

COLLISION RESPONSE OF SURFACE MODELED ORGANS FOR  
VIRTUAL LAPAROSCOPIC SURGERY

by

DIBBESH SHARMA ADHIKARI

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

MASTERS OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2008

Copyright © by Dibbesh S. Adhikari 2008

All Rights Reserved

## ACKNOWLEDGEMENTS

I am very grateful to my thesis supervisor Dr. Venkat Devarajan for his support, motivation and valuable guidance during the period of study at University of Texas at Arlington. I especially appreciate his time and effort in guiding me whenever I got stuck in my research work.

I also extend my sincere thanks to Dr. Michael T. Manry and Dr. W. Alan Davis for agreeing to be in my thesis supervising committee and for all their help and cooperation.

I would like to thank all of my colleagues at Virtual Environment Lab, especially Sumit, Vishal, Koyel and Rupin for their valuable inputs and help in my research work. I extend my thanks to all my friends and roommates; especially to Shiva Mani for his help during the later stage of the research and helping me to generate the figures during the thesis writing.

I am extremely indebted to my parents and siblings who have encouraged, inspired and supported me at the time when I needed them most. I dedicate this work to them.

November 19, 2008

## ABSTRACT

### COLLISION RESPONSE OF SURFACE MODELED ORGANS FOR VIRTUAL LAPRASCOPIC SURGERY

Dibbesh S. Adhikari M.S.

The University of Texas at Arlington, 2008

Supervising Professor: Venkat Devarajan

Laparoscopic surgery is a modern surgical procedure in which, operations are performed in the abdomen by monitoring the image from a laparoscope: a telescopic rod lens system, which is usually connected to a video camera with a halogen or xenon light source, and is inserted into the abdomen through a small incision. Generally, laparoscopic surgeries are chosen over open surgery because of lower pain and speedier recovery process. However, the surgeon needs additional training in performing such surgery since laparoscopic camera reflects 2D mirror image of hand movement and depth information of the scene is harder for the surgeon to judge. Hence, surgical trainers for laparoscopic surgery are very important. Virtual Reality based surgical simulators in particular are becoming popular.

This thesis describes a virtual reality based surgical simulator and examines the problem of collision response that occurs during the interaction of a surgical instrument with deformable tissue. Collision detection algorithms are discrete-time algorithms and take few

milliseconds on average to detect and report a collision. The processing of multiple collisions detection further complicates the matter. There is also a high chance of missing the exact instant of collision. This research mainly investigates the unseemly interpenetration of the objects caused by such discrete time collision detection. It then proposes an innovative method of using the bounding volume of the object in runtime to find collision information to overcome such interpenetration. The proposed algorithm is implemented and validated in an existing surgical simulator at the Virtual Environment Laboratory.

## TABLE OF CONTENTS

ACKNOWLEDGEMENTS.....	iii
ABSTRACT.....	iv
LIST OF ILLUSTRATIONS.....	ix
LIST OF TABLES.....	xi
Chapter	Page
1. INTRODUCTION.....	1
1.1 Virtual Reality.....	1
1.2 Hernia.....	1
1.3 Herniorrhaphy.....	2
1.3.1 Minimally Invasive Surgery.....	3
1.3.2 MIS: Training Procedures.....	4
1.3.3 VR-based Surgical Simulator.....	4
1.4 Motivation for Laparoscopic Surgery Simulator.....	5
1.5 System requirement for the VR based laparoscopic simulator.....	6
1.6 System Architecture.....	6
1.6.1 Offline processing.....	7
1.6.2 Graphical and Special effects.....	8
1.6.3 Haptic Block.....	8
1.6.4 Real time Module and Challenges.....	9
1.7 Problem Statement.....	10
1.8 Organization of thesis.....	11
2. LITERATURE REVIEW.....	12
2.1 Collision Response .....	12

2.1.1 Constrained based approach.....	12
2.1.2 Penalty based approach.....	13
2.1.3 Impulse based approach.....	14
2.1.4 Other approaches.....	15
2.2 Previous work on overcoming interpenetration.....	18
3. COLLISION DETECTION AND COLLISION RESPONSE.....	20
3.1 Model Representation .....	20
3.1.1 Non polygonal models.....	21
3.1.2 Polygonal models.....	21
3.2 Mass spring model.....	22
3.3 Collision detection.....	25
3.3.1 Bounding Volume Hierarchy (BVH).....	26
3.3.2 Spatial Occupancy Test (SOT).....	26
3.4 Collision Detection: OHC Algorithm.....	27
3.5 Collision Response.....	27
3.6 Instrument to Tissue Collision.....	28
3.6.1 Requirement for collision response algorithm.....	29
3.6.2 Finding a point of collision.....	30
3.6.3 Finding the penetration depth and deformation vector.....	32
3.6.4 Finding the penalty force.....	37
4. IMPLEMENTATION DETAILS.....	39
4.1 Implemented Modules.....	39
4.2 Framework of surgical simulator.....	40
4.3 Implementation details of proposed collision response algorithm.....	42
4.3.1 Finding a point of collision.....	44
4.3.2 Checking the validity of intersection point.....	46

4.3.3 Finding the penetration depth and deformation vector.....	48
4.3.4 Finding a penalty force.....	48
5. RESULTS AND FUTURE WORK.....	51
5.1 Simulation details.....	51
5.2 Results.....	51
5.3 Conclusion.....	54
5.4 Future Work.....	55
REFERENCES.....	56
BIOGRAPHICAL INFORMATION.....	62



## LIST OF ILLUSTRATIONS

Figure		Page
1.1	Direct and Indirect inguinal hernia.....	2
1.2	Steps of repairing inguinal hernia.....	2
1.3	Scene from actual laparoscopic surgery.....	3
1.4	Block diagram of Laparoscopic surgery simulator.....	7
1.5	Steps in creating textured mapped 3D organ from 2D images.....	8
1.6	Phantom device used in the surgery simulator.....	9
3.1	Classification of 3D model representation.....	20
3.2	(a) Surface modeled objects and (b) Tetrahedral model objects with rectangular mesh.....	21
3.3	Mass spring structure of an object .....	22
3.4	Instant of collision between instrument and tissue.....	29
3.5	Different orientations of tissue and instrument triangle during collision.....	30
3.6	Cases where point of collision $N$ or $P$ does not lies on tissue surface.....	31
3.7	Calculation of new point of collision $P$ that lies inside the tissue surface.....	32
3.8	Object positions at collision detection instant at $t_1$ and instant $t_2$ .....	33
3.9	Bounding box of an instrument for the collision detected instant of figure 3.8 (b).....	34
3.10	Three axial penetration depth for the collision detect instant of figure 3.8 (b).....	35
3.11	Different deformation vectors for same penetration depth.....	36

3.12	Case where an Instrument penetrates through tissue surface.....	37
4.1	Main flow chart of collision response algorithm.....	43
4.2	Flow chart of finding a point of collision P.....	45
4.3	Flow chart of checking the validity of point P.....	47
4.4	Flow chart of calculating deformation vector.....	49
4.5	Flow chart of calculating distribute penalty force.....	50
5.1	Graphical response of the proposed collision response algorithm instant t1.....	52
5.2	Graphical response of the proposed collision response algorithm instant t2.....	52
5.3	Graphical response of the proposed collision response algorithm instant t3.....	53
5.4	Timing diagram of the collision detection module.....	53
5.5	Timing diagram of the collision response module.....	54

## LIST OF TABLES

Table		Page
2.1	Comparison of widely used collision response approach.....	16
4.1	Major modules in the surgery simulator.....	39

## **CHAPTER 1**

### **INTRODUCTION**

#### 1.1 Virtual Reality

Virtual Reality, a term coined by Jaron Lanier [1] (founder of Visual Programming Language Research), refers to computer-generated, three-dimensional simulations that allow a participant to experience and interact with a setting or situation [2]. The advantage of virtual reality is that, it can immerse people in an environment that would normally be unavailable due to cost, safety, or perception restrictions. In the most successful virtual reality systems, users feel that they are truly present in the simulated world and that their experience in the virtual world matches what they would experience in the environment being simulated

#### 1.2 Hernia

A hernia is a protrusion of a tissue, which occurs when a portion of the tissue lining the abdominal cavity (peritoneum) breaks through a weakened area of the abdominal wall. Hernia can give rise to discomfort as it enlarges and has potential risk of having the tissue's blood supply cut off if it gets strangulated. The most common location for hernias is the groin area where, inguinal hernia, femoral hernia and scrotal hernia may occur.

Seventy five percent of the abdominal hernias are inguinal hernias [3], where inguinal hernias are further divided into direct inguinal and indirect inguinal hernia. In direct inguinal hernia, an intestinal loop pushes through a weak spot of the abdominal wall into the back wall of the inguinal canal, a triangular opening between the layers of abdominal muscle, whereas in indirect inguinal hernia loop of intestine passes down the inguinal canal. Indirect inguinal hernias are more common than the direct case. Figure 1.1 shows both the direct and indirect inguinal hernia.

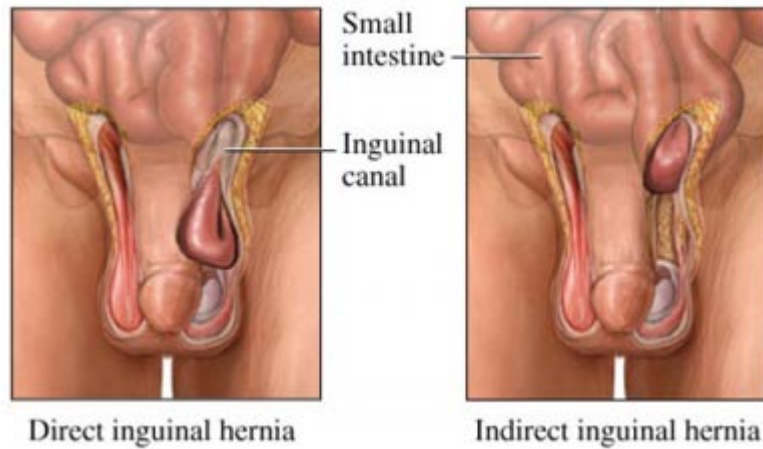


Figure 1.1 Direct and Indirect inguinal hernia [22]

### 1.3 Herniorrhaphy

Herniorrhaphy is a surgical procedure for correcting hernia. In herniorrhaphy, the surgeon pushes back the bulge of peritoneum through the opening and then closes the defect by stitching one side firmly to the other, as shown in Figure 1.2. Generally, there are two ways to repair the hernia in the groin area, open surgery and minimum invasive surgery. Some hernia in the elderly can be treated with a truss, a surgical appliance which helps to keep the hernia under control. In open surgery, skin incision is done over the area of the hernia.

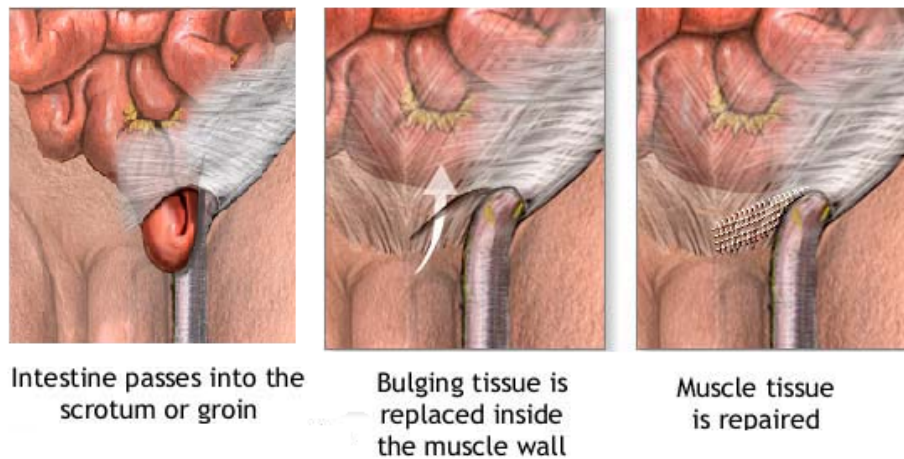


Figure 1.2 Steps of repairing inguinal hernia [23]

### 1.3.1 Minimally Invasive Surgery [22]

Minimally invasive surgery (MIS), also known as key hole surgery, is a modern surgical procedure in which operations in the abdomen are performed through small incisions (usually 2-3 cm) as compared to open body surgery. A small gas-tight tube like structure or trocar, is passed through the incision and peritoneal space is insufflated with carbon dioxide gas to create a working and viewing space. A tiny laparoscope, a telescopic rod lens system, with a video camera and a halogen or xenon light source, is inserted through a trocar port to view the operative field. The surgical instruments to repair the hernia are inserted through other trocar ports in the lower abdomen and surgery is performed by monitoring at the image received from the laparoscope on a television monitor. A scene from an actual laparoscopic surgery is shown in Figure 1.3



Figure 1.3 Scene from actual laparoscopic surgery [24]

There are many advantages of MIS surgery as compared to the open body surgery [5, 6]. The incisions in Laparoscopy are only 3-6 mm in length, which reduce the risk of blood loss and reduce the pain, leave behind less scarring and speed up the recovery time. However, for the surgeon, there are a few disadvantages of performing MIS. As laparoscopic surgery has to be performed by monitoring the images on a television monitor, a new hand-eye coordination skill is required, since the surgeon has little depth information in the 2D scene. Hand-eye coordination is especially difficult in MIS because the laparoscopic camera reflects 2D *mirror* images of hand movements and locations of anatomical landmarks [12]. Also, due to the trocar ports, the surgeons lose some of the manipulative freedom, which is available in open surgery. Difficulty in handling of the instruments, the lack of tactile perception and the limited working area will add to the technical complexities of laparoscopy.

#### 1.3.2 MIS: Training Procedures

A surgeon can practice surgery on mannequins, cadavers, animals, by observing an experienced surgeon or on actual patients under the supervision of an expert surgeon. Practicing on mannequin and cadavers lacks realism and flexibility because of the lack of real life tissue interaction and blood flow. While practicing on live animal can draw criticism and has ethical issue, it also has the problem of limited working area compared to real patients and the resulting limits imposed on the flexibility of the surgical instruments. Practicing on actual patients under an experienced surgeon can lead to surgical error. And none of the above has objective performance evaluation methods. Therefore, VR based simulators can be the best procedure to train the new surgeon and team on which different complicated cases can be practiced until surgeon and teams become expert in the procedure [12]

#### 1.3.3 VR-based Surgical Simulator

VR based surgical simulator simulates an environment for the purpose of surgery training, especially MIS. It provides a safe, real-time interactive deformation of the organs and force (haptic) feedback for different kinds of surgical sub-procedures such as grasping,

cauterizing, suturing, stapling and irrigating. It helps surgeons get familiar with the anatomy and practice the skills of manipulating instruments to expose the hernia site, putting back the hernia sack, placing the plastic mesh and staple it to the tissue.

#### 1.4 Motivation for Laparoscopic Surgery Simulator

Consider the following points:

- a) According to the National Institute of Diabetes and Digestive and Kidney Diseases (NIDDK) around 540,000 cases of inguinal hernia were reported on 1985 [4].
- b) Because of the above stated benefits of laparoscopic surgery, it is always preferred over open surgery for the treatment of the inguinal hernia.
- c) According to a survey [9], the incidence of medical errors has been estimated to run at between 4% and 16% of patient admissions, depending on the definition and threshold for what is an error or adverse event [7, 8]. There are many reports of serious accidents to neighboring organs during the laparoscopic surgery [7, 8]. According to the same survey, 70% of surgical errors led to slight or short-lived disabilities; in 7% the disability was permanent and in 14% it contributed to death.
- d) A novice surgeon needs a lot of training to develop new hand eye coordination to perform a surgery.
- e) Studies [10, 11] also show that, for a surgeon, practice of MIS surgery differs dramatically from practice of open surgery because of greater reliance on the surgery team. Each member of the MIS team plays a more important role than in open surgery [10, 11]. These complexities demand extensive training for the novice surgeon and teams before real surgeries to avoid accidents and reduce the complication and recurrence rate.

