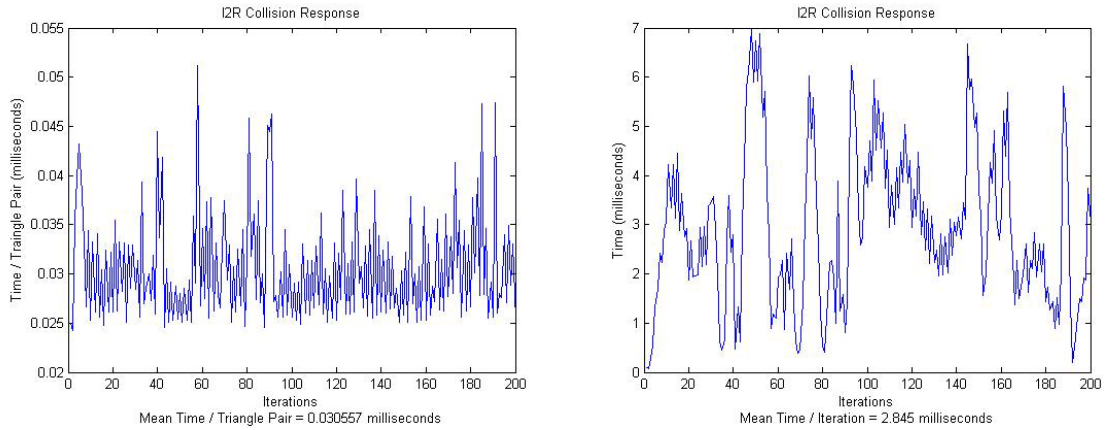


Figure 5.3 I2R collision response without TTI test

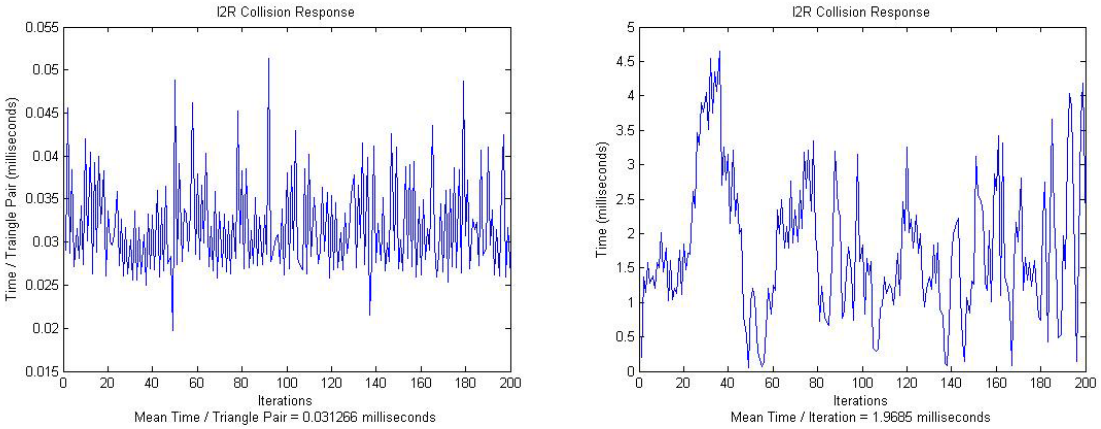


Figure 5.4 I2R collision response with TTI test

The above figures show that the collision response time for I2I without TTI test is about 0.030 milliseconds per triangle pair and 2.845 milliseconds per iteration. The collision response time inclusive of TTI test rises to 0.031 milliseconds and 1.938 milliseconds respectively.

5.2.3 Instrument to Tissue Collision Response

The figures below give the collision response times for an instrument to tissue (I2T) collision response case.

