

Chapter 1 Introduction

1.1 Research Background

Virtual Reality (VR), a new era of technology that allows interactions with human sensory organs and cognitive systems (Steffin 2001). It is the simulation of a real or imagined environment that can project visually in a three-dimensional perspective. It additionally provides an interactive experience such as real time motion with sounds and other forms of feedback. According to Morris Steffin (2001), it is a global approach toward the temporary fusion of experience and function within an artificial environment in which links to reality that fall to a large extent under the control of the VR designer.

Towards these ends, VR systems can produce a high level of immersion by application of sophisticated input/output devices. These include wide-field stereoscopic head-mounted displays for audio and visual stimuli with controllable modification of video and audio in accordance with user's movement. Haptic devices can also provide reactions to a subject's movements by producing forces simulating to those generated within an actual subject and real life environment.

The development of virtual reality in different kind of applications such as engineering, medical, education and others has been carried out for so many years to provide great advantages and improvement in those areas.

Virtual surgery is one of the virtual reality developments that has recently become a popular technique for both diagnosis and treatment of many kinds of human diseases and injuries. Traditional surgical training requires immediate encounter

between the physician and patients. The mechanical facets of surgical technique, including the identification of anatomic landmarks, instrument manipulation, and reaction to changes in the surgical operational environment, require patients and interaction with an experienced surgeon-teacher. With the VR training paradigms, the trainee-surgeon can perform unlimited practice until they are able to demonstrate sufficient manual and visuospatial adaptation to warrant treating the actual patients.

Among all the surgical procedures, laparoscopic cholecystectomy is always the first common surgical procedure learned by surgeons (Webster and Haluck 2003). Since this surgery can be considered as the 'entry level' of laparoscopic operation, they are always a potential to have disastrous complications, for example, bile duct leak or misperformed injuries. Hence there is a necessary to develop a surgical simulator to let the surgeons to perform enough practices before they could enter the surgery operation rooms.

Currently, there are a lot of researchers who have developed subtasks or modules in the laparoscopic cholecystectomy simulation. Although a number of training simulators are available now, but most of them require very high-end platform of operating system, hardware and software, maintenance or availability. The cost of development is often beyond financial reach of research centers, clinics or hospitals, not to talk about single individual such as students. These will limit the trainee-surgeon to obtain and practice particular skills whereby they are only allowed to practice their skills if the hospital able to afford to have the high requirement of a surgical simulator.

In order to provide this facility within an affordable range, it is possible to develop a low cost PC-based virtual reality laparoscopic cholecystectomy training system.

1.2 Statement of research problem

The primary aim of this study is to develop a PC based VR prototype that is capable to simulate the simple surgical task on a laparoscopic cholecystectomy.

Thus, the following problems will be addressed:

- a) Understanding the procedure of performing the laparoscopic cholecystectomy.
- b) Designing an approach to determine collision management
- c) Performing the simple operational tasks that include cutting and grabbing by using virtual surgical tools.
- d) Development of the system.
- e) Evaluation of systems design and performance

1.2.1 Understanding the procedures involved in laparoscopic cholecystectomy

The main focus of this study is to develop the simulation. However, developing a system without understanding the procedures that involve in the real laparoscopic cholecystectomy surgery will lead to the failure in designing a good system.

Since the main concern of this study is on the cutting and grabbing in this surgery, the procedure of these two sections including collision management should be clearly defined.

1.2.2 Designing an approach to determine collision management and performing simple surgical task

The backbone of this study will depend largely on the method of determining the collision detection. All the surgical tasks such as cutting and grabbing by using virtual surgical tools need proper collision detection technique to determine the boundary between the surgical tools and virtual skins before performing the tasks. There is a need to significantly determine when collision detection is happening on the virtual surgical tools so the trainee-surgeon will be able to handle the tools more efficiently.

After understanding the procedures involved, different methodologies will be applied to transform the real movements into virtual reality by using suitable programming languages and associated libraries of subroutines.

1.2.3 Development of the system

The methodologies decided upon, the next step would be towards designing the system. For this stage, all the flows that are involved in the study would be determined to develop the system an operationally success.

The components that were used to develop the system should be defined clearly.

1.2.4 Evaluation of systems design and performance

The testing of the completed prototype will be carried out on the laparoscopic cholecystectomy especially on the functions and surgical tasks that had been mentioned above. Besides, testing will also be performed on different types of

image models especially on the functions for particular surgical tasks that will be developed in this prototype.

1.3 Objectives of study

The objectives of the study are as follows:

- a) Develop a desktop environment for laparoscopic cholecystectomy simulation. The virtual environment was developed by using WorldToolKit (SENSE 8 Corporation) and consists of virtual laparoscopic tools and human organ model. This environment is capable to operate under the Microsoft's Windows operating System.
- b) To enable the user to perform the virtual surgical task especially cutting and grabbing on laparoscopic cholecystectomy surgery in virtual environment, using the virtual laparoscope tools.
- c) To study the collision detection and response method that can be applied in a laparoscopic cholecystectomy operation environment.
- d) To study and implement the grabbing and cutting methods in laparoscopic cholecystectomy.

1.4 Scopes and limitations of the study

Some scopes and limitations that involved in this study are as follows:

- ❖ The medical images which was used in this project is in 3D Studio Max format and consists of small video-framed size due to limitation of processing unit.

- ❖ The focus is only on the laparoscopic cholecystectomy surgical simulation that involves cutting and grabbing tasks.
- ❖ The simulation was displayed using the screen and controlled by a mouse and keyboard. Force feedback is not included in this study.

1.5 Motivation of Study

In 1965, Robert Mann proposed the first medical virtual reality system that was developing a rehabilitation application for virtual reality. The system allows the surgeon to try multiple surgical approaches for a given orthopedic problem. (Mann 1985)

Currently, surgeons are trained during actual operations or in the animal laboratory to obtain the necessary skills. However, training in the operating theatre increases the risk to the patient and slows the operation, resulting in greater cost. At the same time animal training is expensive and cannot duplicate human anatomy.

Besides that, in minimally invasive surgery, learning laparoscopic technique is much more difficult for surgeons than learning open surgery procedures. This is due to difficulties in coordination of multiple instruments in several tiny punctures. The range of task that can be performed is always limited by the manual dexterity of the surgeon, hand tremor and lack of kinaesthetic feedback.

Furthermore, the vision on the internal organ is monoscopic and is always limited by the field of view of the wide-angle camera. The laparoscopic instruments rotate about a fixed point which normally is the point where the trocar is inserted into abdomen and it is not possible to make direct translational movement while interacting with the organs. (Basdogan *et al.* 2001)

Hence, these could be a considerable merit in developing a new training approaches or devices such as desktop environment of surgical simulator to help the surgeons to practice their skills and monitoring their performance especially in laparoscopic surgery.

1.6 Significance of study

The field of medicine is changing and advancing as fast as computer technology. New procedures and techniques are appearing every year. Existing techniques and procedures may have to be changed in order to provide better treatment for the patients, hence virtual reality medical applications are being developed to enhance the skills of doctors.

Developing any medical VR application take a longer time than industrial-based application because the need for accuracy is higher and it is more difficult to produce. Hence, researchers use very high-end technologies to simulate the surgery procedures rather than using medical devices and equipments. However, this status will limit doctor's learning process, as they are only able to study in the learning centers which will have to allocate finance to obtain the simulators for the learning process.