These are some of the reasons that a VR based simulator is being developed at the Virtual Environment Lab (VEL) at UT-Arlington in collaboration with UT-Southwestern Medical Center.



### 1.5 System requirement for a VR based laparoscopic simulator

As the name suggests, a VR based laparoscopic simulator should be able to generate the environment that allows a surgeon to experience and interact with a situation that occurs during real time laparoscopic surgery. The system must simulate the different surgical interactions such as grasping, cauterizing, suturing, stapling, palpation etc. that occur while restoring the bulge of the peritoneum to its original state and stapling the opening with plastic mesh. Creating the real time surgical environments involves the process of creating 3D anatomical models of soft tissue and rigid tissue models of the human organs and surgical tools. Simulating the surgical interaction between surgical tools and soft tissues involves graphical rendering of computer generated models, detecting collisions between instruments and deformable organ models, and haptic rendering of the collision response [12].

### 1.6 System Architecture

To accomplish the above requirements, the system architecture designed by the VEL research team is shown in the figure 1.4. It can be divided into four major blocks - offline processing, graphical and special effects, haptic block and real-time module.

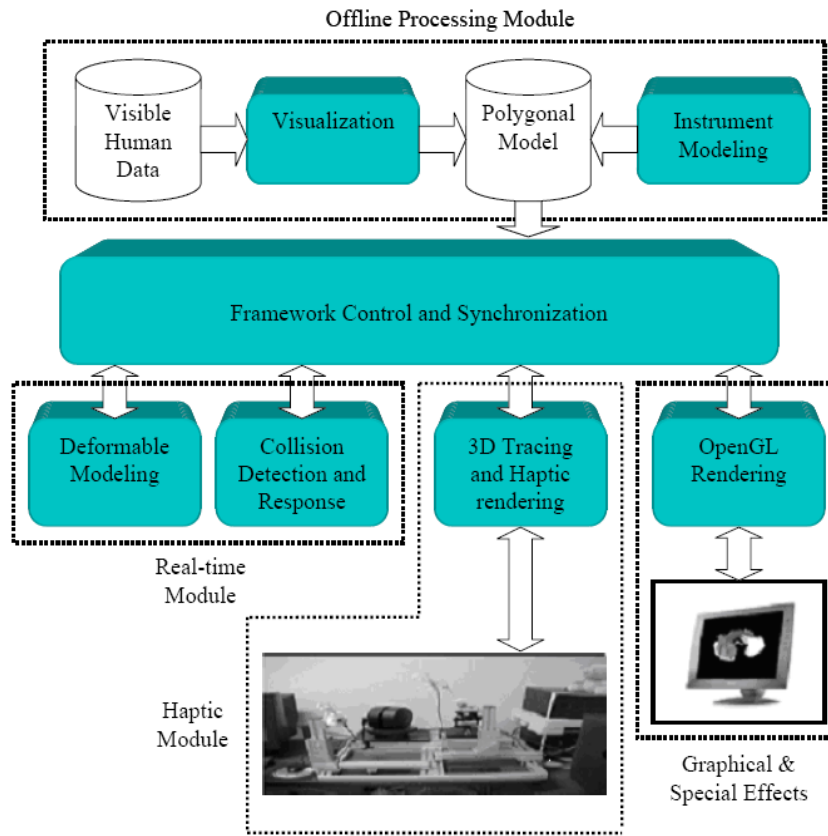


Figure 1.4 Block diagram of Laparoscopic surgery simulator [21]

#### 1.6.1 Offline processing

To create 3D anatomical models of human organ and surgical instruments, each organ and instrument needs to be rendered both geometrically for graphics display and haptically for force feedback. The static versions of these anatomical organs and instrument are created offline before the real time simulation. For our VR based laparoscopic simulator, all the 3D organs were created from the raw image of the Visible Human Data (VHD) from The National library of Medicine (NLM) [13]. Surface model of the organs were generated from those image using the marching cube algorithm [14] and realistic texture image was added to the model [15]. The steps of creating the texture mapped three dimensional anatomical organ from the Visible

Human Data is show in figure 1.5. And realistic inguinal hernia scene is generated [16] from the raw image of visible human data from national library of medicine.

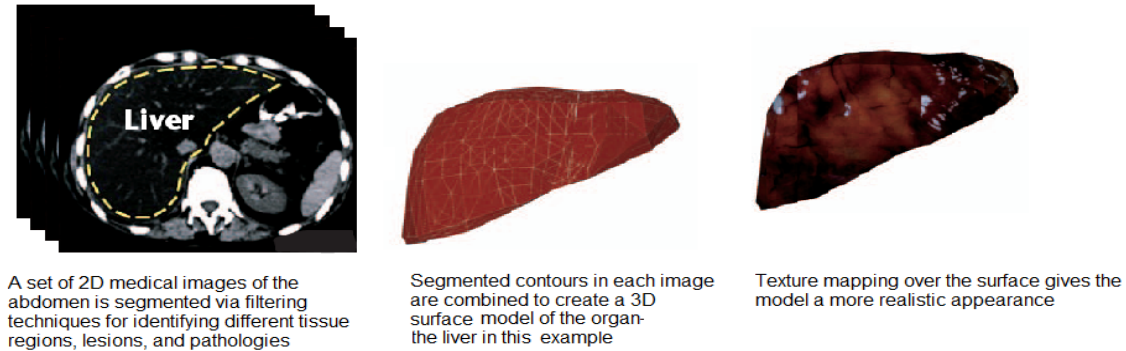


Figure 1.5 Steps in creating textured mapped 3D organ from 2D images [12]

### 1.6.2 Graphical and Special effects

The realistic texture image added to the three dimensional organs and instruments were rendered by this block. It also contains the visual effects such as bleeding, cauterization, irrigation, suction, suturing, stapling [17]. This block also contains the graphical user interface which helps to navigate and provide the method to record the simulated surgery and compare them with some performance metrics [18].

### 1.6.3 Haptic Block

This block provides the haptic rendering of the anatomical organs (soft tissue and bones) to the surgeon. To provide real time haptic rendering, the organ need to be physically modeled, i.e. it should support the deformation modeling and provide force feedback and has to be updated close to or above 1 KHz. This value is set by the somatosensory system threshold for humans [19]. The haptic simulation module also drives the hardware with an arm which can provide the necessary tactile feedback. For the surgery simulator developed at the Virtual Environment lab, the PHANTOM™ device from sensible technology [20] is used for the force feedback purpose. Figure 1.6 shows the phantom device used in the surgery simulator.

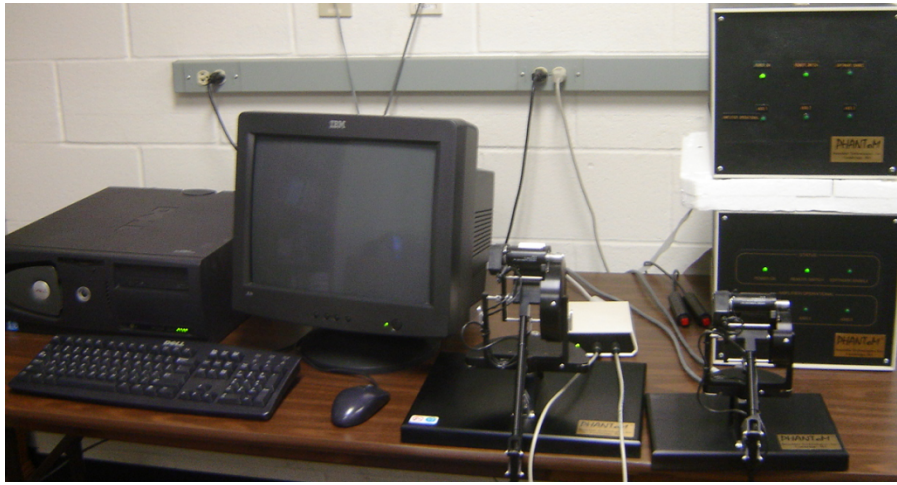


Figure 1.6 Phantom device used in the surgery simulator [27]

#### 1.6.4 Real time Module and Challenges

Real time algorithms are needed to show the interaction of an instrument with the virtual organs. The real time module block contains deformable modeling, collision detection and response algorithm to demonstrate those interactions between instrument and organs. The collision detection algorithm is used to find the intersection between the tissues with other objects which can be other tissue, bone or instrument in the scene. And collision response algorithm is used to calculate the appropriate response that should be applied to the object in the scene so that interpenetration between them can be avoided. The deformation model of the object will display the realistic response of those organs (soft tissue, bones and an instruments) when different responses are applied to them to prevent the interpenetration.

A collision detection algorithm which uses spatial Occupancy tests, with a Hashed and Cascaded data structure (OHC) [21], is implemented for interactions between deformable bodies and instruments with haptic environments. Deformable modeling of the tissue with mass-spring structure along with volume and area constraint is also implemented [26] for the laparoscopic surgery simulator. An appropriate collision response algorithm needs to be implemented in order to avoid penetration of the models

As the geometry of the tissue changes over time, the collision detection and response between instrument and tissue cannot be computed directly and requires intensive calculations to calculate the collision points and the respective responses. Collision detection performs those intensive calculations in discrete time. Collision detection algorithm implemented for the simulator [21] takes a few milliseconds of average time for detecting and reporting the colliding pairs of primitives (the elemental shapes such as triangles that describe an object). This time will further complicate the real time collision response since discrete time collision detection algorithm will miss the collision occurring between successive collision detection instances and hence will not be able show the proper collision response. The purpose of the thesis therefore is to

- Develop an algorithm to find the penetrated depth that is missed due to the discrete-time nature of the collision detection algorithms
- Develop a real-time collision response module for good visual and tactile feedback for the trainees.

In this thesis, some of the existing algorithms are investigated, and a new algorithm for collision response is proposed and implemented to handle the interaction between deformable and rigid bodies for a laparoscopic surgery simulator.

### 1.7 Problem Statement

Hence the problem statement for this thesis can be summarized as:

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

### 1.8 Organization of thesis

Chapter 1 gives a brief introduction of virtual reality, hernia, herniorrhaphy and the need of a VR based simulator for the training of MIS for inguinal hernia. It also covers the basic block diagram of VR based surgical simulator being developed at the Virtual Environment Lab at The University of Texas at Arlington, Chapter 2 reviews the popular collision response algorithms in the literature with the features, pros and cons, for each approach. Chapter 3 describes the theoretical background for collision detection, gives an overview of implemented collision detection in the surgical simulator and the derivation and the design of the proposed collision response algorithm. Chapter 4 discusses the implementation of the proposed collision response algorithm. Chapter 5 discusses the results of implementing the algorithm and concludes with a brief recommendation on potential future work.

## CHAPTER 2

### LITERATURE REVIEW

Collision detection and response have been of great interest for simulation researchers around the world. In this chapter, several collision response algorithms will be reviewed.

#### 2.1 Collision Response

Collision response refers to the dynamic behavior that objects follow after a collision occurs [54]. Accurate collision response calculates the collision impulses (forces) applied to the objects in collision to avoid their interpenetration and to provide accurate force feedback. There is a variety of approaches used to calculate collision impulses. These approaches can be categorized into three major ones;

- constrained-based approach,
- penalty based approach and
- impulse-based scheme.

##### 2.1.1 Constrained based approach

Constraints are used to describe the interactions between the objects, which occur only through physical contact. Constrained based approach computes constrained forces that are designed to cancel any external acceleration that would result in interpenetration [28]. D. Baraff proposed the idea of the constrained based approach [29, 30] which is appropriate to model the motion constraints imposed by contacting bodies. This approach keeps track of all possible contact states such as resting, rolling, sliding, colliding, etc. and calculates the combined collision force of all possible states for the collision response. D. Baraff reported mainly two problems in this approach: a) it is not proven while considering friction and b) the computational convergence in calculating the collision response is not always guaranteed. Also,

an unpredictable number of iterations has to be used to maintain the previously computed values within the correct bound.

F. Faure [31] proposed a method that satisfies both conservation of energy and inequality constraints by computing the resting contact forces based on the energy transfer between the colliding bodies. Another technique proposed by J Merger [32] is mainly applicable for cloth simulation, in which computational cost is reduced by using the coherence information between two discrete times during which the collision status is checked. The velocities of the cloth particle are constrained in the direction of closest point pair of the closest face instead of the normal direction of the closest face. Hence, this method helps to calculate the stable collision response for non-smooth surface and edge collision, where there are no faces available.

#### 2.1.2 Penalty based approach

The penalty approach is based on a traditional constrained optimization technique. This method has been previously used for constraining computer graphics models [33, 34]. The penalty based method adds quadratic energy terms that penalize the violation of the constraints [35]. The penalty method allows objects in the simulation to penetrate each other. Upon penetration, a temporary spring is attached between the contact points. This spring compresses over a very short time and applies equal and opposite forces to each body so that they will separate. As objects interpenetrate, the forces generated increase with the penetration distance.

The penalty based approach was first introduced by Moore and Wilhelms [36]. D. Barraf [37] mentioned that introduction of the quadratic energy for optimization in the penalty based approach has firm theoretical basis, but it is not numerically robust for the cases, where the spring constant is too high. Although the method has high computational cost for some cases, the method is widely followed for the deformable bodies, cloth and articulated rigid bodies simulation. M. Desbrun et al [38], M. McKenna et al [39] and J. Jansson et al [40] have



reported that the penalty based approach was successfully implemented for their deformable and rigid bodies' simulation.

Hirota et al. [41] claimed that the problem of robustness and computational cost of the penalty based approach lead to discontinuities and then to oscillation (instability during the response calculation). Thus, they proposed the idea of computing the smooth contact normal by employing a distance field (the distance between a particle inside the object and its boundary), which help the accuracy only in a limited way as the distance field is not updated upon the deformation. Later Heidelberger et al. [42] proposed to compute the smooth contact normal from a closed surface feature which reduces the oscillation from the discontinuous penetration depth. It was also mentioned that the algorithm can be applied in case of large penetrations to avoid non-plausible, inconsistent penetration depth information.

### 2.1.3 Impulse based approach

Impulse based approach is based on conservation laws and the assumption that there are no explicit constraints on the moving objects. In this approach, the impulses applied to the objects to avoid interpenetration are determined by the equations of conservation of energy and momentum. The post velocities after the collision are determined by the physical parameters, elasticity and smoothness of the collided objects. Impulse based approach was pioneered by Hahn [43]. Hahn proposed this approach to calculate the impulse force with friction at a single point by modeling the sliding and rolling contacts using the usual impact equation. Later Mirtich and Canny [44] extended the applicability of Hahn's method of resting contacts and proposed a more unified treatment for multiple objects in contacts. The main idea of their extension was to model all the contacts between objects as a series of impulses.

Although the approach has reasonable computational cost and was highly accurate, it was reported by Fabiana [28] that the impulse based approach was unable to handle the simultaneous and persistent contacts (for instance, a ball rolling down an inclined plane). In persistent contacts, the multiple micro collisions would bounce the objects from the surface.

#### 2.1.4 Other approaches

J. Spillmann and M. Teschner presented a new approach based on a contact surface [46]. The approach was independent of the actual deformation model and the applied collision detection algorithm, and does not require any user-defined parameters. However, the method introduces physically implausible states for resting contact of stacked objects.

O. Etzmus et al [47] gave a new dimension to the collision response algorithm. Their algorithm adds and subtracts virtual particles as needed to show the response. In this algorithm, collisions are divided into edge-to-edge and face-to-particle intersections. So when an object with a rough surface is involved in the collision, the algorithm will filter the edge-to-edge and the face-to-particle intersections, which allow simulation of a physically accurate system. As it can subtract the virtual particles during the simulation, the algorithm is faster than a regular particle system. However, since the algorithm is based on a particle based system, it cannot be applied to our mass spring model and hence not considered further.