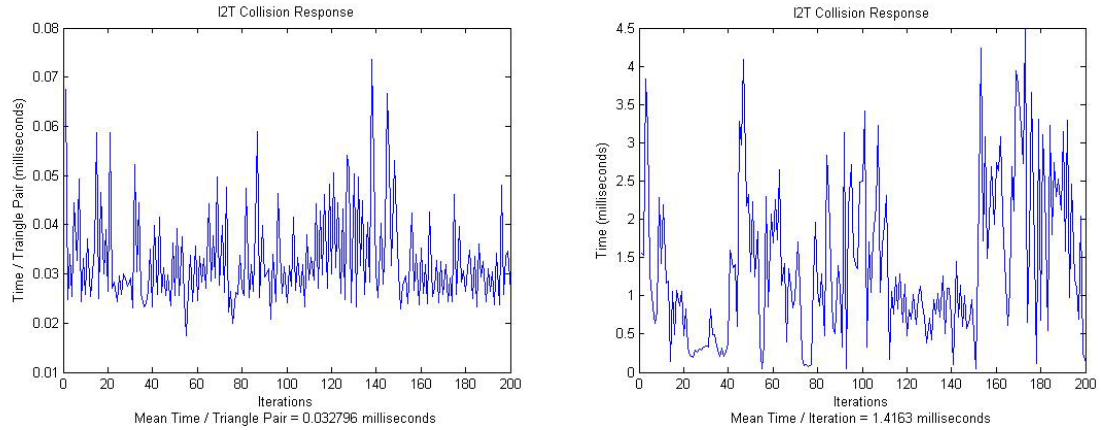


Figure 5.5 I2T collision response without TTI test

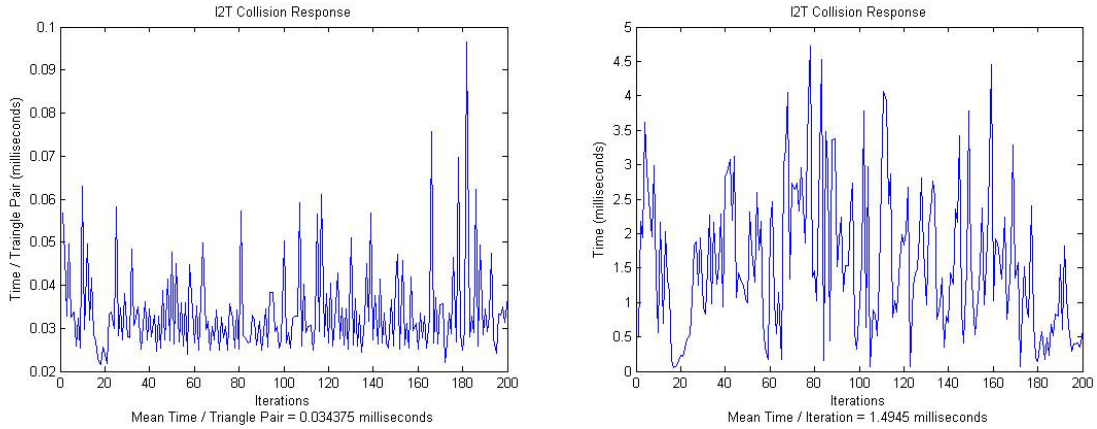


Figure 5.6 I2T collision response with TTI test

The above figures show that the collision response time for I2I without TTI test is about 0.032 milliseconds per triangle pair and 1.416 milliseconds per iteration. The collision response time inclusive of TTI test rises to 0.034 milliseconds and 1.495 milliseconds respectively.

It can be seen from the above figures and the average collision response times that the response calculation is very fast and satisfies the haptic rate.

5.3 Conclusion

Accurate Collision Response calculation is complicated due to the complex tissue behavior and stringent update requirements. We have presented a collision response algorithm that gives very good visual as well as haptic feedback to the user. A few of the advantages of the algorithm that we have implemented are:

- Collision response is classified into different classes and hence the response of one type of collision can be independently changed.

- The algorithm is computationally inexpensive and hence suitable for haptic feedback.
- The visual and haptic feedback of the algorithm is very good.

Thus, we have provided a collision response algorithm that is well suited for virtual laparoscopic surgery simulation.