Flexibility is also a major concern in the learning process. It is important to develop a simple and low requirements simulator in term of hardware and software configuration, so doctors are able to practice their skills in a more conducive environment.

In Malaysia, medical students are still continuing the traditional trends of completing a medical education course in five years time (Khalid 2002). Three

years are normally set aside to the study of clinical medicine and two years to obtain appropriate skills in a teaching hospital.

According to a medical professional (Khalid 2002), skills for medical practice can only be learnt through properly planned practical training in a representative range of health problems. Many hospital cases are definitely not representative of health problems as they are more often regarded as acute serious cases and long-standing illnesses. Further, medical teaching is more inclined towards the care of inpatients. Moreover, the junior surgeons develop their skills through literature review, books, lectures, observation and by performing the procedure under the supervision of an experienced surgeon.

Owing to this natural learning process, the quality of medical education becomes unpredictable. This is because it is mainly depended on the guidance of an instructor and the cases that are exposed to the students during trainings. Furthermore, studies have found that the outcome of surgery is significantly worse on the first procedures performed by an inexperienced surgeon (Davies and Campbell, no date].

Based on the situations above, there is definitely a need to develop medical-based simulators for students to obtain the necessary effective skills besides the investigation, treatment, and observation of inpatients to supplement the experience within a hospital environment.

In conclusion, an immediate potential benefit from virtual reality systems lies in improving teaching and training procedures for doctors. Virtual Reality will have medical application opportunity not only presently but in the future too.

1.7 Summary of chapters

This section provides a brief description of the various chapters.

- a) Chapter One covers the introduction to the research. It provides the definitions and objectives of this study. The justifications for undertaking this study are also elaborated.
- b) Chapter Two provides various information and technology about virtual reality, with examples of virtual reality technologies specifically in the medical field. There is also a review of the literature on steps and information on laparoscope surgery and minimally invasive surgery. Besides, the review also includes the methods that had been implemented for cutting and deformation. The latter information covers several aspects, including the applications of virtual surgery and its development tools.
- c) Chapter Three elaborates the methodological processes that have to be followed in the pursuit of designing a surgery system with medical images. This chapter will discuss the geometry of the graphical model and viewpoint and orientation involved in this study. Discussion also involves the manipulation of the surgical tools. Method used in determining the collision detection between surgical tools and organ surface before performing surgical task and responses after collision detection in term of visual, audio display and message prompt become major concern in this chapter. Besides, the grabbing, cutting method and design outlines will be discussed too.
- d) Chapter Four explains the development and implementation for the system. This chapter proceeds with a description of the method in implementing the

system using the said development tools. This chapter will also report the result of the research with a discussion on several aspects of the presentation of this system, verification and testing results of the systems medical imaging application.

- e) Chapter Five concludes the research by summarizing the main features of this research effort. The strengths and limitations of the system are also listed and assessed. Future enhancements of this research are suggested.

Chapter 2 Literature Review

2.1 Introduction

In deciding to proceed with this proposed simulation, a literature review is performed on the existing technologies, including the laparoscopic surgery and virtual laparoscopic surgery.

The description on the minimally invasive surgery, laparoscopic surgery such as laparoscopic cholecystectomy in real medical practice will be explored to provide a better understanding on the procedures involved in this surgery. This study will also touch on the techniques and methods that are involved in this form of surgery.

This is followed by an attention on virtual reality in medicine. The purpose of the conversion from the real surgery to the virtual environment will be reviewed too. Components advantages and disadvantages of the virtual reality, especially in the medical field, will be discussed. This review will also include the existing VR researches and technologies that have been developed by other researchers and institution especially in medical area.

Finally, an overall view of the appropriate tools that are used to develop the simulation systems will be taken into consideration in this literature review.

2.2 Minimally Invasive Surgery

Minimally invasive surgery (MIS) is a relatively new technique in which a surgeon operates with specially designed surgical tools through access ports requiring incisions of about 1cm in size. The miniature cameras (video laparoscopes) are

placed within the body through this incision and the resulting images will be projected onto a monitor screen for enlargement and easier to see. The instruments enable the surgeon to manipulate the internal organs by using finger loops outside the patient's body linked to the tool tip via a long tube including internal mechanism. (Rosen 1999)

By eliminating the large incision and extensive dissections, the benefits include the limitation of the surgical trauma and damage to healthy tissues, decrease the pain that the patient can experience, less infection, less cosmetic damage and results in a significant shortened recovery period.

Type of Minimally invasive surgical techniques used: (Anon 2001)

❖ **Hysteroscopy**

A procedure uses hysteroscope to view the inside of the uterus through the cervix. A liquid or gas may be introduced into the uterus so the physician has a better view. Additional small surgical instruments may be introduced to allow for the diagnosis as well as treatment of gynecological conditions.

❖ **Laparoscopy**

A surgical procedure that involving specialized devices such as laparoscope that enters to the patient's abdomen and pelvis. The laparoscope is introduced into the abdomen through a small incision (1 to 3mm) or ports made inside or below the navel in order to enable the physician to visualize the surgery through video display of the surgical site. Highly sophisticated cameras, monitors and robotics are used for these procedures. The

physician may also make one or more additional small incisions so he or she can pass surgical instruments or lasers into the abdomen.

Laparoscopy is typically performed under general anesthesia. During the procedure, the patient's abdomen will be filled with a gas such as carbon dioxide or nitrous oxide to keep the walls separated, so the physician gets a better view of the organs. After the procedure is complete, the incision will be sutured and covered with a small surgical strip. Sometimes, patients are surprised that they can't even find their incisions, because they are so small.

Patients can normally return to work and perform other everyday activities in a day or two compared to the 6-8 weeks required with traditional open abdominal surgery.

❖ **Cystoscopy**

Cystoscope is used to view the inside of the bladder through urethra in this procedure. Saline is introduced through the cystoscope to fill the bladder allowing the surgeon to view the bladder wall. Cystoscopy is a short procedure that takes only 15 to 20 minutes. Patient normally may feel sore while urinating for a day or two, but are able to return to regular activities after the procedure.

2.3 Laparoscopic cholecystectomy (LC)

Laparoscopic cholecystectomy, involves removal of gallbladder due to bile duct or injury, is the most commonly performed Minimally Invasive Surgery by a General surgeon. In 1985, Erich Muhe in Germany performed the first documented

laparoscopic cholecystectomy (Misra 2003) and P.Moruet at Lyon performed the first laparoscopic cholecystectomy in 1987.

Gallbladder is part of the digestive system that stores and secretes the bile salts. These salts are useful in the process of breaking down food, especially fatty foods into its adsorptive components. Inflammation of the gallbladder arises when the flow of bile is stopped or interrupted due to gallstone or if infection of biliary tract occurs. This is known as cholecystitis. Currently, LC is the best standard treatment for cholecystitis. (Misra 2003) This is because LC treatment is able to reduce the risk of infection due to a small cut. The patient is able to recover fast and experience less post-operative pain.