V. Vuskovic et al [48] used the Finite Element approach to model the 3D object and P. Sovis et al [49] describe the algorithm which creates the V-region [50] of the object and then treats it as a tetrahedral mesh. Although the algorithms are accurate and easily implemented, they cannot be applied to our system, which is based on a mass-spring model with surface modeled object.

In table 2.1 [28], all three major approaches which were used to calculate the force between the objects are summarized based on their advantages and disadvantages.

Table 2.1 Comparison of widely used collision response approach [28]

	<b>Penalty-based Approach</b>	<b>Constraint-based Approach</b>	<b>Impulse-based Approach</b>
<b>Types of Object</b>	Rigid and Deformable	Rigid, deformable and articulated	Rigid
<b>Concept and Implementation Complexity</b>	Simple	Complex	Medium
<b>Computational Cost</b>	High	Low	Medium
<b>Number of Times Steps Required</b>	High	Low	Low to Medium
<b>Supported Contact Types</b>	Problem with stiff contacts	Problem when contacts mode change frequently	Problem with resting contacts
<b>Parallel Computation</b>	Possible	Complicated	Possible
<b>Physical Accuracy</b>	Depends upon time discretization	Accurate in most cases	Accurate
<b>Accuracy Verification</b>	Very difficult	Easy	Easy
<b>Advantage</b>	<ul style="list-style-type: none"> <li>- Simple</li> <li>- Incorporates</li> </ul>	<ul style="list-style-type: none"> <li>- Calculates exact answer</li> </ul>	<ul style="list-style-type: none"> <li>- Conceptually and</li> </ul>

Table 2.1 – continued

	<ul style="list-style-type: none"> <li>- static frictional model</li> <li>- Able to simulate qualitatively for all surface</li> <li>- Easily extendable for flexible objects</li> </ul>	<ul style="list-style-type: none"> <li>- Requires fewer steps in simulation</li> <li>- Easily proved</li> <li>- Computationally more efficient</li> </ul>	<ul style="list-style-type: none"> <li>- Algorithmically simpler</li> <li>- Works on system of bodies where contact surface change rapidly</li> </ul>
<b>Disadvantage</b>	<ul style="list-style-type: none"> <li>- Computationally expensive</li> <li>- Gives only approximate results</li> <li>- Difficult to verify the accuracy</li> </ul>	<ul style="list-style-type: none"> <li>- Computationally inefficient for the gentle collision</li> <li>- Pretty Complex to derive and implement</li> <li>- Assumption of rigid bodies without friction are too restrictive</li> </ul>	<ul style="list-style-type: none"> <li>- Applicable for rigid bodies only</li> <li>- Need to check possible contacts frequently</li> <li>- Inability to handle simultaneous and persistent contacts</li> </ul>

From the above comparison, it has been found that there is no single best approach to simulate collision response. A suitable approach for any modeling depends upon the application requirement. The penalty-based approach is chosen for this simulation because it is easy to implement and supports simultaneous and persistent contact besides being applicable for rigid, deformable and articulated objects.

## 2.2 Previous work on overcoming interpenetration

Collision detection has been a fundamental challenging problem in a real time virtual application because of its computational cost – it takes up so much of the real time budget. To overcome the real time computational load, the typical collision detection algorithm performs the "rejection test" whenever the two objects are far apart. When two objects are in close proximity, the collision detection method uses different algorithms to find the possible contacts. All the collision algorithms are discrete time algorithms and with slow performance because of collisions among multiple bodies. There is high chance that the algorithm will miss the exact instant of collision. So the collision response algorithm must determine the missed instant and then missed contacts between the colliding objects to show a plausible collision response. The algorithms to overcome the interpenetration caused by the missed collisions will be reviewed in this section

Heidelberger et al. [42] proposed an algorithm to find consistent penetration depth of deformable model objects. In the implementation, the objects were tetrahedral models. As our virtual organs are surface modeled, this algorithm cannot be used in our simulator.

Gerald Grabner and Andr es Kecskem ethy [51] proposed a continuous detection algorithm that will not miss any instant of collision by the using Runge-Kutta integration method. In this algorithm, the object models are "locked" during the integration step. Any interpolation or locking of the model cannot be implemented in our surgical simulator since the haptic module has to be updated around 1000 Hz rather than the graphics update rate of 30 Hz. It will be quite unrealistic if surgeon moves the haptic instrument and it is not updated in the scene because of locking of the instrument model.

V. Garcia-Perez et al. [52] presented an algorithm in which the interpenetration depth during discrete-time collision detection was calculated using fuzzy logic. The approach obtains the new position of each collided vertex of the organ by taking account of kinematic information of the surgical tool and geometric information of the organ. The fuzzy logic system predicts the

nature of the tool motion with respect to the organ, i.e. as penetration/extraction and sliding [52]. Although this method addresses the same problem as this thesis, the results presented are not in any compatible form to compare with the thesis results.

D. Coming and O. Staadt [53] have developed an algorithm that uses velocity-aligned discrete oriented polytopes (VADOP), bounding volumes based on k-DOPs to find the missed penetration depth. It was reported that the tight bounding volume representation VADOP offers fast update rates and is more suitable for applications with fast-moving objects. This paper attempts to solve the same problem as this thesis except using a completely different approach of dynamic collision detection. It is not clear if this approach is applicable to surface polygons or for haptic modeling.

## CHAPTER 3

### COLLISION DETECTION AND COLLISION RESPONSE

In this chapter, the theoretical background for collision detection, current implementation of collision detection in surgical simulator, and the proposed algorithm for collision response are presented.

#### 3.1 Model Representation

Object modeling is used in Computer Aided Design (CAD) to simulate the behavior of industrial materials and tissues. In image analysis, the object models are used for fitting curved surface and boundary smoothing. The modeling of an instrument and, especially soft tissue and organs is very important for virtual surgery, where interaction with virtual objects are required to simulate physically realistic representation of complex organs.

There are many types of model representation used in computer graphics [54]. One possible classification is shown in Figure 3.1. Broadly, the 3D models are classified as:

- Non-polygonal model
- Polygonal model

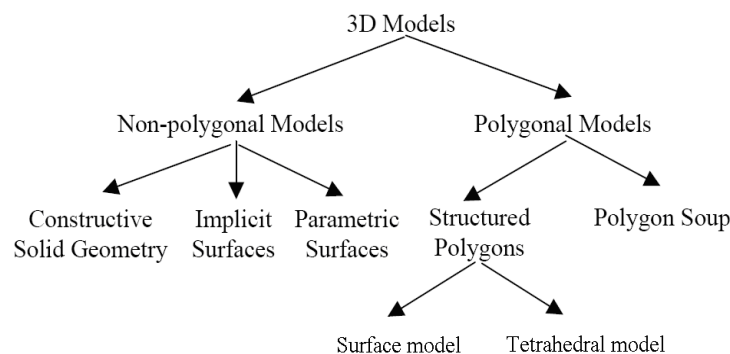


Figure 3.1 Classification of 3D model representation [54]

### 3.1.1 Non polygonal models

Non polygonal models can be represented in three different ways, Constructive Solid Geometry (CGS), implicit surface and parametric surface. In constructive solid geometry, the objects are generated from primitives such as cone, block, sphere etc., by combining them with set theoretic operations such as union, intersection, and set difference. In implicit surface representation, objects are defined by implicit mathematical functions. These functions unambiguously define what is inside and outside the model. In parametric surface representation, objects are defined by parametric equations. Unlike constructive solid geometry and implicit surfaces, parametric surface models do not represent the complete solid model.

### 3.1.2 Polygonal model

Polygonal models are the most commonly used models in computer graphics. They are simple to implement and have versatile properties. In a polygon soup, objects are represented as a collection of polygons that are not geometrically connected.

In a structured polygonal representation, the objects are formed from a closed manifold of polygons. Structured polygons are mainly constructed from triangular or rectangular primitives. A structured polygon can model objects in two ways - surface model and tetrahedral model. In a surface model, the objects are modeled only with surface information while in a tetrahedral model, they are modeled with volume information. In a surgical simulator, a surface modeled with triangles is chosen over the tetrahedral model because of the former's low computation cost. The lowest level primitive is a triangle. Figure 3.2 (a) and (b) show the surface modeled object and tetrahedral model object with a rectangular mesh. All the objects in the virtual surgery environment are represented using polygonal models with triangular mesh.

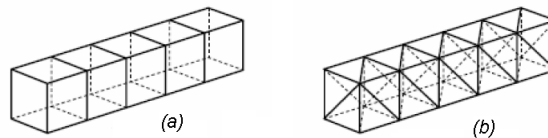


Figure 3.2 (a) surface modeled objects and (b) tetrahedral model objects [45] with rectangular mesh



### 3.2 Mass spring model [55]

In a mass spring model, an object is represented as collection of point masses connected by a spring in lattice structure (Figure 3.1). It is generally used for polygon/polyhedral objects, where every vertex is assigned mass  $m_i$  and interconnected to other mass  $m_j$  by a spring of natural length  $l_{ij}$  and spring constant  $k_{ij}$ . The linkage of mass particles is achieved in different ways; springs linking mass particles  $[i, j]$  with  $[i + 1, j]$  and  $[i, j]$  with  $[i, j + 1]$  are referred as structural springs. Springs linking mass particles  $[i, j]$  with  $[i + 1, j + 1]$  and  $[i + 1, j]$  with  $[i, j + 1]$  are referred as shear spring. Similarly, springs linking mass particles  $[i, j]$  with  $[i + 2, j]$  and  $[i, j]$  with  $[i, j + 2]$  are referred as flexion or bending spring. Depending upon the kind of force, shear, bending, compression or traction, applied to the object, these springs are used to preserve the original shape of an object

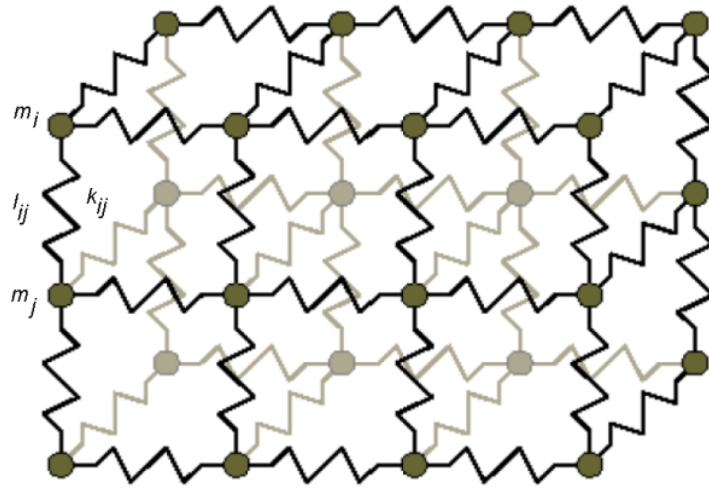


Figure 3.3 Mass spring structure of an object [55]

Both linear and non-linear springs can be used to model the deformable objects such as human organs that exhibit non-linear behavior. When a force is applied to mass  $m_i$  and if it is displaced from its rest position, it exerts a force to neighboring masses connected by springs.

Newton's second law governs the motion of each mass point in the lattice structure in mass spring model

$$m_i \ddot{\vec{x}}_i = -\gamma_i \dot{\vec{x}}_i + \sum_j \vec{f}_{ij} + \vec{f}_e \quad \dots \quad (3.1)$$

where  $m_i$  is the mass of  $i^{th}$  particle,  $\vec{x}_i \in \mathbb{R}^3$  is its position,  $\vec{f}_{ij}$  is the force exerted on it by the spring between mass particle  $i$  and  $j$ ,  $\vec{f}_e$  is an external forces i.e. sum of gravity and all other forces, which are not caused by the springs, acting on particle  $i$ ,  $\gamma_i$  is the damping coefficient for the resistance from the environment against the motion of mass. Vector  $\dot{\vec{x}}_i$  represents the velocity vector  $\vec{v}_i$  (or first derivative of position vector with respect to time) and  $\ddot{\vec{x}}_i$  represents the acceleration vector  $\vec{a}_i$  (or second derivative of position vector with respect to time) of mass  $m_i$ .

Modeling the springs with linear spring constants and considering the linear damping resistance for the relative motion between the mass  $i$  and  $j$ , we have

$$\vec{f}_{ij} = k_{ij} (\|\vec{x}_i - \vec{x}_j\| - l_{ij}) \frac{\vec{x}_i - \vec{x}_j}{\|\vec{x}_i - \vec{x}_j\|} + \lambda_{ij} (\dot{\vec{x}}_i - \dot{\vec{x}}_j) \quad \dots \quad (3.2)$$

where,  $l_{ij}$  is the rest length of spring between mass  $i$  and  $j$ ,  $k_{ij}$  is the Hooke's constant of the spring and  $\lambda_{ij}$  is the damping coefficient for the resistance against relative motion between mass particle  $i$  and  $j$ , which used to model the frictional energy loss during the deformation of the continuum object.

From equation (3.1) and (3.2), we can have

$$\ddot{\vec{x}}_i + \left( \frac{\gamma_i}{m_i} + \sum_j \frac{\lambda_{ij}}{m_i} \right) \dot{\vec{x}}_i - \sum_j \frac{\lambda_{ij}}{m_i} \dot{\vec{x}}_j - \sum_j \frac{k_{ij}}{m_i} (\vec{x}_i - \vec{x}_j) + \sum_j \frac{k_{ij}}{m_i} l_{ij} \frac{\vec{x}_i - \vec{x}_j}{\|\vec{x}_i - \vec{x}_j\|} = \frac{\vec{f}_e}{m_i} \quad \dots \quad (3.3)$$

Thus, for a system with  $N$  mass particles, the equation can be written as:

$$\ddot{\vec{X}} + \mathbf{D} \dot{\vec{X}} + \mathbf{K}(\vec{X}) \vec{X} = \vec{A}_e \quad \dots \quad (3.4)$$

where  $\mathbf{D}$  and  $\mathbf{K}(\vec{X})$  are  $3N \times 3N$  damping matrix and stiffness matrix respectively,  $\vec{X}$  is a column vector of the positions of  $N$  masses,  $\vec{A}_e$  is a column vector of the acceleration of the  $N$  masses due to external forces.  $\mathbf{D}$  and  $\mathbf{K}(\vec{X})$  are symmetric matrices. The system described by

equation (3.4) is a non-linear system since  $K(\vec{X})$  is a function of  $\vec{X}$  and can be converted to a first-order equation for convenience of analysis and integration.

$$\dot{\vec{V}} = -D\vec{V} - K(\vec{X})\vec{X} + \vec{A}_e \quad \dots \quad (3.5)$$

where,

$$\dot{\vec{X}} = \vec{V} \quad \dots \quad (3.6)$$

In simulation, continuous integration solution is approximated in discrete time steps, i.e. position velocity and acceleration of each mass are updated at discrete time points spaced by certain time steps. Many discrete numerical methods exist which can be used to solve the differential equation (3.5) numerically. For a surgical simulator, Euler's method is implemented for its simplicity.

Following are the main reasons that the mass spring model is used in our surgical simulator [24]

- Simplistic approach and well defined dynamics
- Relatively low computation cost
- Enables achieving haptic feedback rates
- Easy to construct the discrete medium for modeling deformable objects

This well understood model has been widely used for cloth and surgical simulation [56, 21] and facial animation [57-58]. Much of the research has been carried out to improve speed and accuracy. J. Zhang, S. Payandeh, and J. Dill [59] have improved the speed by updating Hooke's constant after refinement of the mesh. Wang and Devarajan [60] has shown that mass-spring model with triangular mesh has better performance than the mass-spring model with rectangular mesh for the bending force. They have also shown that the results with preload are much better than the one without preloaded springs.