5.4 Future Work

Presently, we are using 2D surface triangular meshes for representing objects. The collision detection algorithms will always be discrete-time and hence there are always going to be instances in which the collision detection algorithm misses the exact moment of collision. As we are dealing with a real-time system, we cannot backtrack to the point where there was no collision and then continue from there, taking care of the collision. If this was possible, we could have used larger time-steps. But, for a real-time system, the time-steps should be as small as possible so that the collision times are not missed. The solution to the interpenetration caused by the missing collision time is to find the *penetrating depth* of the various vertices of the objects inside each other. At present, the penetration depth is estimated by some calculations. But this would not be necessary if we use 3D volumetric meshing, at least for a few layers beneath the surface. Also, the 3D volumetric meshing would allow the relaxation or the total removal of the volume constraint on the deformable objects for the deformation. This may decrease the computational complexity of the deformation and the collision response algorithms.

The PHANToM devices (Version 1.5) that we use currently have only 3 degrees of freedom for force rendering. If we change the PHANToM devices to Version 3, which provide with 6 degrees of freedom for rendering (force and moment), the collision response may be better felt. This will also reduce the complexity of the collision response algorithm as there will be no necessity to translate the forces on the vertices of the instrument to equivalent forces on the instrument tip.

REFERENCES

[1] J. Amanatides, K. Choi, "Ray Tracing Triangular Meshes," *Proceedings of the Eighth Western Computer Graphics Symposium,* pp 43-52, April 1997.

[2] D. Baraff and A. Witkin, "Dynamic simulation of non-penetrating flexible bodies," *Computer Graphics Proceedings, Annual Conference Series, Proceedings of SIGGRAPH 92,* pp. 303-308, ACM SIGGRAPH 1992.

[3] D. Baraff and A. Witkin, "Large steps in cloth simulation," *Michael Cohen, editor, SIGGRAPH 98 Conference Proceedings, Annual Conference Series,* pp. 43-54, ACM SIGGRAPH Addison Wesley, July 1998.

[4] D. Barraf, A. Witkin, M. Kass, "Untangling Cloth," *ACM SIGGRAPH,* pp. 862–870, 2003.

[5] C. Basdogan, C. Ho, M. A. Srinivasan, **"**Virtual environments for medical training: graphical and haptic simulation of laparoscopic common bile duct exploration," *IEEE/ASME Transactions on Mechatronics,* vol. 6, issue 3, pp. 269-285, September 2001.

[6] D. Bourguignon and M. P. Cani, "Controlling Anisotropy in Mass-Spring systems," *Proceedings of the 11th Eurographics Workshop,* Interlaken, Switzerland, August, pp 21-22, 2000

[7] J. Brown, K. Montgomery, et al, "A microsurgery simulation system," *Fourth International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI 2001),* pp. 137-144, Oct. 2001.

[8] Z. Cai, Dill, S. Payandeh, “Towards deformation modeling with haptic feedback,” *Proc. ASME, Symposium on tele-operation and haptic environment,* pp. 1133-1139, 2000.

[9] E. Colgate, M. Stanley, M. Brown, “Issues in the Haptic Display of Tool Use,” *International Conference on Intelligent Robots and Systems,* vol. 3, pp. 3140, 1995.

[10] M. Desbrun, P. Schroder, A. Barr, “Interactive animation of structured deformable objects,” *Graphics Interface’99 proceedings,* pp. 1-8, 1999.

[11] O. Etzmus, B. Eberhardt, M. Hauth, W. Straser, "Collision Adaptive Particle Systems," *Proc. The Eighth Pacific Conference on Computer Graphics and Applications,* pp. 338, 2000.

[12] B. Geiger, “Real-Time Collision Detection and Response for Complex Environments,” *Computer Graphics International 2000 (CGI'00),* pp. 105, 2000.

[13] A. Gande, “Instructor Station for virtual laparoscopic surgery,” *Masters Thesis,* 2003.

[14] G. Gopalakrishnan, “StapSim: Virtual reality based stapling simulation for laparoscopic herniorraphy,” *Masters Thesis,* 2003.

[15] V. Gupta, “Extraction of realistic anatomical texture from visual human data for laparoscopic herniorraphy,” *Masters Thesis,* 2003.

[16] B. Heidelberger, M. Teschner, R. Keiser, M. Müller, M. Gross, “Consistent Penetration Depth Estimation for Deformable Collision Response,” Proceedings of Vision, Modeling, Visualization VMV'04, Stanford, USA, pp. 339-346, November 16-18, 2004.