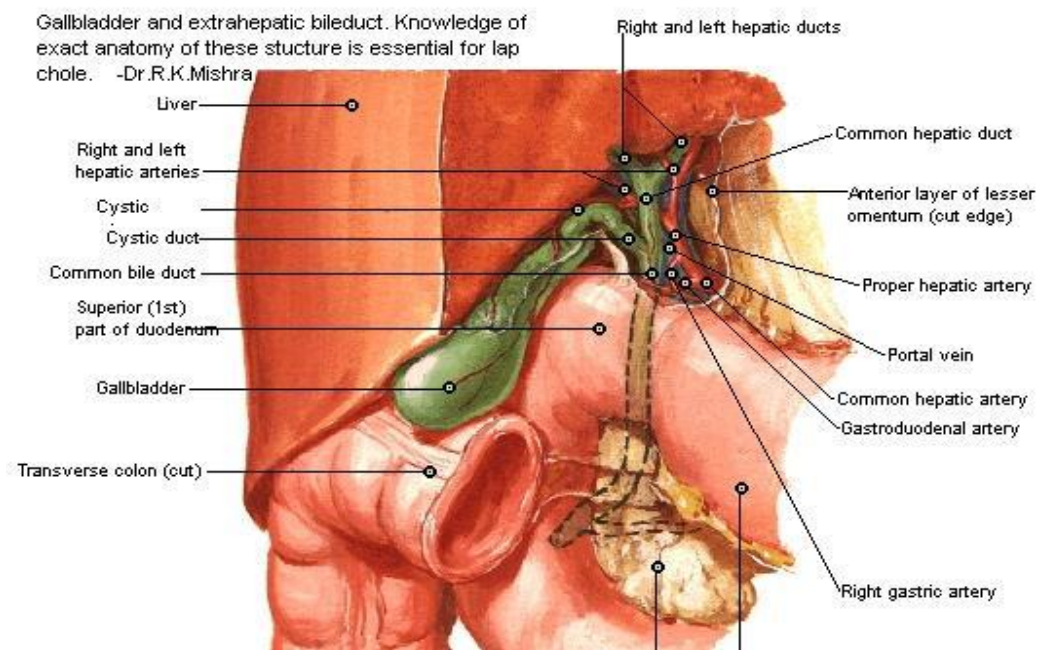


Figure 2.1: Anatomy of Gallbladder

2.3.1 Procedure (Misra 2003) (Bolton, no date)

Laparoscopic cholecystectomy did not involve complicated procedures. The entire procedure normally takes around 30 to 60 minutes. Some basic techniques are briefly discussed:

a) Insufflations

First, the surgeon will make a tiny incision at the navel and a thin tube carrying the video camera is inserted. A standard four-port access has been obtained, comprising one 10mm umbilical port, one 10mm epigastric port, and two 5mm right upper quadrant ports.

Next, the surgeon inflates the abdomen with carbon dioxide, for easier viewing and to provide room for the surgery to be performed. Two needle-like instruments are inserted to serve as tiny hands within the abdomen to perform tasks such as pick up the gallbladder, move intestines around, and generally assist the surgeon.

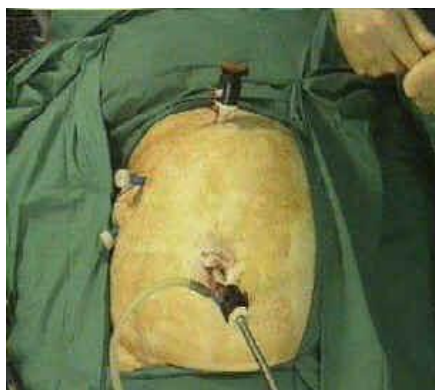


Figure 2.2: Standard four-port will be obtained during insufflations.

b) Dissection

During a laparoscopic gallbladder operation, the surgeon will use the grasper to expose the triangle of Calot. Then follow by dissection to the cystic pedicle with two-handed technique. Cystic pedicle is a triangle fold of peritoneum containing the cystic duct, artery, the cystic node and a variable amount of fat. The dissection was done with downward traction by duckbill grasper placed on the anterior edge of a Hartmann's pouch.

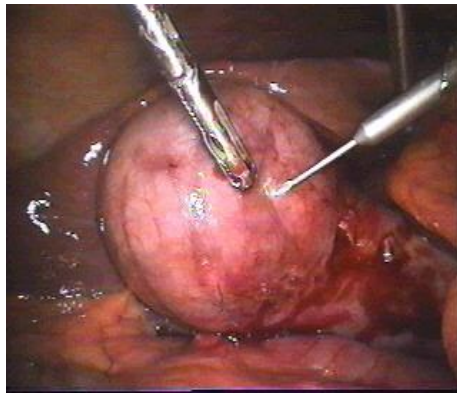


Figure 2.3: Apply grasper clamps for dissection and manipulation

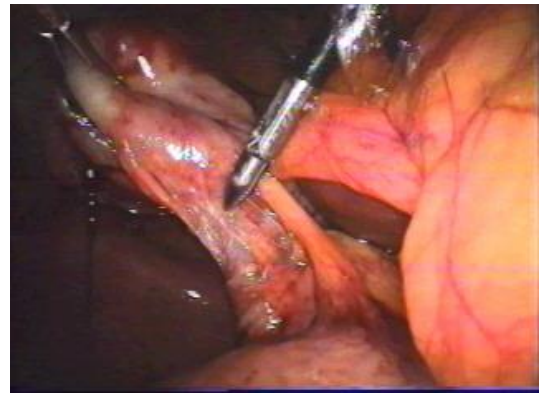


Figure 2.4: Triangle of Calot

Figure 2.3 shows that a needle is used to drain some bile so grasper clamps can be applied for dissection and manipulation. Meanwhile figure 2.4 shows that the Hartmann's pouch is retracted laterally and upward, exposing the triangle of Calot.

Then, the surgeon will divide superficially the peritoneum of the superior leaf of the cystic pedicle as far as back as the liver. A pledget mounted securely in a pledget holder is then used for blunt dissection. Maryland's grasper or electro surgical hook knife will be used to separate the cystic duct interiorly from the

cystic artery. Sufficient length of the cystic duct and artery on the gallbladder side should be mobilized.

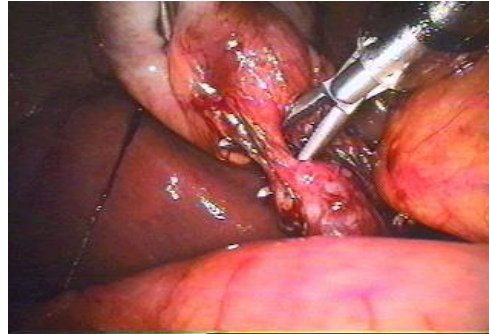


Figure 2.5: A "right angle" clamp is used to dissect a short cystic duct.

b) Ligation

Ligation will be involved in two sections:

i. Ligation of cystic artery

Two clips are placed proximally on the cystic artery when arises from a looped right hepatic artery and divided separately by hook scissors. The dissection of the cystic pedicle is considered complete by placement of a clip to occlude the cystic duct at its junction with the gallbladder.

ii. Ligation of Cystic Duct

Tie the cystic duct using catgut Roeder external slipknot. Clipping the cystic duct is not encouraged due to formation of stone from internalization clip.