Despite the vast research in mass-spring models, there are a few drawbacks of mass spring models which are listed below.

- a) Discrete representation of true organs
- b) Difficult in specifying the correct values of parameters for models
- c) Physical accuracy is often lacking
- d) Numerical instability phenomena often occur

Of these, problems b) and c) have been resolved to a large extent by Wang and Devarajan [25]. However, a) and d) remain. The interpenetration problem caused during collision detection is primarily due to a).

### 3.3 Collision detection

Collision detection has been an active research area in computer graphics, robotics, games, surgery simulation, cloth simulation and other simulated virtual environment. Deformable collision detection has emerged as a new sub-field in collision detection with increasing number of interesting applications. Interactive environment and deformable objects further complicate collision detection context in a surgical simulation. In interactive surgical simulation, the collision between a surgical instrument and deformable tissues has to be detected. However, during the surgical procedure, the topology of deformable objects can change due to cutting, grasping and self-collision of deformable object are likely to occur. The collision detection algorithm should be able to handle these scenarios.

There appears to be no universally optimal collision detection algorithm. Different research labs are using different collision detection algorithms based upon input data, requirements, and the required computational efficiency. All these approaches can be basically categorized into two; the Bounding Volume Hierarchy (BVH) and the Spatial Occupancy Test (SOT).

### 3.3.1 Bounding Volume Hierarchy (BVH)

In the bounding volume hierarchy, in which hierarchical structure of objects is created, every subdivision of model is represented by a bounding box. Each sub-division of a bounding box may be further sub divided into a set of smaller bounding boxes. During the pre-processing stage, a suitable method is chosen for creating and updating of the hierarchy in run time. For the collision detection test, BVH's of two objects are traversed top-down, i.e. starting from a big bounding box representing a whole module to a smaller bounding boxes representing sub modules, and primitive pair (elemental shape that describes the object, triangle in our case) of objects are tested for overlap. The research work in this area includes implementing different types of bounding volume and development of efficient algorithms to perform overlap tests while updating the hierarchy at the same time. Different bounding volumes, like Axis Aligned Bounding Box (AABB) [61], Object Oriented Bounding Box (OOBB) [62] and box tree [63] were explored in the past.

### 3.3.2 Spatial Occupancy Test (SOT)

In the spatial occupancy test method, a space is partitioned into uniform and non-uniform subdivisions or cells. Different algorithms are used to find potential colliding cells. These potential colliding cells are further refined to find exact colliding primitives. In this approach, researchers are mainly focused on making data structures much more efficient, so that collisions can be detected faster.

Most collision detection algorithms work in two stages; a broad phase followed by a narrow phase. The broad phase identifies the potential colliding bounding volumes. Hence it need to be applied to the whole set of primitives in the scene. The narrow phase performs the individual overlap test which calculates the information about overlap. Thus they are applied to the individual primitives.

### 3.4 Collision Detection: OHC Algorithm

Y. Shen [21] proposed and implemented the Spatial Occupancy test based collision detection algorithm using Hashed and Cascaded data structure (OHC). The algorithm is based on a subdivision of the 3D space into equal shape and size. In broad phase, the algorithm performs the spatial occupancy test (SOT). Spatial occupancy test classifies the cell into any of the three types

- Empty cell: containing no primitives
- Solo cell: containing primitives of the same object
- Potential colliding cell: containing primitives belonging to more than one object

In narrow phase, the algorithm performs, primitive to primitive intersection test on the primitives of the potential colliding cells. When collisions are detected, the algorithm provides information about the colliding objects and primitives.

### 3.5 Collision Response

Once collisions are detected, the next step is to show a realistic response. For producing a realistic response, information about deformable objects involved in the collision is required. It is not sufficient to just detect the overlapping of objects, but instead, precise information about the degree of collision, i.e. the penetration depth of the object needs to be found. The OHC algorithm only provides information about the objects and the primitives in the collision.

Therefore, the main focus of this research is to calculate the appropriate collision information and finding the force to be applied on the object to prevent interpenetration. Collision response will be different for different objects based on their nature. Hence, collisions must be first classified into different cases. In our surgical simulator application, they are classified into the following types:

- (surgical) Instrument to (surgical) Instrument

- Instrument to Rigid body (bones)
- Instrument to Tissue
- Tissue to Rigid body
- Tissue to Tissue

The first two can be categorized generically as rigid to rigid body collision. The third and fourth can be categorized into rigid to deformable body collision and, the last one falls under the category of deformable to deformable body collision. For an interactive surgery simulator, tissues to instrument collisions are the most important. Hence, the research mainly focuses on instrument to tissue collisions.

### 3.6 Instrument to Tissue Collision

In collision detection, triangle to triangle intersection tests [64] are performed and once collisions are detected, only the triangle and object information are provided for showing the collision response. The first step in our collision response algorithm is to find the points of collision on the tissue and the instrument surface. The second step is to find the penetration depth and then the deformation vector with respect to the point of collision. Finally, the penalty force has to be calculated, which is applied on the point of collision of tissue and instrument surface to avoid the interpenetration between them.

### 3.6.1 Requirement for collision response algorithm

Figure 3.4 shows the instant of collision detected in the virtual laparoscopic surgery simulator. It can be seen from the figure that several triangles of an instrument and tissue are involved in the collision. The collision detection algorithm will detect each of the collided triangles and report to the collision response algorithm.

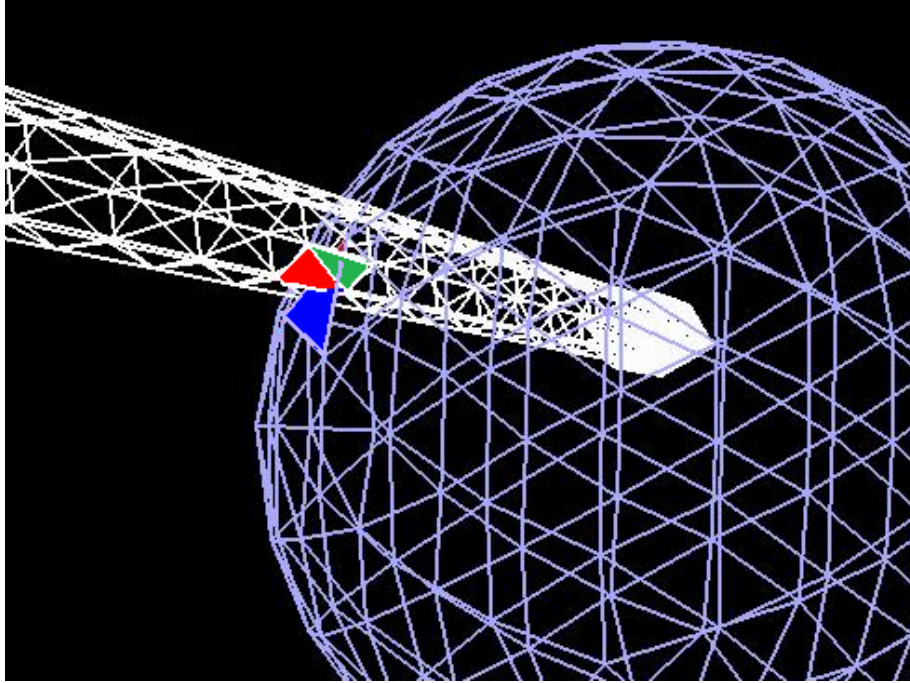


Figure 3.4 Instant of collision between instrument and tissue

The blue triangle represents the tissue triangle and red and green triangles represent the instrument triangles that are involved in the collision. While closely analyzing each collision, it can be discovered that for collision between blue tissue triangle and red instrument triangle, only one vertex of instrument triangle is beneath the tissue surface. For collision between blue tissue triangle and green instrument triangle, two vertices of the instrument triangle are below the tissue surface. Similarly, for red instrument triangle and blue tissue triangle collision, both triangles intersect each other completely. On the other hand, green instrument triangle and blue tissue triangle have an edge intersection only. The proposed collision response algorithm has to be dynamic so that it can properly handle each such scenario.



### 3.6.2 Finding the point of collision

The instrument and the tissue are both represented by a set of triangles. So, when the collision eventually occurs, it does so at the level of an instrument triangle and a tissue triangle. There can be a number of ways in which a tissue triangle and an instrument triangle intersect each other. The collision response algorithm should be able to handle all such scenarios. Let  $A_1, A_2, A_3$  be the vertices of the instrument triangle and  $B_1, B_2, B_3$  be the vertices of the tissue triangle which are detected in the collision detection as shown in figure 3.5 (a). In this case, a vertex from the instrument has collided with and penetrated a tissue triangle. The point  $A_3$  is now inside the tissue. When the tissue plane and the instrument plane intersect, they do so along a line. Let  $MN$  be that line of infinite length and let  $P$ , point of collision, (not shown in the figure), be any point on the line  $MN$ . Once the point of collision is determined we calculate the penetration depth by using OOB of an instrument (as described later).

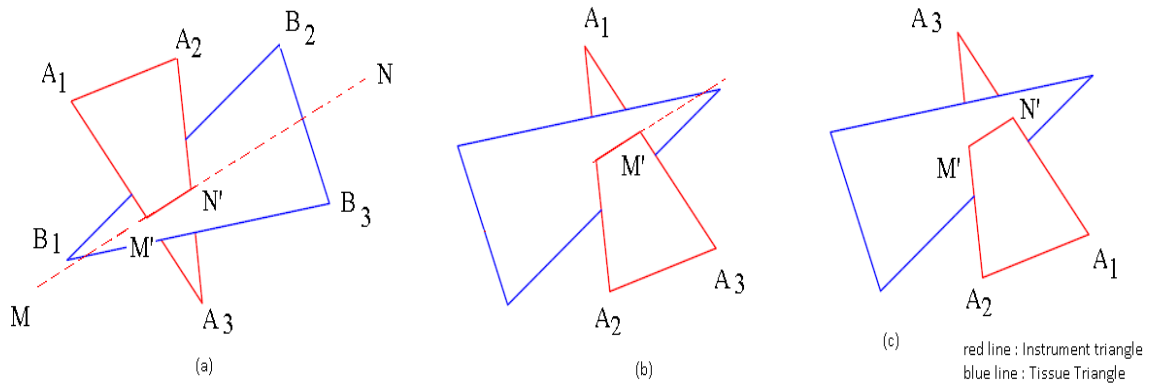


Figure 3.5 Different orientations of tissue and instrument triangle during collision

To ensure that  $P$ , the point of collision always lies on the tissue surface, line  $MN$  should be further restricted to  $M'N'$ . In our algorithm, we designate  $M$  as the point where the line between vertex  $A_1$  and  $A_2$  intersects the tissue surface and  $N$  is a point where the line between vertex  $A_2$  and  $A_3$  intersects the tissue surface. It is easy enough to calculate  $MN$  in the scenario

shown in fig. 3.5 (a). Our algorithm is designed such that by definition, the two vertices of the instrument which are on one side of the tissue plane are always designated  $A_1, A_2$  and the instrument vertex that is on the other side of the tissue plane is designated  $A_3$ . Once this is done a routine then simply calculates the penetration depth. However, in figure 3.5 (b) we show the scenario where the entire line  $A_2A_3$  of the instrument has already penetrated the tissue triangle by the time the collision is detected. In this case, the algorithm simply re-designates the instrument vertex nomenclature so that  $A_1$  and  $A_2$  are on one side of the tissue and  $A_3$  on the other as shown in figure 3.5 (c). A similar re-designation is carried out when the original instrument triangle penetrates the tissue with  $A_3A_1$  inside the tissue.

For simplicity, let us assume that point of collision  $P$  is  $N$ . If the orientation of instrument triangle and tissue triangle is as shown in figure 3.6 (a), where the tissue triangle is penetrating the instrument triangle or as shown in figure 3.6 (b), where the instrument triangle is penetrating the tissue triangle, then the point  $N$  is lying outside the tissue triangle. The penetration depth and deformation vector calculated using this point will be incorrect. Thus, once  $P$  is calculated, it must be validated as described below.

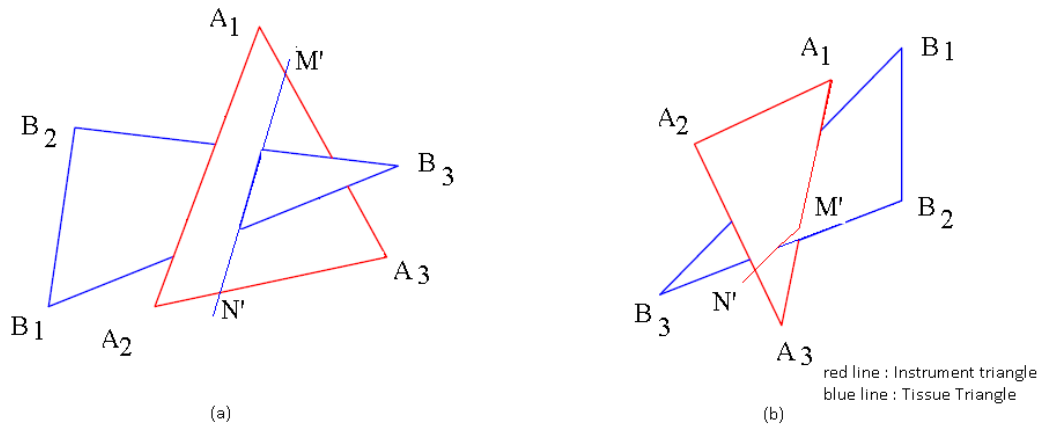


Figure 3.6 Cases where point of collision  $N$  or  $P$  does not lies on tissue surface

The collision response algorithm checks for the validity of point  $N$ . The Barycentric coordinate technique [65] is used to find the point  $N$ , which might be lying inside or outside the

triangle  $B_1, B_2, B_3$ . If point  $N$  lies outside the triangle, then the new point  $N$  is calculated such that it lies at the edge of the tissue triangle. Figure 3.7 (a) and 3.7(b) shows the calculation of the new point  $N$  which falls on the edge of the tissue surface for figure 3.6(a) and 3.6(b). Among the figures 3.5 and 3.6, all possible collision scenarios between a tissue triangle and an instrument triangle are covered.

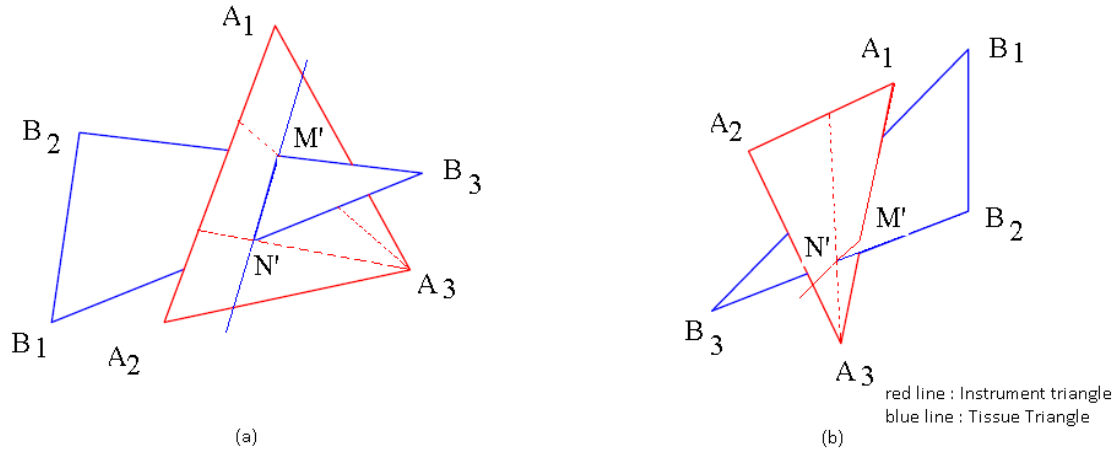


Figure 3.7 Calculation of new point of collision  $P$  that lies inside the tissue surface

### 3.6.3 Finding the penetration depth and the deformation vector

For  $n$ -body processing, the discrete-time collision detection will likely miss the exact instant of collision. If collisions are detected after such instance as shown in figure 3.8, where  $t_1$  is the instance when no collision is detected and  $t_2$  is the instant when collision is detected, then the penetration depth calculated from the triangles detected at that instant will not be accurate.