[17] D. Hutchinson, M. Preston, et al, "Adaptive Refinement for mass-spring simulations," *7th Eurographics Workshop on Animation & simulation,* pp. 31-45, 1996.

[18] S. Kapdoskar, "Simulation of inguinal hernia condition using visible human data and 3D modeling techniques for virtual laparoscopic heniorraphy," *Masters Thesis,* 2003.

[19] T. Kurihara, K. Anjyo, D. Thalmann, "Hair Animation with Collision Detection," *Models and Techniques in Computer Animation, Springer-Verlag, Tokyo,* pp.128-138, 1993.

[20] M. Kurzer, A. Kark, G. Wantz, "Surgical Management of abdominal wall hernias," *Martin Dunitz Ltd.,* 1999.

[21] M. Kurzer, P. Belsham, A. Kark, "Pre-peritonial mesh repair of recurrent inguinal hernia," *British Journal of Surgery,* vol.89-1, pp.90, January 2002.

[22] T. Larsson, "Collision Detection Algorithms in Virtual Environments," *Invired presentation, SIGRAD*, 2001.

[23] B. Lee, D. Popescu, and S. Ourselin, "Contact modelling based on displacement field redistribution for surgical simulation," *Medical Imaging and Augmented Reality: proceedings of the 2nd International Workshop (MIAR 2004), Beijing, China,* pp. 337-345, 2004.

[24] M. Lin, D. Manocha, Jon Cohen, S. Gottschalk, "Collision Detection: Algorithms and Applications," *Algorithms for Robotics Motion and Manipulation,* pp. 129-142.

[25] M. Lin, S. Gottschalk, "Collision Detection between Geometric Models: A Survey," *Proc. of IMA Conference on Mathematics of Surfaces,* pp. 37-56, 1998.

[26] W. Lorrenson, H. Clin, "Marching Cubes: A high resolution 3D surface construction algorithm," *Computer Graphics,* vol. 21, no. 4, pp. 163-169, July 1987.

[27] M. Moore, J. Wilhelms, "Collision Detection and Response for Computer animation," *Computer Graphics,* vol. 22. no. 4, pp. 289-298, August 1988.

[28] J. Mahovsky, B. Wyvill, "Fast Ray-Axis Aligned Bounding Box Overlap Tests With Plücker Coordinates," *The Journal of Graphics Tools,* vol. 9, no. 1, pp. 37-48, 2004.

[29] J. Mezger, S. Kimmerle, O. Etzmuß, "Progress in Collision Detection and Response Techniques for Cloth Animation," *10th Pacific conference on Computer Graphics and applications,* pp. 444, 2002.

[30] B. Mirtich, J. Canny, "Impulse-based simulation of rigid bodies," *Symp. on Interactive 3D Graphics*, Monterrey, CA, pp. 181, 1995.

[31] T. Möller, "A Fast Triangle-Triangle Intersection Test," *Journal of Graphics Tools*, vol. 2, no. 2, pp. 25-30, 1997.

[32] R. Naidu, "Creation of static and dynamic models of instruments for a virtual reality trainer for laparoscopic surgery," *Masters Thesis,* 2002.

[33] S. Platt and N. Badler, "Animating facial expressions," *Computer Graphics,* 15(3), pp. 245-252, 1981.

[34] F. Policarpo, A. Conci, "Real-Time Collision Detection and Response," *XIV Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'01),* pp. 376, 2001.

[35] X. Provot, “Deformable constraints in a mass-spring model to describe rigid cloth behavior,” *Proc. Graphics Interface,* pp. 147-164, 1995.

[36] R. Radovitzky, M. Ortiz, “Tetrahedral Mesh Generation Based on Node Insertion in Crystal Lattice Arrangements and Advancing-Front-Delunay Trangulation, *Technical Report, Center for Silumation of Dynamic Response of Materials,* 2000.

[37] L.Raghupathi, “Simulation of bleeding and other special effects for virtual laparoscopic surgery,” *Masters Thesis,* 2002.

[38] Y. Shen, “Medical Visualizations and Surgical Simulation,” *Internal report, Virtual Environment Laoratory,* March 2004.