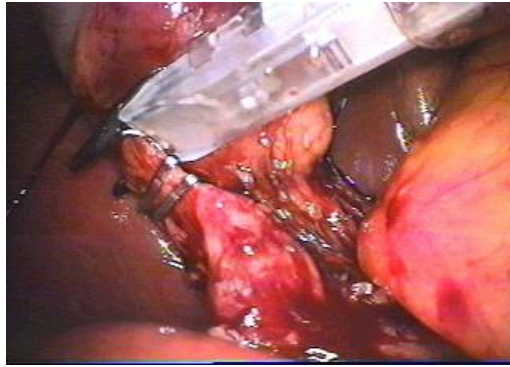


Figure 2.6: Clips are applied to the cystic duct.

c) Extraction

After ensuring that there is no bleeding or injury, the gallbladder can be extracted through the operating port. If this is not possible, the gallbladder maybe pulled to the abdominal skin and emptied by suction. However, it is recommended that the extraction be done inside a bag as a safeguard if the surgeon is concerned that the gallbladder may rupture during extraction, or if the gall bladder has been damaged during its resection.

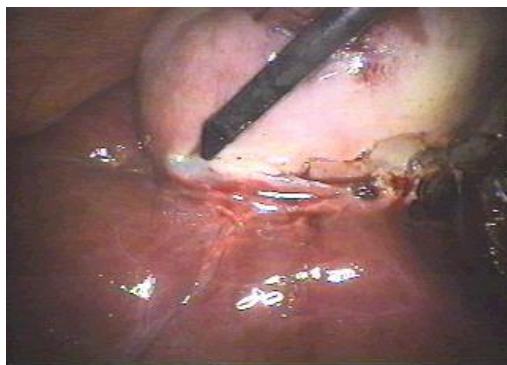


Figure 2.7: Hook electrocautery is used to dissect the gallbladder off the liver bed

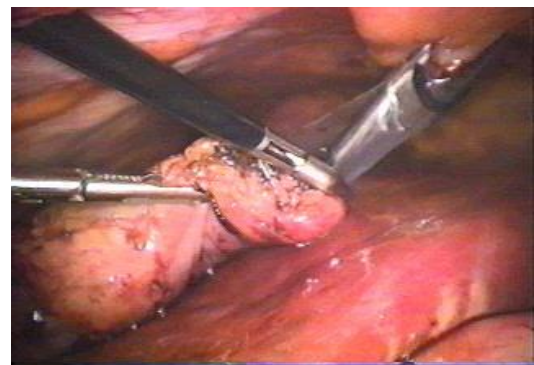


Figure 2.8: The gallbladder is now free and placed into a Pleatman sac for retriever

d) Suturing

After examining the abdomen from any possible bowel injury, remove the instrument and telescope from the port, suturing will be involved in order to close the wound. Suturing uses a small curved needle to swallow enough curves to allow it to pass down a 10 mm port then grasping the suture proximal to the junction with the needle and passing it down the port. Once the needle is visible it is grasped by an instrument in the left hand and held in the correct orientation for the needle holder to grasp and suture. Suturing uses Vicryl for rectus and Un-absorbable intra-dermal or staler for skin.

2.3.2 Benefit

The main benefit of this procedure are shown as below:

- ❖ Fast and ease of recovery for the patient.
- ❖ No incision pain as compared with standard abdominal surgery.
- ❖ The 90% of the patient can recover and out patient about the same day (Jackson Gastroenterology, 2002). The rest are usually discharged the next day.
- ❖ Within several days, normal activities can be resumed.
- ❖ There is no scar on the abdomen.

2.3.3 Complication

Even though the gallbladder removal procedure seems very simple for the patient, it still carries the same risks as general surgery. Current medical reports (Jackson Gastroenterology 2002) indicate that the low complication rate is about the same

for this procedure as for standard gallbladder surgery. These complications may include:

- In about 5 to 10% of cases, the gallbladder cannot be safely removed by laparoscopy. Standard open abdominal surgery is then immediately performed.
- Nausea and vomiting may occur after the surgery.
- Injury to the bile ducts, blood vessels, or intestine can occur, requiring corrective surgery.
- A diagnostic error or oversight is quite uncommonly to occur.

2.4 Virtual Surgical Training

Virtual reality's biggest contribution to medicine might be providing virtual surgical training in improving doctors' learning curve. It is not only used to improve the practical skills, but also used to teach new advances and procedures in surgery.

Surgical simulators can be used to ensure that minimum standards are met prior to performing the next level of training. New and complex procedures can be practiced safely on a simulator before proceeding to the real patient. Simulators can decouple the physician-student time dependency. The student can practice on his own schedule. Practice sessions can be stored for later review by the physician.

(Liu *et al.* 2003)

2.4.1 Components of Surgical Training Simulator

According to Satava (1993) (Moline 1998), there are five elements that affects the realism of a virtual environment for medical applications:

- ❖ Fidelity - high resolution graphics
- ❖ Display of organ properties- such as deformation from morphing or kinematics of joints
- ❖ Display of organ reactions- such as bleeding from an artery or bile from the gallbladder
- ❖ Interactivity- between objects such as surgical instruments and organs
- ❖ Sensory feedback – tactile and force feedback

In order to provide more understanding about the components of surgical training simulator, some of the components will be discussed briefly, more details as below.

2.4.1.1 Display of organ properties

Accuracy modeling of human organs and tissue requires deformation to be considered. The model must be able to respond rapidly to interactive manipulation. It should closely approximate the behavior of tissues as they are being cut or stretched and the deformation should appear realistic when rendered.

Kinematics models and physical models are two types of deformable model. Kinematics models include free form deformation and do not consider the effect of physical properties such as forces during deformation. This type of model seldom used in surgical simulation compared to physical based models.

Meanwhile, physical-based models can incorporate material properties. Two types of common method used in this model are mass spring and finite element. Mass spring model formulations are discrete and represent regions of interest as point mass topologically connected by spring dampers. Deformation the model changes the level of potential energy in the model. In contrast, finite element model computes deformation over the entire volume. This permits tissue properties to be more accurately modeled.

2.4.1.2 Interactivity

Collision detection is the main issues that need to be concerned in providing the interaction between surgical tools, tissues and organs. Some common techniques used are collision bounding and collision refinement. First method bounds the regions of intersection such as oriented bounding box and static partitioning. Meanwhile, second method will determine whether the intersection actually occurred and the exact collision loci. Example of this method is ingenious hardware-based collision detection.

2.4.1.3 Visual and Haptic Display

Combination of visual and haptic display in surgical simulator able to provide efficient interactive between surgeon and simulator especially in performing tasks which requiring hand eye coordination.

Haptic always refer to the manual interactions with virtual environment that allow touching, feeling and manipulating the object in the environment. By using special input and output such as joysticks and glove, user can always receive the feedback

from the computer applications in the form of felt sensations either in the hand or others part of the body.



Figure 2.9: Example of haptic devices. (a) Cybergrasp Force Feedback Glove (Virtual Technologies, Inc.). (b) Laparoscopic Impulse Engine (Immersion Corporation)

To obtain an enhanced view of the real environment, external devices such as head mounted displays, stereoscopic monitors, environment displays and retinal display are used.