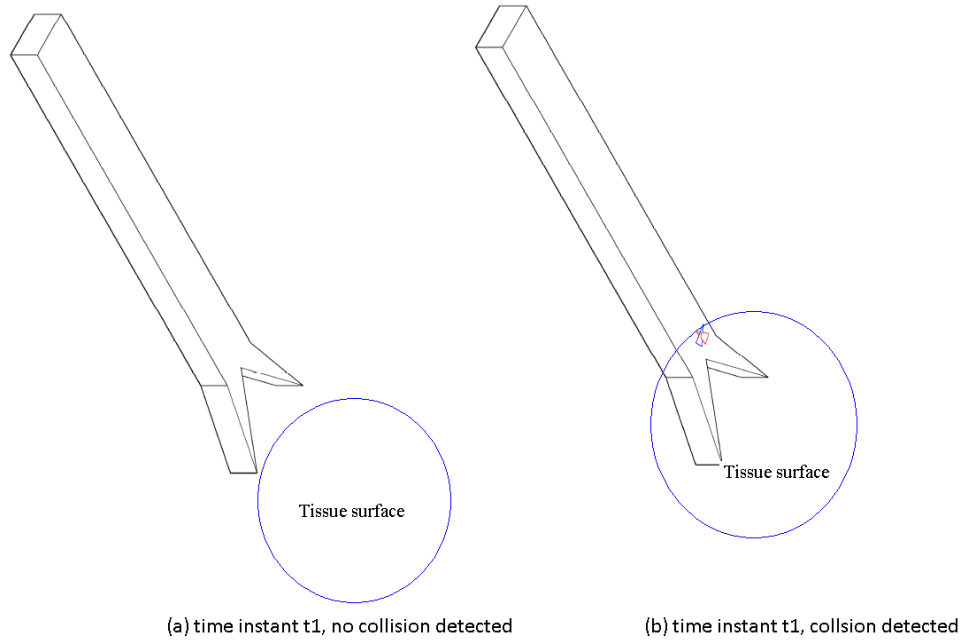


Figure 3.8 Object positions at collision detection instant at  $t_1$  and instant  $t_2$

The conventional method to overcome this discrete-time situation is to find the exact instant of collision by interpolating the time between the instance  $t_1$  and  $t_2$ , and then finding the penetration depth for the correct instant. Whether such a method can or cannot be implemented is determined, among other things, by the update rate. For visualization, the typical update rate is 30 Hz and hence this method is practical. However, this method cannot be implemented in the surgical simulator, because the haptic device has to be updated at 1000 Hz rate. The collision response algorithm will not be able to show a realistic response, if the interpolation is done.

J. Butala [24] has implemented the collision response algorithm by calculating the penetration depth from the intersecting triangles and by moving the instrument and tissue surfaces in opposite directions, on a primitive by primitive basis. This collision response algorithm performs reasonably for small penetrations, but has unrealistic response for large penetrations.

This thesis proposes to use the Object Oriented bounding box (OOBB) method for calculating penetration depth. As the instrument will not deform during the collision response,

the object oriented bounding box information of the instrument is used to find the penetration depth. Although different types of bounding box approaches are proposed [61, 62, 63], OOB is preferred because it is computationally less expensive to build and it provides nice compaction for uniformly shaped instruments.

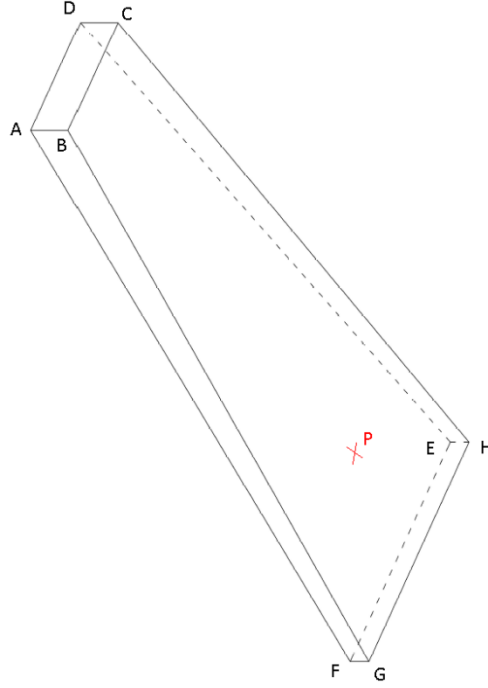


Figure 3.9 Bounding box of an instrument for the collision detected instant of figure 3.8 (b)

After building the bounding box ( $ABCDEFGH$ ) for the collision detection instance, the point of collision  $P$  will always lie inside the object oriented bounding volume as shown in figure 3.9. The depth information for the instrument overlapping the tissue surface is calculated by using the outer normal information of the tissue plane. Figure 3.10 shows  $\overrightarrow{Pd_1}$ ,  $\overrightarrow{Pd_2}$  and  $\overrightarrow{Pd_3}$ , the penetration depth in three axial directions of the instrument overlapping with the tissue surface. The final penetration vector  $\overrightarrow{PD}$  will be the resultant of these vectors.

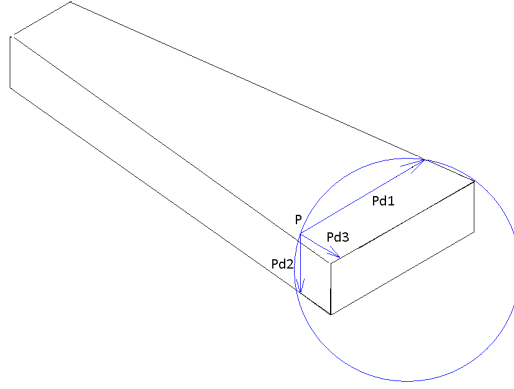


Figure 3.10 Three axial penetration depth for the collision detect instant of figure 3.8 (b)

Let  $\vec{V}_{I1}, \vec{V}_{I2}$  and  $\vec{V}_{I3}$  represent the velocities of the three vertices of the instrument triangle and  $\vec{V}_{T1}, \vec{V}_{T2}$  and  $\vec{V}_{T3}$  represent the velocities of the three vertices of the tissue triangle which are detected by the overlap test. The velocity of the instrument surface  $\vec{V}_I$ , and the velocity of the tissue  $\vec{V}_T$ , at the point of collision are calculated from the velocities of an instrument and tissue triangle by using Barycentric coordinate technique [65]. Finally, the deformation vector  $\vec{D}$  is calculated based on the above information. The magnitude of the deformation vector will be the projection of the final penetration vector on the normalized relative velocity of the instrument with respect to the tissue, i.e. the dot product of  $\vec{PD}$  and  $(\vec{V}_I - \vec{V}_T)$ . The direction of the deformation vector for the tissue will be the direction of relative velocity of an instrument with respect to tissue,  $(\vec{V}_I - \vec{V}_T)$ . Figure 3.11 shows different deformation vectors for the same penetration depth depending on the relative velocity of instrument with respect to tissue in two-dimensional case.

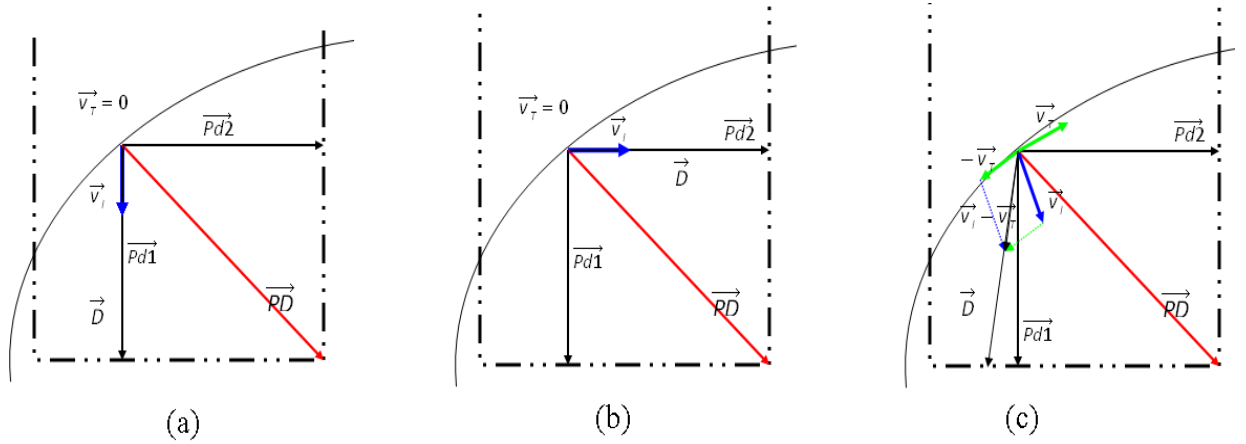


Figure 3.11 Different deformation vectors for same penetration depth. Case (a) instrument velocity is in the direction of  $\overrightarrow{Pd_1}$ , Case (b) instrument velocity is in the direction of  $\overrightarrow{Pd_2}$ , Case (c) instrument and tissue velocities are in different directions

$$\overrightarrow{PD} = \overrightarrow{Pd_1} + \overrightarrow{Pd_2} + \overrightarrow{Pd_3} \quad \dots \quad (3.7)$$

$$\overrightarrow{D} = \overrightarrow{PD} \cdot (\overrightarrow{V_i} - \overrightarrow{V_t}) \cdot \frac{(\overrightarrow{V_i} - \overrightarrow{V_t})}{|\overrightarrow{V_i} - \overrightarrow{V_t}|} \quad \dots \quad (3.8)$$

During an interactive surgery simulation, there can be a special case, where an instrument completely penetrates the tissue as shown in figure 3.12 (a). In such a case, the collision detection algorithm will detect the collision on both surfaces (surface 1 and surface 2). The proposed collision response algorithm will always calculate the penetration depth in the opposite direction of the outer normal of the tissue surface. For the collision detected on surface 2, the algorithm will calculate an incorrect penetration depth.

To handle such a situation, the vertex on top of the instrument was marked during the offline stage of collision detection algorithm. And for every collision detected, the outer normal of tissue triangle is checked with the vector joining the tissue vertex to the vertex marked at offline stage. If these vectors are in the same direction, then the algorithm will process further, and if they are in opposite directions, it will ignore such cases. As shown in figure 3.12 (b), collision response algorithm will discard the collision on surface 2 and will only determine the

penetration depth and deformation vector for collision detected at surface 1. At the end of one collision detection cycle, the deformation vector calculated from surface 1 will be applied to the tissue to prevent further penetration.

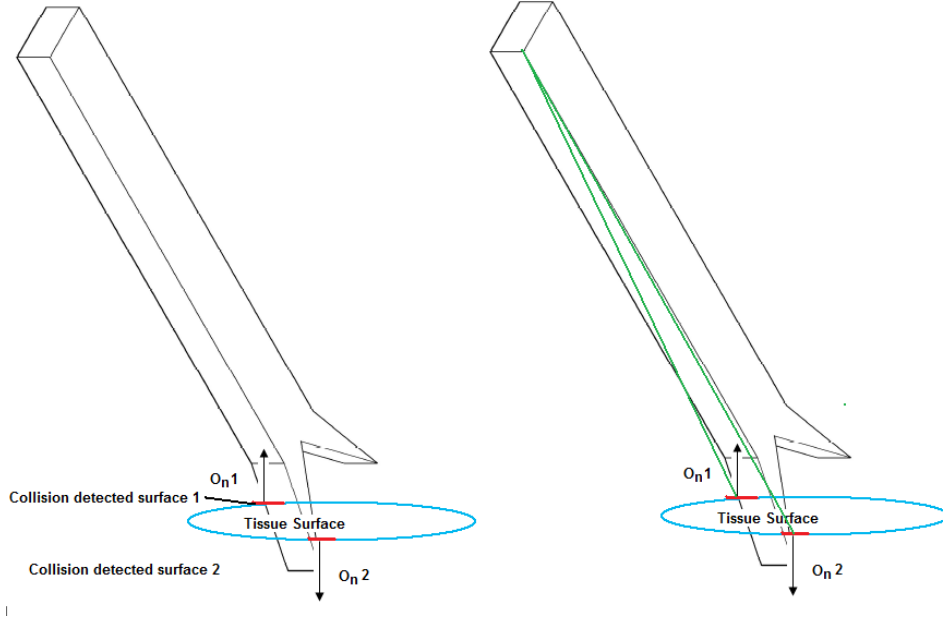


Figure 3.12 Case where an Instrument penetrates through tissue surface

### 3.6.4 Finding the penalty force

The penalty based algorithm is the most appropriate for interactive simulation of deformable objects due to its simple implementation [28], and hence it is used in our simulation. The objects are moved out of penetration by inserting a temporary spring between the contact points, which exerts equal and opposite forces on each body. As the objects interpenetrate more, the forces generated will be increased with the increasing penetration distance. The penalty force,  $\vec{F}$  is calculated from the deformation vector  $\vec{D}$  and relative velocity of the instrument with respect to tissue.

The penalty force that has to be applied on the tissue surface is given by

$$\vec{F} = \left( K |\vec{D}| - \mu (\vec{V}_t - \vec{V}_r) \cdot \frac{\vec{n}_i}{|\vec{n}_i|} \right) \frac{\vec{n}_i}{|\vec{n}_i|} \quad \dots (3.9)$$



where,  $K$  is the Hooke's constant,  $\mu$ , is the damping constant and  $\vec{n}_i$  is the unit inner normal of the tissue surface. An equal amount of force is applied to the instrument on opposite direction so that surgeon can feel the surface of the deformable objects using the haptic device.

The collision response algorithm is able to find the exact point of collision and the depth of the instrument overlapping with the tissue, to calculate the penalty force even with the discrete collision detection and  $n$ -body processing. Thus, the penalty force is applied to both tissue and instrument surface to prevent the object from penetrating even deeper. As the result of penalty force, the objects will bounce away from the colliding objects.

## CHAPTER 4

### IMPLEMENTATION DETAILS

In this chapter, the current software framework of the surgery simulator and the implementation of the proposed collision response algorithm are described in detail.

#### 4.1 Implemented Modules

The proposed collision response algorithm is implemented and integrated with other existing modules in the surgery simulator. Table 4.1 summarizes the functional modules implemented in the surgical simulator.

Table 4.1 Major modules in the surgery simulator [21]

Collision detection	OHC algorithm, a spatial tessellation method [21]
Collision Response	<i>Proposed Algorithm, my work.</i>
Soft tissue modeling	Mass spring structure along with volume and area constraint modeling
Framework Implementation	Software developed with C++ on dual processor station
Geometric Modeling of tissue and instrument	Mesh reconstructed from cross-section data and then decimated and smoothed to 5K-7K triangles, CAD based modeling and texture generation
Haptic rendering and instrument motion tracking	Bimanual haptic feedback rendered with two PHANTOM <sup>TM</sup> devices, programmed with Ghost SDK <sup>TM</sup>
Motion Tracking for instrument handle	Sensor, A/D and DSP circuits
Visual Rendering	OPENGL graphics library

#### 4.2 Framework of surgical simulator [24]

The framework of the surgical simulator is a real-time application which is capable of simulating a complete surgery. The different classes which are implemented are listed below.