[39] Y. Shen, V. Devarajan, R. Eberhart, “Haptic Herniorrhaphy Simulation with Robust and Fast Collision Detection Algorithm,” *The proceedings of Medicine Meets Virtual Reality,* Long Beach, CA, pp. 458-464, January 2005.

[40] P. Sovis, “Collision Detection and Impulse Dynamics in Real time Applications,” *CESCG,* 2000.

[41] C. O’Sullivan, J. Dingliana, “Real-time collision detection and response using sphere-trees,” *15th Spring Conference on Computer Graphics,* pp. 83–92. Budmerice, Slovakia, April 1999.

[42] C. O’Sullivan, R. Radach, S. Collins, “A Model of Collision Perception for Real-Time Animation,” *Computer Animation and Simulation,* pp. 67-76, 1999.

[43] H. Z. Tan, M. A. Srinivasan, et al, “Human Factors for the Design of Force-Reflecting Haptic Interface,” *Dynamic System and Control,* vol. 55-1, pp. 353-359, 1994.

[44] D. Terzopoulos and K. Waters, “Physically-based facial modeling analysis and animation,” *Journal of Visualization and Computer Animation,* pp. 73-80, 1990.

[45] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, A. Fuhrmann, M. P. Cani, F. Faure, N. M. Thalmann, L. Raghupathi, W. Straßer, P. Volino, “Collision Detection for Deformable Objects,” *Computer Graphics Forum,* no. 1, vol. 24, pp. 61-81, March 2005.

[46] S. Uno, M. Slater, “The Sensitivity of Presence to Collision Response”, *Proc. of IEEE Virtual Reality Annual International Symposium,* pp. 95, 1997.

[47] P. Volino and N. M. Thalmann, "Developing Simulation Techniques for an Interactive Clothing System", *Proc.VSMM'97,* Geneva, Switzerland, pp.109-118, 1997.

[48] P. Volino and N. M. Thalmann, “Accurate collision response on polygonal meshes,” *Computer Animation,* pp. 154, May 2000.

[49] P. Volino and N. M. Thalmann, “Implementing fast Cloth Simulation with Collision Response,” *Computer Graphics International,* pp. 257, June 2000.

[50] V. Vuskovic, M. Kauer, G. Székely, M. Reidy, “Realistic force feedback for virtual reality based diagnostic surgery simulators,” *Proc. ICRA '00, IEEE International Conference on Robotics and Automation,* pp. 1592-1598, 2000.

[51] X. Wang, “Review of deformable models in virtual surgery”, *Internal Report, Virtual Environment Laboratory,* 2004.

[52] X. Wang, “Collision Response”, *Internal Report, Virtual Environment Laboratory,* 2004.

[53] X. Wang, V. Devarajan, "1D and 2D structured mass-spring models with preload," *The Visual Computer,* in print, August 2005.

[54] K. Waters, "A muscle model for animating three-dimensional facial expression," *Computer Graphics Proceedings, Annual Conference Series, Proceedings of SIGGRAPH 87,* pp. 17-24, ACM SIGGRAPH 1987.

[55] K. Yuan, C. Xu, Q. Du, Y. Fu, "Collision Detection for a Haptic Interface," *Proc. International Conference on Robotics, Intelligent Systems and Signal Processing,* Changsha, China, pp. 278-283, October 2003.

[56] J. Zhang, S. Payandeh, et al, "Haptic Subdivision: an Approach to Defining Level-of-detail in Haptic Rendering," *The 10th international symposium on Haptic Interfaces for Virtual Environment and Teleoperator system,* Orlando,FL, pp. 201-208, Mar. 2002.

BIOGRAPHICAL INFORMATION

Jitesh Butala was born on 4th November 1981 in Pune, India to Devidas and Deepali Butala. He completed his schooling from St. Vincents High School, Pune, India. He received his Bachelors degree in Electronics from Vishwakarma Institute of technology, Pune University in August 2003. He received his Masters degree from University of Texas at Arlington in August 2005. He is a sports fanatic. He has represented his state in roller-skating and his school in basketball. He is the recipient of the Rajya Puraskar, one of the highest awards given for scouting in India.