Tracking devices is also an essential device that allows a virtual reality system to determine the position and orientation of the user's selected body parts. The computer can update the displayed image to reflect the current body parts pose. Now, many interaction devices such as glove incorporate a tracking device to measure the position and orientation of the hand. Based on this information, a hand can be rendered at the same position with respect to the user and provide feedback for necessary manipulation.

Stereoscopic display can be achieved by generating a stereo pair image. Each human eye is presented with a slightly different image of the scene. The images are

similar to that perceived by each eye when viewing an actual 3D scene. When viewed together, the brain fuses the image pair, producing an illusion of depth. (Liu et al. 2003)

The main functions of the visual devices will be summarized at the table below.

Visual Device	Functions
Head mounted display a) Optical see through display b) Video see through display	User can see the real world through half transparent mirrors placed in front of the user's eye. The real world is captured with two video cameras mounted on the headgear, and the computer-generated images are overlaid with this view.
Tracking device	Determine the position and orientation of the user's head, and then computer can update the displayed image to reflect the current head pose.
Stereoscopic displays	Designed to give the user a perception of depth.

Table 2.1: Types of visual devices

2.4.2 Advantages

Some potential advantages of developing virtual simulator for such a virtual surgery are:

- ❖ Experimental and active learning (Mantovani *et al.* 2003)

Students are encouraged to participate actively since learning in virtual environment requires interaction. They can assimilate knowledge when they have the freedom to move and engage in self-directed activities within the

learning context. They have to invest mental effort for the construction of conceptual models that are both consistent with what they already understand and with the new content presented in finding and structuring content autonomously.

❖ Evaluation and assessment

Since the performance of the trainees can be easily monitored and recorded through the session that conducted by the trainers using the virtual surgical simulation, it become a great evaluation and assessment tools to certify medical personnel base on their practical skills in executing a surgical training.

❖ Cost Savings

It will save the precious resources such as the use of animals in the training of medical personnel. Besides, anatomical models of the human body will be shared for use in various surgical simulation applications when there is availability of sharing the high performance network between medical centers and universities.

❖ Adaptability

Virtual reality learning offers the possibility to be tailored to student's characteristics and needs due to different student may have different learning rates and styles. Students are allowed to proceed through an experience at their own pace, and during a broad time period not fixed by a regular class schedule. (Mandovani *et al.* 2003)

Furthermore, student could develop new techniques, to practice on the unfamiliar and difficult parts of the surgery technique for unlimited time and to predict results of particular surgical procedures without harming to any patient. Student can try different techniques and observe the anatomy in different perspective.

❖ Learning in context impossible or difficult to experience in real life

Virtual reality allows observation and examination of areas and events unavailable or impossible such as traveling inside the human body.

Virtual reality also able to provide the effective training in situation requiring the use of the expensive or hard to obtain equipments.

2.4.3 Disadvantages

Although surgical simulator able to provide the significant benefit to the student, there also have some disadvantages as mention below:

❖ Side effect so called simulator sickness (headache, vomiting, nausea, dizziness) and confusion can occur when using the virtual reality system for a period of time. Researchers found that 89 out of 146 otherwise healthy adults suffered temporary simulator sickness after using helmet-mounted display for just 20 minutes. (Langreth 1994)

❖ The sense of the smell in virtual environment simulation has been largely ignored. However, smells are extremely important because it can help to distinguish specific substances and give a sense of reality to a situation. (Mantovani *et al.* 2003)

2.4.4 Current researches on virtual surgical simulator

Currently, there are still a lot of ongoing researches in virtual reality that bring efforts to medicine environment. Some of the examples are listed as below:

- ❖ Massachusetts Institute of Technology (MIT) Touch Lab in Cambridge mainly concerns on developing machine haptics, and enhance human-machine interactions in virtual reality and teleoperator systems. Their laparoscopic surgery simulator implementing fast algorithms for graphical rendering of the appearing and haptic rendering of reaction forces arising from tool-tissue interactions during surgical simulations. (MIT 2002)
- ❖ Mechatronics Working Group in cooperation with the Department of Obstetrics and Gynecology of the University Hospital Zurich, the Anatomical Institute of the University of Zurich, and the Institute of Mechatronic Systems of the University of Applied Science in Winterthur, started the LaSSo-project which aims in building a Virtual reality (VR) based surgical simulator.

This project has implemented the algorithms both for the preparatory steps of partitioning the mesh and for the actual parallel computation, as well as a complete system model, simulating not-yet-existing devices. (Image Science Group)

- ❖ Virtual Bone-Setter is an international research project with Singapore General Hospital which mainly focuses on virtual orthopedic surgery. This program able to run on the personal computer and accept mouse as main

input device. All the images will be processed by the freeware to reconstruct 3D models from their 2D slices.

This program able to guide the surgeon to learn how to fix fractured bones, and to perform pre-operation planning without damaging costly plastic bones. (Sourin, 1997)

- ❖ CSIRO scientists and researchers able to demonstrate real time virtual surgery training by planning and rehearsing surgical procedures within a shared virtual environment between two separate locations. According to Chris Gunn, a CSIRO virtual reality researcher, their software manage to travel at long distance and able to handle the problem of network latency and jitter. This can be shown by the success of conducting the demonstration of simulated cholecystectomy between surgeon in Parliament House in Canberra and the trainee in the head office of Reachin Technologies AB in Stockholm. (CSIRO 2002)

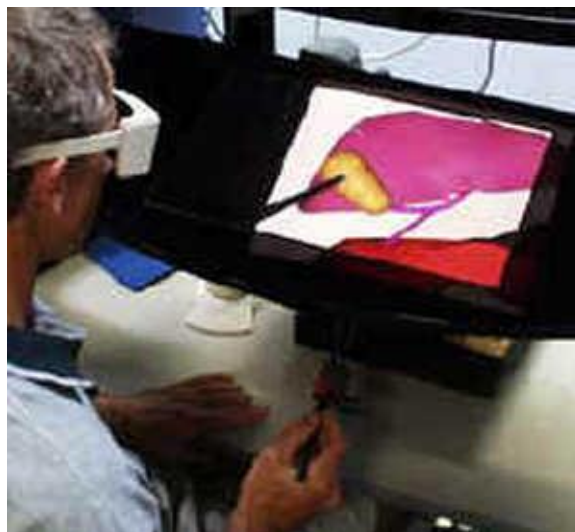


Figure 2.10: A CSIRO scientist performing virtual gallbladder surgery using a haptic workbench

- ❖ Virtual Reality and Active Interface Group at the Swiss Federal Institute of Technology have been working on the development of a laparoscopic

surgery training system called Virgy. This project consists of two complementary design phases. The first one relates to the creation of a virtual environment by means of the Libptk, a 3D graphic sensor engine which generates the population of all the objects in the simulation. A so-called surgical ghost provides the physician with force feedback through the endoscopic tools that are used for manipulation of the virtual organs. The second phase involves a multimedia communication platform, allowing surgeons at remote locations to co-operate and offering students an opportunity for distant learning. (Virtual Medical Worlds 1998)

2.5 Surgical Tasks Methods used in Current Research

There are a lot of methods had been implemented in the virtual application to perform collision management, cutting and grabbing (deformation). Some of the useful methods that implemented by other researchers will be discussed to provide a better guidance in developing this project.