- *CGeoModel*
- *CSetDialog*
- *CDeformable*
- *CShare*
- *CDeformation*
- *CGraphic*
- *CThreadControl*
- *CMap*
- *CHaptic*
- *CHandleControl*
- *CPerformanceMonitor*
- *CMI*

The *CGeoModel* class loads the polygonal models of instruments and tissue surfaces, from the VRML (Virtual Reality Modeling Language) file format. It also has the information about the texture for different objects in the environment. It stores all the information in the corresponding *CGeoModel* instances.

The *CSetDialog* class allows the various settings for the simulator, which varies from the parameter ranging from force constants to the different objects loaded in the scene.

The *CDeformable* class implements the mass spring model which is used for the deformation of objects.

The *CShare* is a synchronization class that ensures the data integrity of objects during the parallel access from *CGeoModel* and *CDeformable* class.

The *CThreadControl* class gives the different priorities to the different classes so they can use the required amount of memory and processing power. There are four main threads running in the system.

*CHaptics*: regulates the haptic device thread

*CDeformation*: regulates the deformation thread

*CCollision*: regulates the collision detection and collision response thread

*CGraphics*: regulates the real-time graphics thread

The first three threads have equal priority to the operating system, since they are time critical; while the *CGraphics* thread has a lower priority since it requires a low update rate.

The *COMap* class defines the various functions for collision detection and collision response. The parts of collision response will be described in details later on.

The *CHaptic* class is used to create an instance of PHANTOM™ haptic device. This class is used to update the position, orientation and rotation of the haptic device on runtime.

The *CHandleControl* updates the handle motion and data connectivity with motion tracking circuits of the haptic devices.

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

The *CGraphics* class calls OpenGL, a standard graphics library that implements real-time texture rendering and special visual effects.

The *CMI* is the main interaction class. It controls all the interfaces of software framework.

#### 4.3 Implementation details of proposed collision response algorithm

As discussed in section 3.4.2, the broad phase of collision detection algorithm identifies the objects that are not colliding. In other words, it performs the rejection test and possible colliding objects are passed to the narrow phase. In narrow phase, the collision detection algorithm finds the primitives of objects that overlap with others. These primitives and objects information are passed to the collision response module.

The main flowchart of the collision response algorithm is shown in figure 4.1. To reduce the build time of the object-oriented bounding box for an instrument, each instrument in the scene is analyzed in the offline stage. All the memory locations of the vertices used for creating bounding volume in the offline stages are stored in a separate array, so that there is no need to build the object oriented bounding box for collision detection instant. The newly created array is referred whenever bounding information is required and is able to save some computational time during run-time.

As mentioned earlier, this research only focuses on the instrument to tissue collision. Whenever the primitive's information is passed to the collision response module, the collision response algorithm checks the ID of an object involved in the collision. If the objects are instrument and tissue then it proceeds with further processing. There can be a special case of instrument and tissue collision i.e. when instrument is grasping the tissue. For such cases tissue has to simply follow the instrument movement, which is accomplished by maintaining the same relative position of tissue triangle with collided instrument triangle. For other cases, the collision response algorithm first finds the point of collision, then penetration depth and deformation vector and, finally the penalty force that has to be applied to the tissue triangle and instrument triangle as shown in the flowchart.

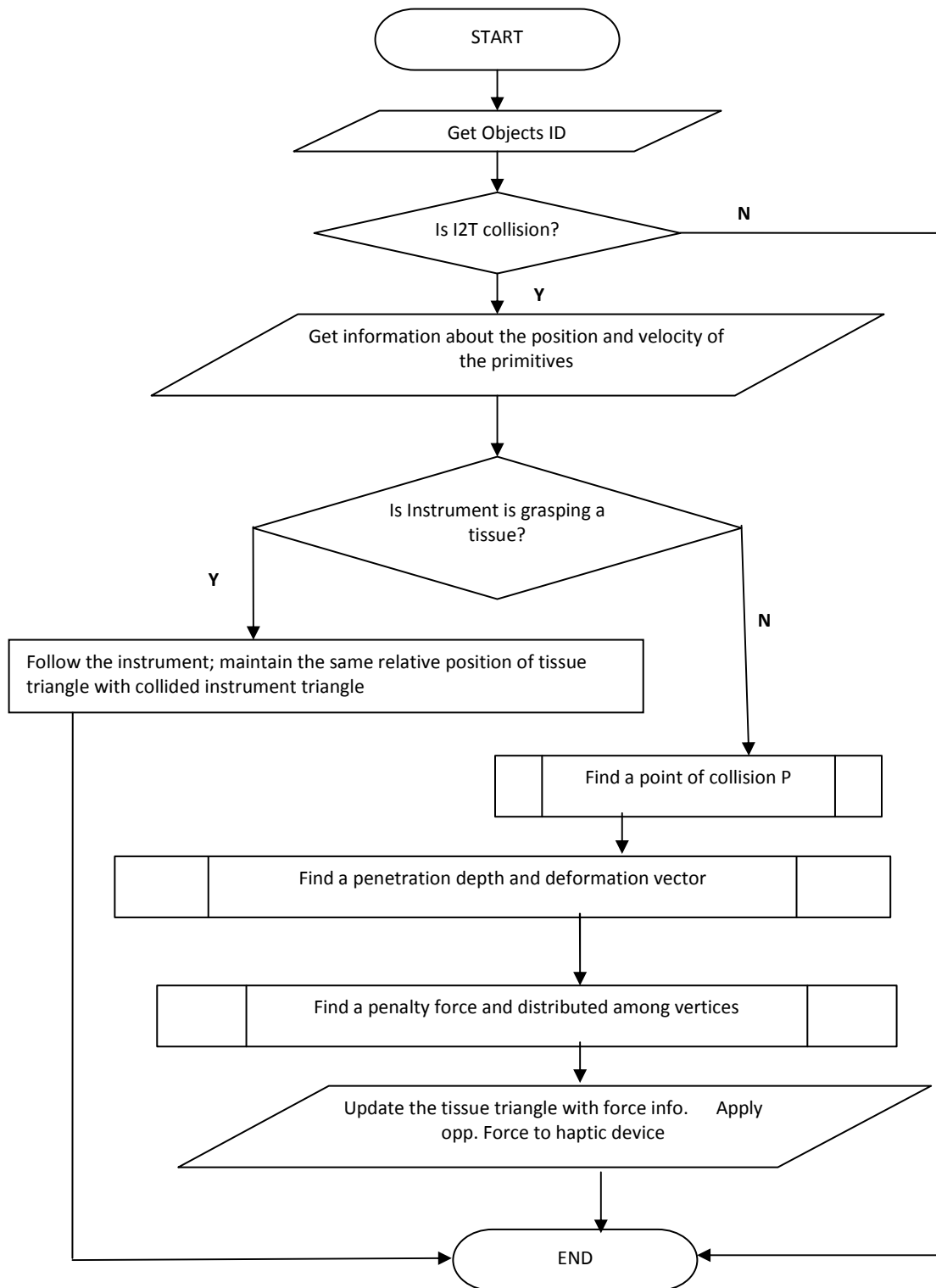


Figure 4.1 Main flow chart of collision response algorithm

#### 4.3.1 Finding a point of collision

The flow chart for finding a point of collision is shown in figure 4.2. To find the point of collision, the outer normal information for tissue surface is calculated from the cross product of the vectors that points to two other vertices of the tissue triangle from the first one. Each vertex of the instrument is analyzed, with respect to the outer normal of the tissue surface to determine the location with respect to the tissue plane.

In the next step, the intersection point of lines connecting the two vertices on the same side of the tissue to the third is calculated. Although the intersection point lies on a tissue surface, due to different orientations of tissue and instrument triangles, such intersection points may not lie inside the tissue triangle. These points are further checked to find out if they lie inside the tissue triangle or not. If they are outside the tissue triangle, new intersection points are calculated which lie at the edge of the tissue triangle. The detailed steps for checking the validity of the point and calculation of the new point is described in the next section. One of the intersecting points is returned as the point of collision in the end.

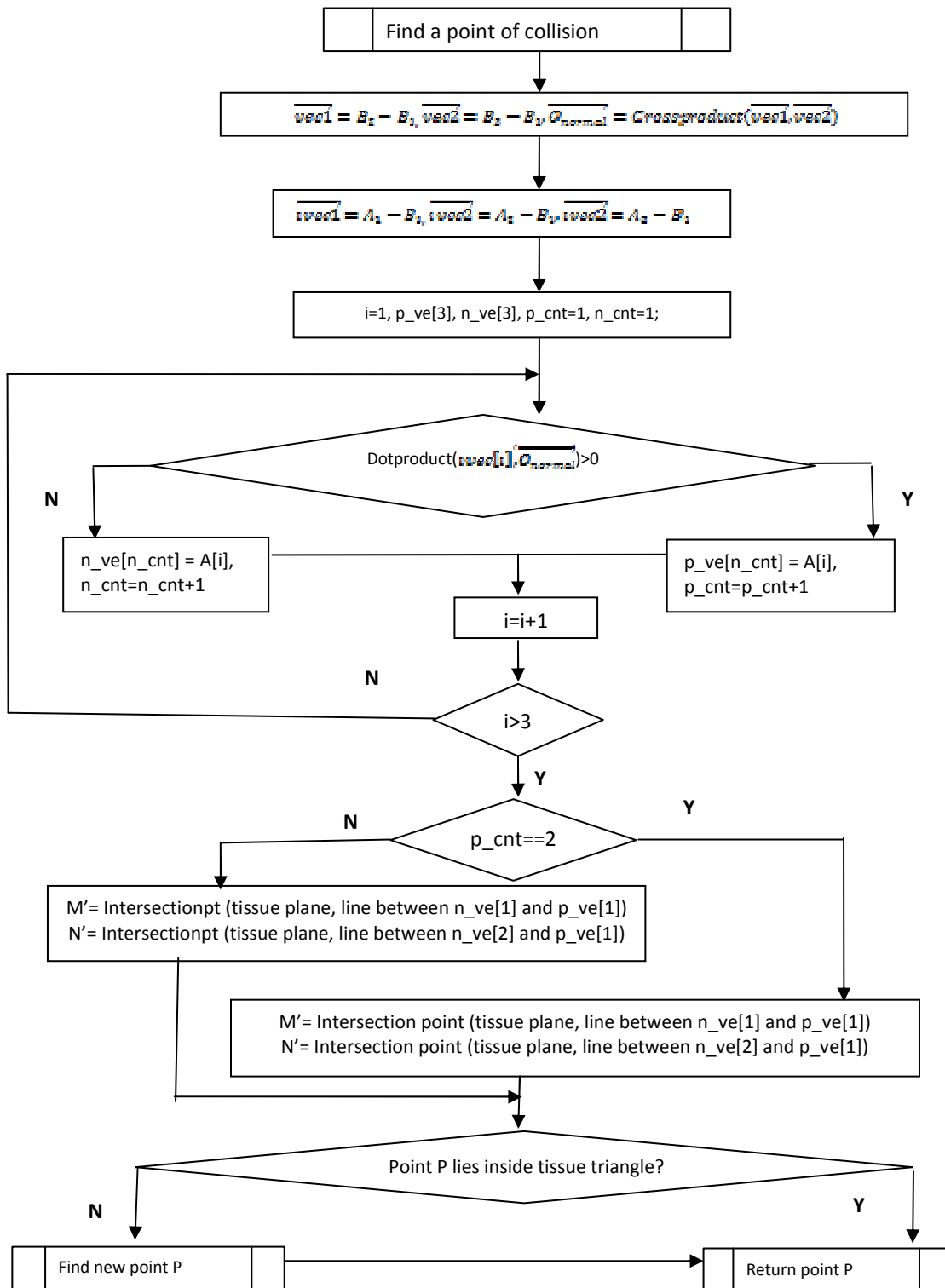



Figure 4.2 Flow chart of finding a point of collision 



#### 4.3.2 Checking the validity of intersection point

The flowchart for checking the validity of the intersecting point is shown in figure 4.3. The Barycentric coordinate technique is used to find the validity of the intersecting point. The barycentric coordinate, at any point  $P$ , point of collision, can be defined by the vertices of the simplex ( $n$ -dimensional analogue of a triangle). In the case of a triangle  $B_1, B_2$  and  $B_3$ , any point  $P$ , located in the triangle can be written as the weighted sum of these vertices.

$$P = \lambda_1 B_1 + \lambda_2 B_2 + \lambda_3 B_3 \quad \dots \quad (4.1)$$

where  $\lambda_1, \lambda_2$  and  $\lambda_3$  are the barycentric coordinate, which are subjected to the constraint of

$$\lambda_1 + \lambda_2 + \lambda_3 = 1 \quad \dots \quad (4.2)$$

In the case of a triangle, the barycentric coordinate of point  $P$  simply represents the ratio of the signed area of  $PB_1B_2, PB_1B_3$  and  $PB_2B_3$  with the total area of  $B_1B_2B_3$ . It is also known as the areal coordinate. For any point  $P$  that lies inside the triangle, the barycentric coefficient will always lie between  $[0,1]$ . If one the coefficient is negative, then a new intersection point is calculated from the lines connecting the points  $MN$ , with the side of the triangle lying close to the intersection point. Algorithm will return a new point if the given intersection point does not lie inside the triangle.

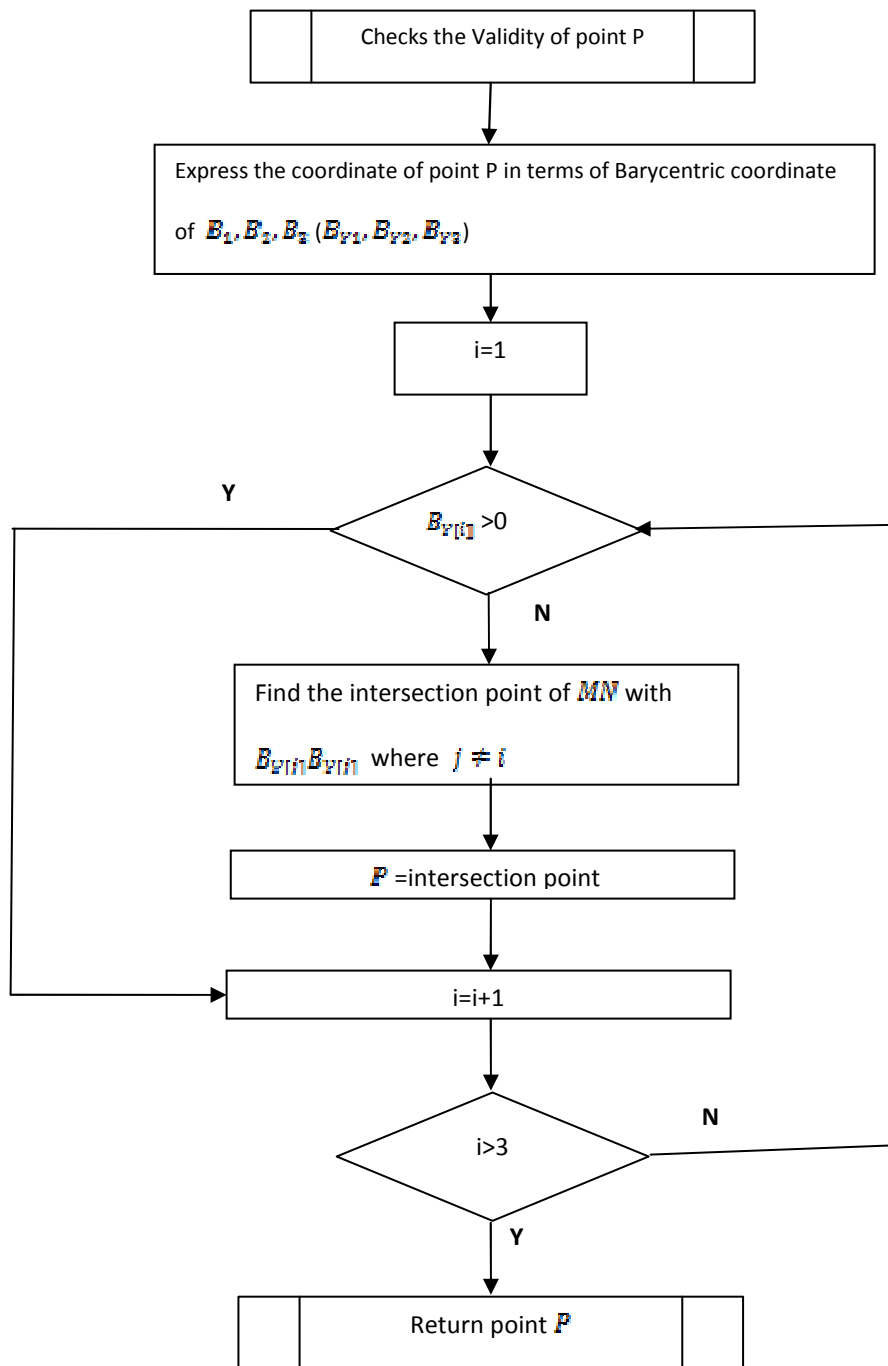


Figure 4.3 Flow chart of checking the validity of point  $P$

#### 4.3.3 Finding the penetration depth and deformation vector

The flowchart for finding the penetration depth and deformation vector is shown in figure 4.3. Once the point of collision is determined, the object oriented bounding box and axial vector information are retrieved from the array, which were created during the pre-processing stage. All intersecting points formed by the line, from the point of collision, vectored in the direction of the axial vector to each plane of the bounding box are determined. With the inner normal information of the tissue surface, the vectors (from point  $P$  to the intersection point on the plane) which lie inside the tissue surface are identified. All three axial vectors which are overlapped in the tissue surface are added to find the final penetration vector.