2.5.1 Vertex Manipulation (Salleh, 2001)

Among all of the methods, this is the simplest methods to perform the surgical tasks. Improvements of these methods become the main topic to be discussed in this project. In this research, basic methods had been implemented to enable the user to experience stretching, touching and cutting with force feedback during the virtual laparoscopy surgery.

In grabbing/stretching, the nearest intersected vertex to the tip of the virtual scissor become the centre of the stretching point. When the user stretches the skin, the

centre vertex will follow the position of the tip. The surrounding vertices will also be set to move towards the stretching point but with half of the distance as compared the centre vertex such as figure below.

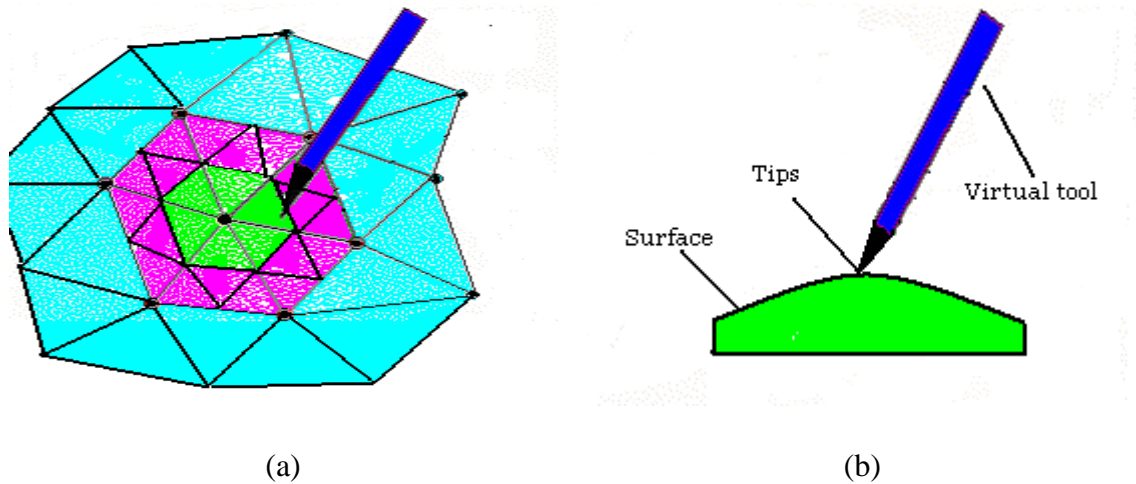


Figure 2.11: (a) Distance of the first level (green) and second level (dark pink) of vertexes reduced half from the center vertex during grabbing (b) Side view during grabbing

Two functions are written to perform the task. First function was written to search the centre vertex. The direction and magnitude of movement can be found by comparing the initial position of this centre vertex and its current position. Second function will search for polygons that have the centre vertex as one of their vertex. All the vertices initial and current positions were stored for manipulation and for later use when all the vertices need to be restored to their original position.

In cutting, the simulation had been performed on a virtual vein. The closest vertex of the polygon to the point of contact and all the vertices surrounding the point of contact are considered the peak point and other surrounding vertex were made to deform. In cutting procedure, polygons with those vertices that are in the peak deformation were removed from the simulation effectively.

2.5.2 Tissue cutting Using Auxiliary Surfaces (Basdogan, 1999)

Cagatay Basdogan and his friends modeled the cutting instrument as a line segment for fast detection of collision. The tip and tail coordinates of the simulated instrument are then used to detect if the instrument collided with the 3D object (edge polygon collision). Then, the collision between the line segment model of the instrument and the edge of the 3D object (edge-edge collisions) are detected to determine the collision points as the instrument passes through the surface. Then the vertices of polyhedron that are close to the collision points along the line of cut are determined and duplicated. Finally, a simple polynomial model is used to separate the duplicated vertices from each other to visually expose the cut.

To simulate the physics involved in cutting and to compute the new position of the separated vertices, one can use an auxiliary grid, made of a 2D network of point masses connected to each other with initially stretched springs and dampers. Updating the visual database involved when exposing the cutting procedure.

2.5.3 Virtual Cutting of Anatomical Structures [Voss, 1999]

In deformation, a spring mass system is used to simulate the dynamic object behavior. These systems are widely used in physical simulation systems because their calculation complexity is small enough to allow real time simulations.

To approximate the tool form in a better way and no original particle is moved after a collision was detected. Instead new particles are created on the fly using some

templates. The templates are used is based on the number of instrument vertices inside the object.

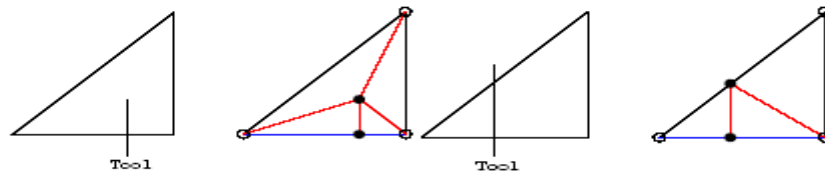


Figure 2.12. : Deformation Templates

After collision detection determined, the active faces that direct affected particles are displaced by calculated amount and marked fixed. Then a simulation step is executed to adjust the remaining particles and before returning to the main loop the marked ones are released. If in one of the following simulation steps, a collision between the new faces and the tool occurs, no new elements are generated, instead the current ones are displaced as if the collision occurred with the old face.

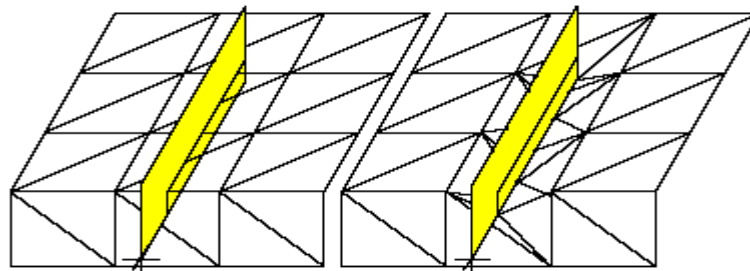


Figure 2.13: Deformation Point Movement

Meanwhile, cutting algorithms is based on the usage of different templates to create the structures generated by the cutting operation of a knife. In a first step, the algorithm decides if the cutting threshold (border between deformation and cutting) is exceeded. After finding the appropriate template the old object surface is changed in two steps. First the old active surface faces are splitted to simulate the incision, after that the structures inside the old objects are created to ensure a closed surface.

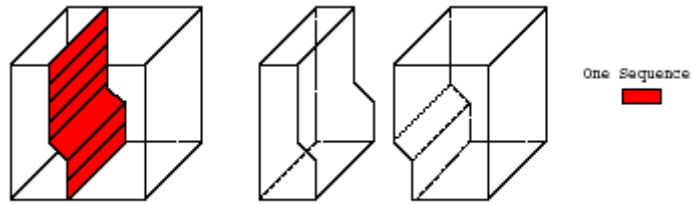


Figure 2.14 : Example of Cutting

2.6 Collision Management Methods

The simplest algorithms for collision detection are using bounding volumes and spatial decomposition techniques in a hierarchical manner. Typical examples of bounding volumes include axis-aligned boxes and spheres. Currently, a lot of hierarchical structures that used for collision detection include cone trees, k-d trees and octrees [Samet, 1989], sphere trees [Hubbard, 1993] and others.

For example, in order to compute collision between the simulated instrument and 3D objects in an efficient manner, Basdogan and friends [Basdogan, 1999] create and utilize two types of hierarchical database. The first one is a hierarchical bounding box tree for collision detection. This hierarchical tree reduces the number of collision checks and enables to find the collision point in a fast manner. During the pre-processing stage, this bounding box tree is created for each polyhedral object. At the highest level of the tree is a bounding box that covers the whole polyhedron. At lower levels, the bounding box of each parent has two children. Each child of the parent has a bounding box that covers approximately half the number of polygons of its parent. The hierarchical tree branches down in this manner until each child covers only one polygon.

During the simulations, first the collision is checked between the surgical tool and the highest level bounding box. If there is a collision, the collision between the tool and the bounding box at branches are checked. The process is repeated till the lowest level is reached. Finally, the collision between the polygon and the line segment model of the surgical instrument is checked and the collision point is computed.

Limitation of these algorithms is that for deformable objects, one need to update the hierarchy trees at every step of the simulation. This is a time consuming step that significantly reduces the efficiency of these algorithm.

There are other spatial representations that are based on BSP's [Naylor, 1990] and its extensions to multi-space partitions [Bouma and Vanecek, 1991], spatial representations based on space-time bounds and many more. All of these representations able to perform very well in tests, whenever two objects are far apart. However, when the two objects are in close proximity and can have multiple contacts, these algorithms either use subdivision techniques or check very large number of bounding volume pairs for potential contacts. In such cases, their performance will slows down and become a major bottleneck in the simulation [Hahn, 1998].

Many theoretically efficient algorithms have been proposed for polyhedral objects in computational geometry. However, most of them are restricted to static environments, convex objects, or only polyhedral objects undergoing rigid motion [Chazelle and Dobkin, 1987]. Their practical utility is not clear as many of them have not been implemented in practice.

OBBs have been extensively used to speed up ray-tracing and other interference computations [Arvo and Kirk, 1989]. However, it is practical for only small models. As for ray-tracing, algorithms using structure editors and modeling hierarchies have been used to construct hierarchies of OBBs. However, they cannot be directly applied to compute tight-fitting OBBs for large unstructured models.

Since OBBs are convex polytopes, algorithms based on linear programming [Preparata and Shamos, 1985] and closest features computation can be used as well. Overall, efficient algorithms were not known for computing hierarchies of tight-fitting OBBs for large unstructured models, nor were efficient algorithms known for rapidly checking the overlap status of two such OBBTrees.

2.7 Technologies in Virtual Surgical Simulator

2.7.1 Laparoscopic Cholecystectomy Surgical Simulator

(Webster and Haluck 2003) (Mahoney 1999)

Roger Webster, a computer scientist at Millersville University of Pennsylvania, and Randy Haluck, director of minimally invasive surgery at Penn State's Hershey Medical Center, have joined to develop a laparoscopic cholecystectomy surgical training software system to practice the majority of LC operation.

The haptic hardware is developed using the Immersion Laparoscopic Surgical Workstation and the hardware for endoscopic camera is the Verifi Technologies™ USB thirty-degree endoscopic Vcamera™ unit. Entire software is running on a

WindowXP workstation with a dual 2.2 GHz Pentium processor and an NVIDIA GeForce OpenGL graphic accelerator.



Figure 2.15: Laparoscopic Cholecystectomy Surgical Simulator

This surgical simulator was designed to train and test for many laparoscopic skills such as grasping, cutting the cystic duct and others. Besides, they also use texture motion algorithm to simulate patient breathing. Meanwhile, the researchers developed a basis mass-spring model to simulate the impact on the skin and underlying tissue of pushing, pulling and cutting. In this case, the skin deform relative to weight, or pressure that applied to the springs through the surgical tools. Then the software manages to calculate the contact forces in the soft tissue surrounding the wound and applies the appropriate forces to the user through the Phantom.



Figure 2.16: Laparoscopic cholecystectomy simulator showing the user exposing the cystic duct (Webster and Haluck 2003)

The main goal of this simulator is to provide an effective method to learn key elements of laparoscopic cholecystectomy especially on the surgical skills. For future, the researchers would like to include visco-elastic modeling of the deformation and fidelity force computation and system validation.

2.7.2 The LapSim System (Surgical Science 2003)

Surgical Science has provides a professional worldwide surgical training and practice product which known as The LapSin System. This company is based in Gothenburg, Sweden, maintaining close ties to the Gothenburg University. By ongoing research and through close cooperation with the medical community they manage to provide professionals worldwide with the means to improve surgical training and practice.



Figure 2.17: The LapSim System

The LapSim System is the first series of laparoscopic simulation and training system that can provide effective learning experience. The LapSim System consists of LapSim Basic Skills 2.0 with add on LapSim Dissection and LapSim Gyn.

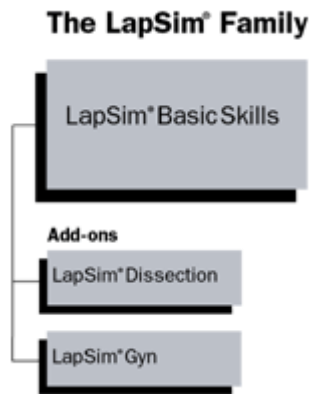

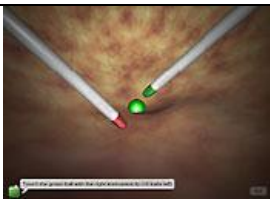



Figure2.18: LapSim System Family

LapSim Basic Skill 2.0 utilized the advanced 3D technology to provide the student with realistic virtual environment. The trainers can easily modify and create the courses to fit a student’s specific need. Somehow, practice session can vary in graphic as well as in the level of the complexity. Some basic skills that can be studied in this system are as below:

Basic skill	Simulation	Description
Camera Navigation		Basic navigational skills such as handling camera are learned, and an initial feel for how laparoscopic instrument are handled is developed
Instrument Navigation		This module allows the student to move two instruments in three-dimensional image in a precise manner.
Cutting		Cutting is the most important skill in laparoscopic surgery. This module use a foot pedal to trigger the cutting instrument

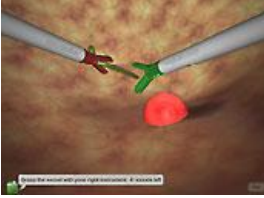

Grasping		Student can learn to move on the actual manipulation of objects such as grasping, stretching and move the blood vessel.
Clipping		Allow student to use multiple instruments such as grasper, scissor to apply clipping in a save environment.

Table 2.2: Some basic skills in LapSim2.0

LapSim Dissection use to simulate a critical phase in the frequently performed laparoscopic cholecystectomy procedure such as the dissection and subsequent clipping and cutting of the gall bladder's bile ducts and blood vessels.

Meanwhile LapSim Gyn provides a realistic environment for the training of three gynaecological laparoscopy procedures: sterilization, ectopic pregnancy removal and the final suturing stage of the myomectomy procedure.