The current velocities of the vertices of both triangles are obtained from the primitives and the instrument velocity and tissue velocity at the point of collision are calculated by using the areal coordinate technique. From all the above information, the final deformation vector is calculated by using the equation (3.8)

#### 4.3.4 Finding a penalty force

By using deformation vector and velocities of tissue triangle and instrument triangle at the point of collision, the penalty force to be applied to the objects to prevent interpenetration is calculated by using equation (3.9). This penalty force is later distributed to the vertices of tissue triangle by using the areal coordinate technique. The detail flow of this work is shown in figure

4.5

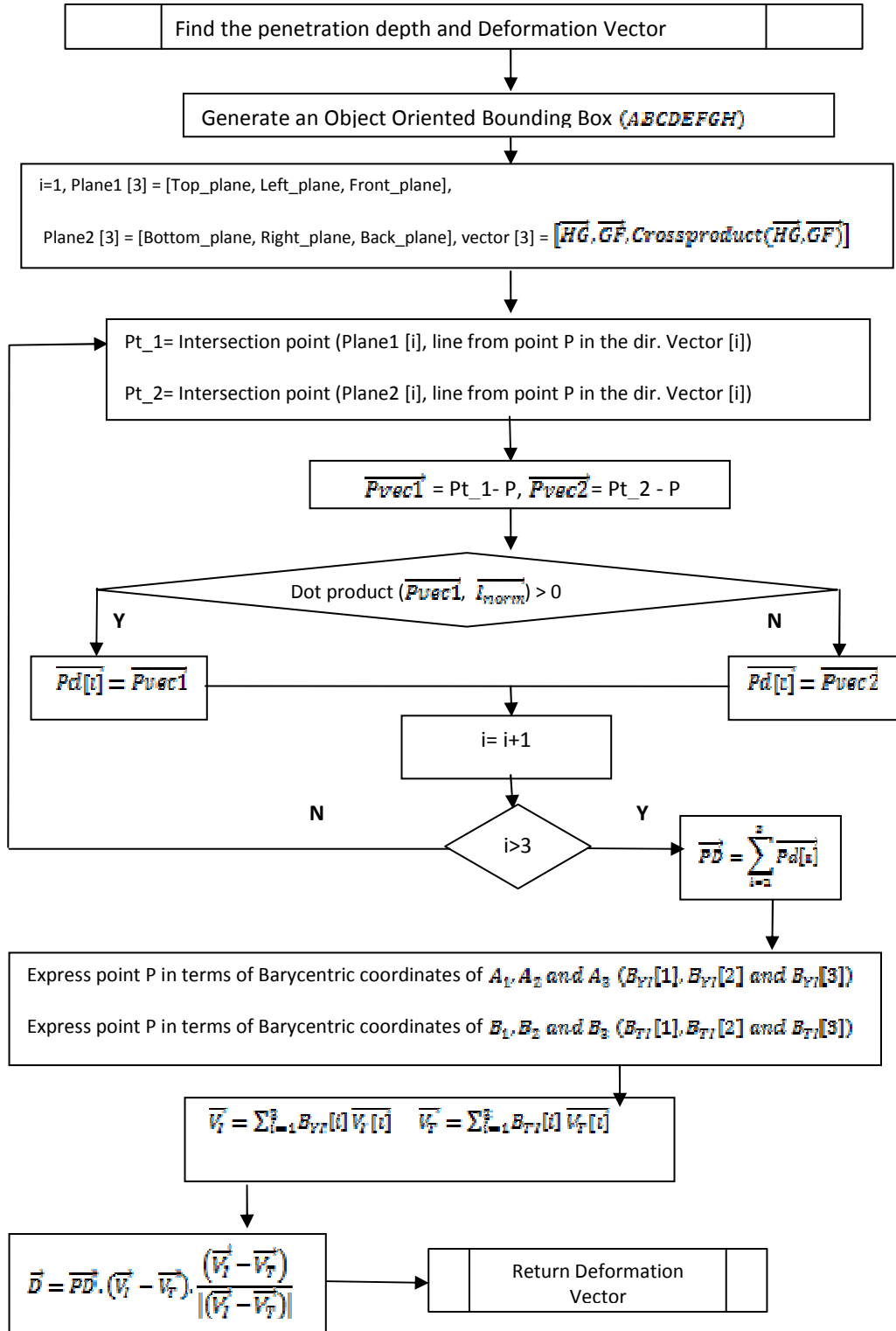


Figure 4.4 Flow chart of calculating deformation vector

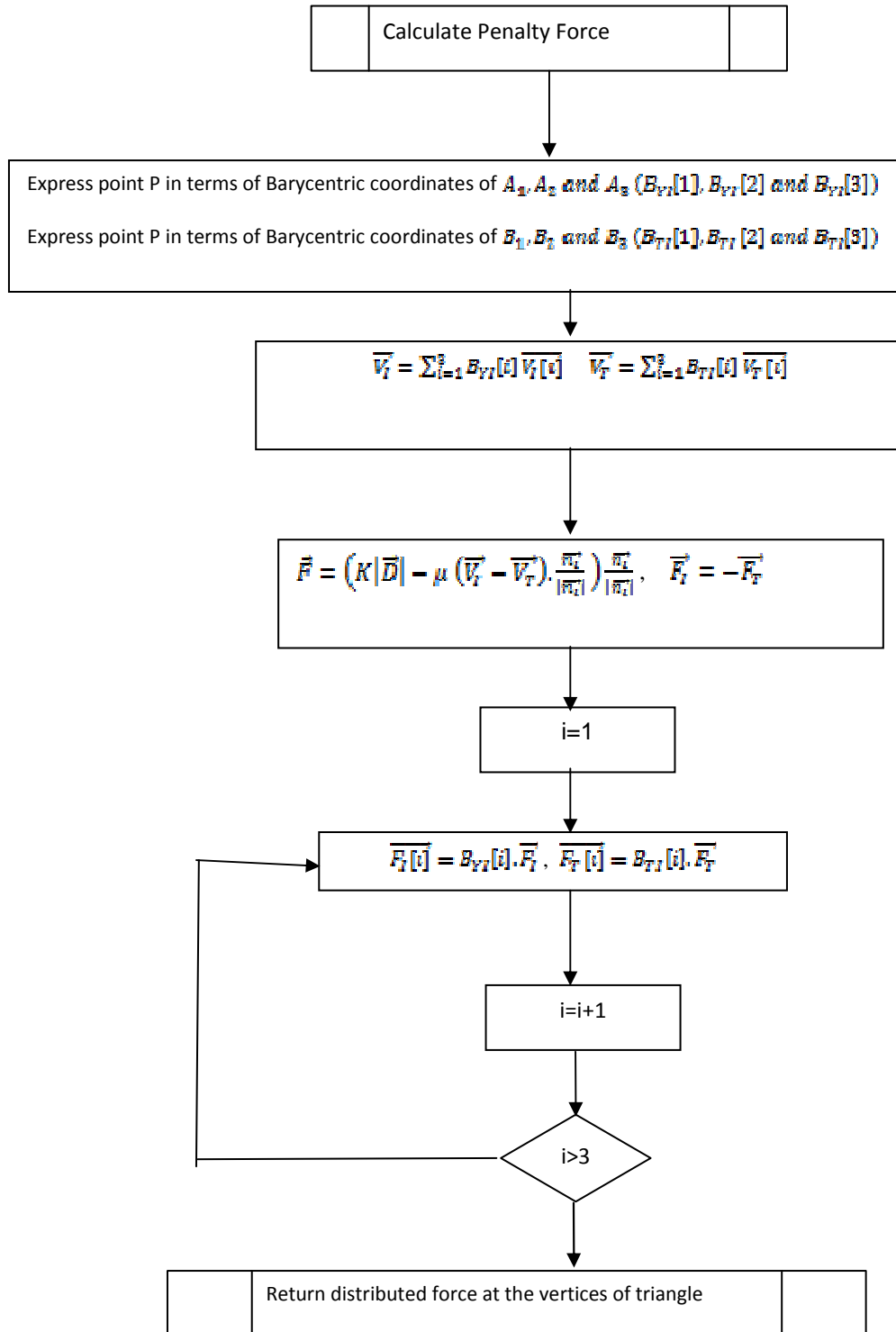


Figure 4.5 Flow chart of calculating distribute penalty force

## **CHAPTER 5**

### **RESULTS AND FUTURE WORK**

In this chapter, the graphical results of the proposed collision response algorithm and its statistical analysis are presented. A brief description of possible future work is also presented.

#### 5.1 Simulation details

The collision response algorithm is implemented on a Windows platform along with other existing modules. The simulation software modules are written in VC++.NET with OpenGL graphics, Glut, Glu libraries and a 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. The proposed collision response algorithm runs in the Collision Detection thread. The machine that was used for simulation runs has the following specifications and all the results that 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

The simulations were run on the above configuration and the graphical response of the proposed algorithm is shown in figure 5.1 to 5.3.

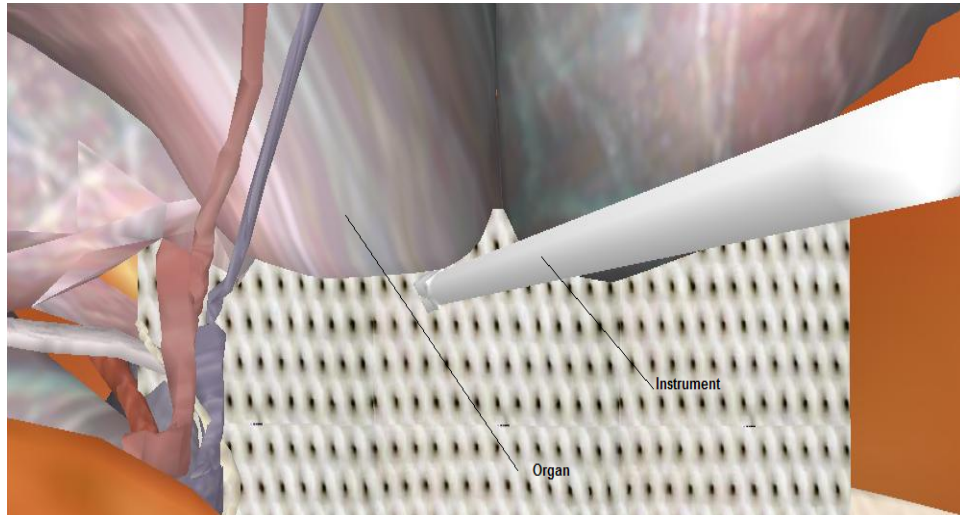


Figure 5.1 Graphical response of the proposed collision response algorithm instant 1

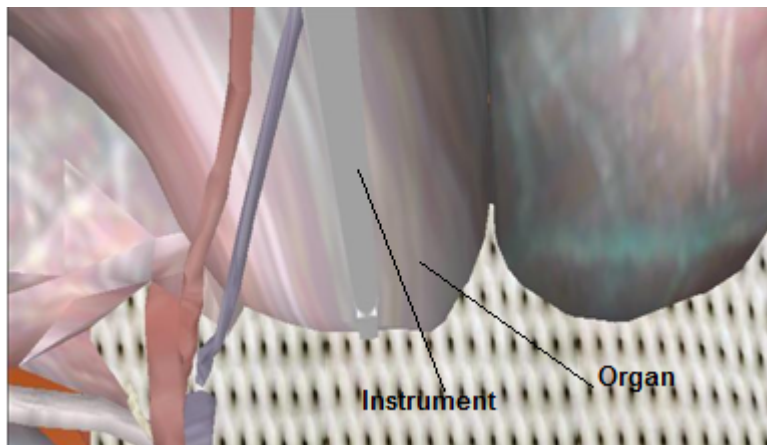


Figure 5.2 Graphical response of the proposed collision response algorithm instant 2

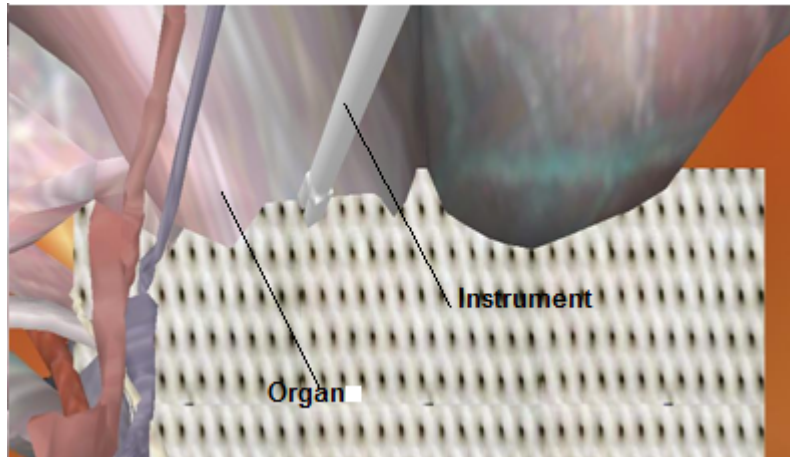


Figure 5.3 Graphical response of the proposed collision response algorithm instant 3

The timing diagram of the collision detection and collision response algorithm is shown in figure 5.4 and figure 5.5. The timing diagram shows that the average collision response time of the proposed algorithm is 1.4 micro seconds per iteration, which is significantly lower than the average time taken for the collision detection, i.e. 479.8 micro seconds.

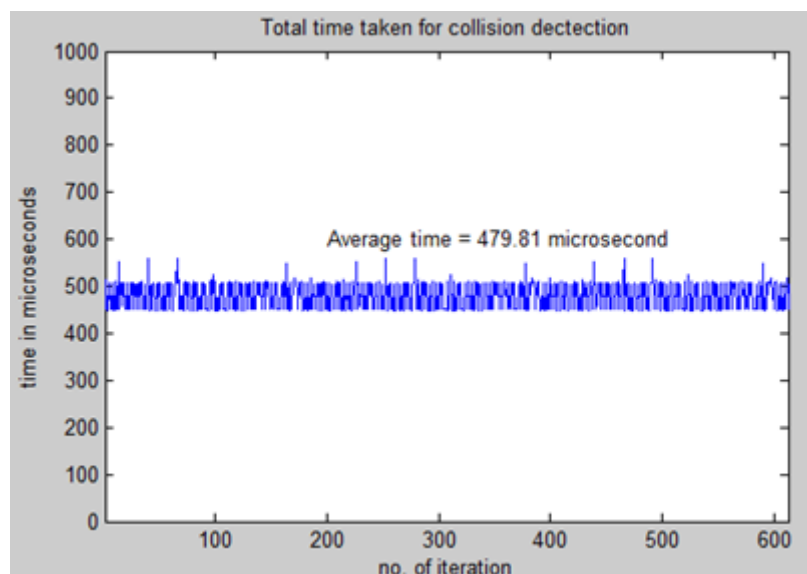


Figure 5.4 Timing diagram of the collision detection module



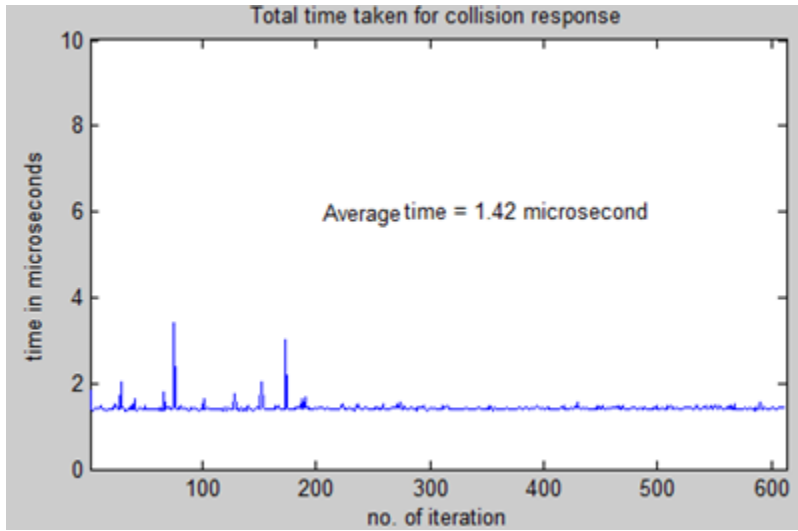


Figure 5.5 Timing diagram of the collision response module

### 5.3 Conclusion

The collision response of the deformable object is a very complicated issue mainly in the interactive haptic environment. A previously developed, best-in-class collision detection algorithm that we have used takes few milliseconds for detecting and reporting colliding pairs of primitives. Even this has very significant chance of missing the exact instant of collision resulting in interpenetration of objects. We have proposed, implemented and validated an innovative method using the bounding volume information of the object during runtime to overcome such interpenetration, specifically for instrument and deformable object collisions. The new algorithm takes in to account all possible scenarios of collision at the primitive level between an instrument and a tissue. The average collision response time of the proposed algorithm is in the range of microseconds and hence can be easily used to support the haptic feedback requirement for a surgical simulator.

#### 5.4 Future Work

The scope of the proposed collision response algorithm can be easily extended to other kinds of collision, i.e. instrument to instrument and instrument to rigid objects.

Currently, the virtual objects used in the surgical simulator are surface modeled objects. Hence the collision detection algorithm will almost always miss the exact instant of the collision. If tetrahedral volume modeled objects are used, then collision detection algorithm will not miss the exact of the instant of collision because of having lots of layer beneath the top surface layer of an object. The use of tetrahedral objects will however increase the computational cost and introduce other complications some known and some not. Therefore, new algorithms can be investigated that are based on tetrahedral models of organs and instruments.

Different kinds of discrete oriented polytopes can be used for bounding the instrument to show more accurate response by the collision response algorithm.

## REFERENCES

- [1] "Cato Unbound", Cato institute, Online available at <http://www.cato-unbound.org/contributors/jaron-lanier/>
- [2] US History Encyclopedia
- [3] "When you need an operation...About Hernia Repair", American College of Surgeons Online available at <http://www.slrsurgery.org/files/hernrep.pdf>
- [4] National Library of Medicine, MEDLINEplus. Online available at <http://digestive.niddk.nih.gov/statistics/statistics.htm>
- [5] K. Fuchs, "Minimally Invasive Surgery," Endoscopy, vol. 34, no. 2, pp. 154–159, 2002.
- [6] T. Tang, S. Dolan, B. Robinson, and L. Delbridge, "Does the surgical approach affect quality of life outcomes? A comparison of minimally invasive parathyroidectomy with open parathyroidectomy," International Journal of Surgery, Volume 5, Issue 1 , pp. 17 – 22, 2006.
- [7] The MRC Laparoscopic Groin Hernia Trial Group, Laparoscopic versus open repair of groin hernia: a randomized comparison," The Lancet, vol. 354, pp. 185-190, 1997.
- [8] J. Hureau, P. Vayre, V. Chapuis, M. A. Germain, J. L. Jost, J. Murat, and G. Spay, "Risk of laparoscopic surgery, 100 records of complications. a qualitative study," Chirurgie, vol. 121, no. 1, pp. 1-8, 1996
- [9] S. Bann, V. Datta, M. Khan and A. Darzi, "The surgical error examination is a novel method for objective technical knowledge assessment", The American Journal of Surgery 185, pp 507-511, 2003.
- [10] JD Seus and T. Wood, "Reaping maximum benefits from minimally invasive surgery", J Healthe Mater Manage, pp. 20-24, 1994.
- [11] TA Kenyon, DR. Urbach and JB Speer, "Dedicated minimally invasive surgery suites increase operating room efficiency", Surgical Endoscopy, pp. 1140-1143, 2001.

- [12] C. Basdogan, S. Kim, J. Manivannan, Muniyandi Kim and H. Srinivasan, "Haptics in Minimally Invasive Surgical Simulation and Training," IEEE Computer Graphics and Applications, vol. 24, no.2, pp. 56-64, 2004.
- [13] Online Available at <http://www.nlm.nih.gov/research/visible/>
- [14] W. Lorrenson and H. Clin, "Marching Cubes: A high resolution 3D surface construction algorithm," Computer Graphics, vol. 21, no. 4, pp. 163-169, July 1987.
- [15] V. Gupta, "Extraction of realistic anatomical texture from visual human data for laparoscopic herniorrhaphy", Master's Thesis, University of Texas at Arlington, August 2003.
- [16] S. Kapdoskar, "Simulation of inguinal hernia condition using visible human data and 3D modeling techniques for virtual laparoscopic herniorrhaphy", Master's Thesis, University of Texas at Arlington, August 2003.
- [17] L. Raghupathi, "Simulation of bleeding and other special effects for virtual laparoscopic surgery", Master's Thesis, University of Texas at Arlington, December 2002
- [18] A. Gande, "Instructor Station for virtual laparoscopic surgery", Master's Thesis, University of Texas at Arlington, August 2003.
- [19] H. Z. Tan, M. A. Srinivasan, B Eberman and B. Cheng, "Human factors for the design of force-reflecting haptic interfaces', ASME Dynamic System and Control Division, DSC-55, pp. 353-359, 1994.
- [20] Phantom from Sensable Technology, Online available at <http://www.sensable.com/products-haptic-devices.htm>
- [21] Y. Shen, V. Devarajan and 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.
- [22] Nucleus medical Art, "Inguinal hernia", Online available at [www.nucleusinc.com](http://www.nucleusinc.com)
- [23] Health central Network "Inguinal hernia repair – series", Online available at <http://www.healthcentral.com/ency/408/presentations/100027.html>

- [24] J. Butala, "Collision Response for laparoscopic surgery", Master's Thesis, University of Texas at Arlington, August 2005.
- [25] X. Wang, V. Devarajan, "Improved 2D mass-spring damper model with unstructured mesh", *The Visual Computer*, pp. 57-75 , August 2005.
- [26] V. Kalaiselvsn, "Extent of force propagation with hybrid numerical integrators in deformable tissue modeling" Master's Thesis, University of Texas at Arlington, December 2003.
- [27] Picture of haptic device PHANTOMTM device used in Virtual Environment lab, University of Texas at Arlington
- [28] F. Benedetti, "Physical response to the collision of the deformable bodies", Post graduate course of computer visualization and communication, Swiss Federal Institute of Technology, 2002.
- [29] D. Baraff, "Analytical Methods for Dynamic Simulation of Non-Penetrating Rigid Bodies", *ACM Computer Graphics*, 23 (3): pp. 223-232, July 1989.
- [30] D. Baraff, "Curved Surfaces and Coherence for Non-Penetrating Rigid Body Simulation", *ACM Computer Graphics*, 24 (4): pp. 19-28, 1990.
- [31] F. Faure, "An Energy-Based Method for Contact Force Computation", *Proceedings of Eurographics'96, Computer Graphics Forum*, Vol. 15, No. 3, pp. 357-366, 1996.
- [32] J. Mezger, S. Kimmerle and O. EtzmuB, "Progress in Collision Detection and Response Techniques for Cloth Animation," 10th Pacific conference on Computer Graphics and applications, pp. 444, 2002.
- [33] A. Witkin, K. Fleischer and A. Barr " Energy Constraints on parameterized models", *Proc. SIGGRAPH* pp 225-232, 1987.
- [34] D. Terzopoulos, J. Platt and A Barr " Elastically Deformable Modles", *Proc. SIGGRAPH* pp. 205-214,1987.
- [35] J.C. Platt and A. H. Barr, "Constraint Methods for Flexible Models", *Computer Graphics, Proc. SIGGRAPH*, Vol. 22, No. 4, pp. 279-288, August 1988.

- [36] M. Moore and J. Wilhelms, "Collision Detection and Response for Computer Animation", *Computer Graphics*, Vol. 22, No. 4, pp. 289-298. August 1988.
- [37] D. Baraff, "Non-penetrating Rigid Body Simulation", *EUROGRAPHICS'93 Barcelona*, September 6-10, 1993.
- [38] M. Desbrun, P. Schröder and A. Barr, "Interactive Animation of Structured Deformable Objects" In *Graphics Interface*, pp. 1-8. 1999.
- [39] M. McKenna and D. Zeltzer, "Dynamic simulation of autonomous legged locomotion", *Computer Graphics Proc. SIGGRAPH*, Vol. 24, pp. 29-38, ACM, August 1990.
- [40] J. Jansson and J.S.M Vergeest, "A General Mechanics Model for Systems of Deformable Solids", *Proceedings International Symposium On Tools and Methods for Competitive Engineering (TMCE 2000)*, pp. 361-372, 2000.
- [41] G. Hirota, S. Fisher and A. State, "An improved finite-element contact model for anatomical simulations", *The Visual Computer*, vol. 19, pp. 291–309, Springer, 2003.
- [42] B. Heidelberger, M. Teschner, R. Keiser, M. Mueller and M. Gross, "Consistent penetration depth estimation for deformable collision response", *Proc. Vision, Modeling, Visualization*, pp. 339–346, 2004.
- [43] J. K. Hahn, "Realistic Animation of Rigid Bodies", *Computer Graphics Proc. SIGGRAPH*, Vol. 22, No. 4, pp 299-308. August 1988.
- [44] B. Mirtich and J. Canny, "Impulse-based Dynamic Simulation", In *Proc. of Workshop on Algorithmic Foundations of Robotics*, pp. 402-418, February 1994.
- [45] M. S. Norhaida, A. Bade, D. Daman and M.S. Sunar, "The development of deformable bodies collision response algorithm for interactive virtual environment", *Project Report University Technology Malaysia*, 2007.
- [46] J. Spillmann and M. Teschner, "Contact Surface Computation for Coarsely Sampled Deformable Objects", *Vision, Modeling and Visualization*, Erlangen, Germany, November 16–18, 2005.

- [47] O. Etzmus, B. Eberhardt, M. Hauth and W. Straser, "Collision Adaptive Particle Systems," Proc. The Eighth Pacific Conference on Computer Graphics and Applications, pp. 338, 2000.
- [48] V. Vuskovic, M. Kauer, G. Székely and 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.
- [49] P. Sovis, "Collision Detection and Impulse Dynamics in Real time Applications", CESCg, 2000, also available online at <http://www.cescg.org/CESCg-2000/PSovis/index.html>
- [50] B. Mirtich, "V-Clip: fast and robust polyhedral collision detection", ACM Transactions on Graphics, Volume 17, Issue 3, pp. 177 - 208, 1998.
- [51] G. Grabner and A. Kecskemethy, "Reliable Multibody collision detection using runge-kutta integration polynomials", In Multi-body dynamics, pp. 301-316, 2005.
- [52] V. García-Pérez, E. Muñoz-Moreno, R. Luis-García and C. Alberola-López, "A 3D Collision Handling Algorithm for Surgery Simulation Based on Feedback Fuzzy Logic" In Proceedings of the IEEE EMBS International Conference on Information Technology Applications in Biomedicine (ITAB'06), "Best Student Paper Award", Ioannina, Greece, 2006.
- [53] D. Comin and O. Staadt, "Velocity-Aligned Discrete Oriented Polytopes for Dynamic Collision Detection", IEEE Transaction on visualization and computer graphics, pp. 1-12, 2008.
- [54] M. Lin and S. Gottschalk, "Collision Detection between Geometric Models: A Survey," Proc. of IMA Conference on Mathematics of Surfaces, pp. 37-56, 1998.
- [55] X. Wang, "Deformable Haptic Models for Surgical Simulation", PhD Dissertation, University of Texas at Arlington, August 2005.
- [56] J. Brown, K. Montgomery, J. C. Latombe, and M. Stephanides, "A microsurgery simulation system", Medical Image Computing and Computer-Assisted Interventions, pp. 137-144, 2001
- [57] S. M. Platt and N. I. Badler, "Animating facial expressions", Proc. SIGGRAPH'81, ACM Press, pp. 245-252, 1981.

- [58] K. Waters, "A muscle model for animating three-dimensional facial expression", ACM Computer Graphics, vol. 21, no. 4, pp. 17-24, July 1987.
- [59] J. Zhang, S. Payandeh, and J. Dill, "Haptic subdivision: an approach to defining level-of-detail in haptic rendering", in Proc. 10th Symposium on Haptic Interfaces for Virtual Environment and Tele-operator Systems, pp. 201-208, 2002.
- [60] X. Wang, V. Devarajan, "1D and 2D structured mass-spring models with preload", The Visual Computer, pp 429-44 , August 2005.
- [61] G. V. D.Bergen, "Efficient collision detection of complex deformable models using AABB trees", Journal of Graphics Tools, pp .1-14, 1997.
- [62] S. Gottschalk, M. Lin and D. Manocha, "OBB-Tree: A hierarchical structure for rapid interference detection", SIGGRAPH 96 Conference Proceedings, Aug. pp. 171-180, 1996.
- [63] G. Zachmann, "Minimal hierarchical collision detection", Proc. ACM Symposium on Virtual Reality Software and Technology, pp. 121–128, Nov 2002.
- [64] T. Möller, "A Fast Triangle-Triangle Intersection Test", Journal of Graphics Tools, vol. 2, no. 2, pp. 25-30, 1997.
- [65] Barycentric Technique, Online available at <http://www.blackpawn.com/texts/pointinpoly/default.html>



## BIOGRAPHICAL INFORMATION

Dibbesh S. Adhikari was born on 1<sup>st</sup> January, 1981 in Kathmandu, Nepal. He completed his schooling from Bhanubhakta Memorial Higher Secondary School, and received his Bachelors degree in Electrical and Electronics with specialization in Communication from Kathmandu University in November 2002. He worked as a Telecom Engineer in United Telecom Limited, Nepal from March 2003 to December 2005. He then pursued higher studies at The University of Texas at Arlington, from January 2006 and received a Master of Science (MS) degree in Electrical Engineering in December 2008. He worked as an intern in Ericsson Inc. (Ericsson Mobile Platform) from August 2007 to May 2008 and as a consultant for Ericsson Mobile Platform from September 2008 to present. His research interests are telecommunication, wireless communication and algorithm development. He is passionate about music and sports.