2.7.3 Virtual Arthroscopic Knee Surgery Simulator (Hollands and Trowbridge 1996a) (Hollands and Trowbridge 1996b)

Robin Hollands and Tony Trowbridge, researchers in University of Sheffield, UK had developing a PC based virtual reality arthroscopic training system. The system is being developed on a Pentium 133 MHz PC with Windows NT platform and low cost Matrox Millenium accelerator card as the graphic cards. The VR toolkit used is the Visual C++ based WorldToolKit from Sense 8.

This simulator is able to represent many pathologies and patient types, and permitting the practice of diverse surgical procedures. Some of the functions are listed below:

- Allow the trainee surgeon to practice navigating around the knee
- Identifying significant landmarks and rehearsing standard inspection routes
- To practice triangulating an instrument and camera

Although this surgical training did not provide the force feedback, this simulator is able to provide the polygon-based collision detection algorithms. Constant checks can monitor the position of the virtual instruments within the knee and any collision with the internal structures can be indicated by aural and visual cues.

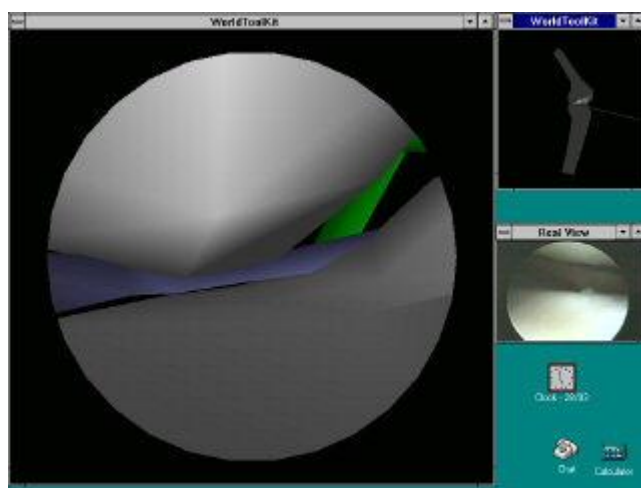


Figure 2.19: Virtual Arthroscopic Knee Surgery Simulator showing the virtual view of the knee (left), the whole knee (right top) and the captured image from inside a real knee (right bottom).

This simulator has been identified by training centers as a lower risk method of training due to the trainee only using the low cost plastic replicas of the arthroscope and instrument. Currently, this simulator has been used by a number of surgeons from Royal Hallamshire Hospital and Northern General Hospital in Sheffield. The surgeons are impressed by the sense of realism afforded by the necessity for correct

hand-eye co-ordination when manipulating the dummy instruments together with the leg model within the virtual environment.

2.7.4 Karlsruhe Endoscopic Surgery Trainer (Kuhn 1997) (Versweyveld 1998a)

Karlsruhe Endoscopic Surgery Trainer is one of the well known virtual reality endosurgery simulator by Forschungszentrum Karlsruhe (FZK), which in Technik and Umwelt, Germany. The research team is working with a high performance graphic workstation, Silicon Graphics' Onyx, including an RE2 graphics subsystem and two processors, to provide the trainee with a virtual operation area and a real time "synthetic" endoscopic view. This workstation as a core unit and running on multipurpose KISMET (*Kinematic Simulation, Monitoring and Off-line Programming Environment for Telerobotics*) software which under development by FZK since 1986.

Specifically in laparoscopic procedures, a "phantom box" which is a rough imitation of the outward human abdomen is introduced into the simulation environment to serve as a surgeon-computer interface. This phantom box able to offer trainee an artificial cavity, incorporating electromechanical instrument guidance and tracking systems for cameras. In addition, several typical surgical tasks, such as grasping, cutting, coagulating, and setting of clips were implemented.

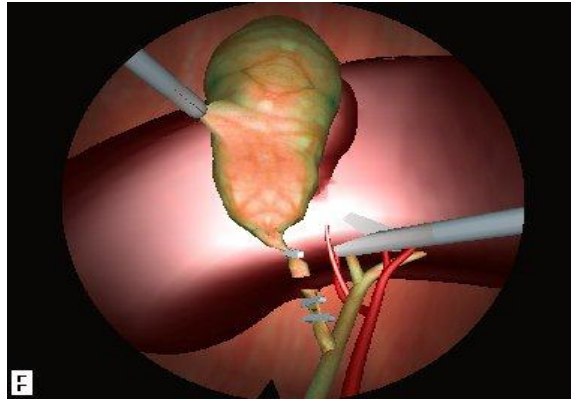


Figure 2.20: Virtual endoscopic view for Laparoscopic Cholecystectomy
simulation by FZK

The KISMET package is also able to provide haptic feedback interfaces, in order to calculate in real time the interaction between deformable objects and surgical instruments. This is very important as to let the trainee to obtain a realistic sensation of the interaction of the soft tissue and its physical behavior with the surgical instruments.

2.8 Development tools

In this section, the main development tool that was used in this project is discussed for better understanding.

2.8.1 Sense8

Gelband and Gullichsen founded Sense8 on January 1990. Gelband derived the name from the adjective 'sensate' which perceiving through the sense. (Primentel and Teixier 1995)

At the same year, Gelband and Gullichsen demonstrated one of the few VR systems in the world to Intel. Intel also was very interested in finding a partner to collaborate on the development of a VR system based on Intel's Digital Video Interactive technology. After the presentation, Intel approved funding to pursue research into PC-based VR system to Sense8 due to ability to deliver the prototype within six-month deadline. Furthermore, the VR system that demonstrated previously also able to support various hardware platforms.

Finally, in late 1990, Sense8 and Intel unveiled the result of their collaboration at the San Francisco Meckler VR conference. For two days, many attendees waiting for hours just to get opportunity to experience the virtual trip which developed by Sense8. The demonstration successfully represented a breakthrough in PC realism and rendering performance for virtual worlds.

2.8.1.1 World Tool Kit (WTK) (SENSE 8 2001)

Based on the success in San Francisco Meckler conference, Sense8 plans to market the product based on the DVI chip set's capabilities. Hence, Gelbans and Gullichsen had created a set of powerful and elegant programming functions calls and finally WorldToolKit become commercially available in June 1991. Sense8 also adapted this product to run on Silicon Graphics, Sun and DEC/Kubota workstation. In March of 1994, Sense8 managed to bring the VR product to the Windows operating system by releasing the WorldToolKit for Windows.

Now, WorldToolKit releases version 10 is a portable, cross-platform software development system for building high-performance, real-time, integrated 3D applications for scientific and commercial use.

WorldToolKit R10 is an object-oriented library written in C with more than 1100 high-level functions calls that enable the users to develop the most complex application while improving organization productivity. The functions included for device instancing, collision detection, controlling rendering and others. With the high-level application programmer's interface (API), user applications can be quickly prototyped, developed, and reconfigured as required.

WorldToolKit R10 also supports network-based distributed simulations, CAVE-like immersive display options, and the industry's largest array of interface devices, such as headmounted displays, trackers, and navigation controllers.

The architecture of WorldToolKit R10 has been designed to incorporate the power of scene hierarchies. This efficient visual database representation provides increased performance, control, and flexibility through features such as hierarchical object culling and efficient use of transform information and level-of-detail switching.

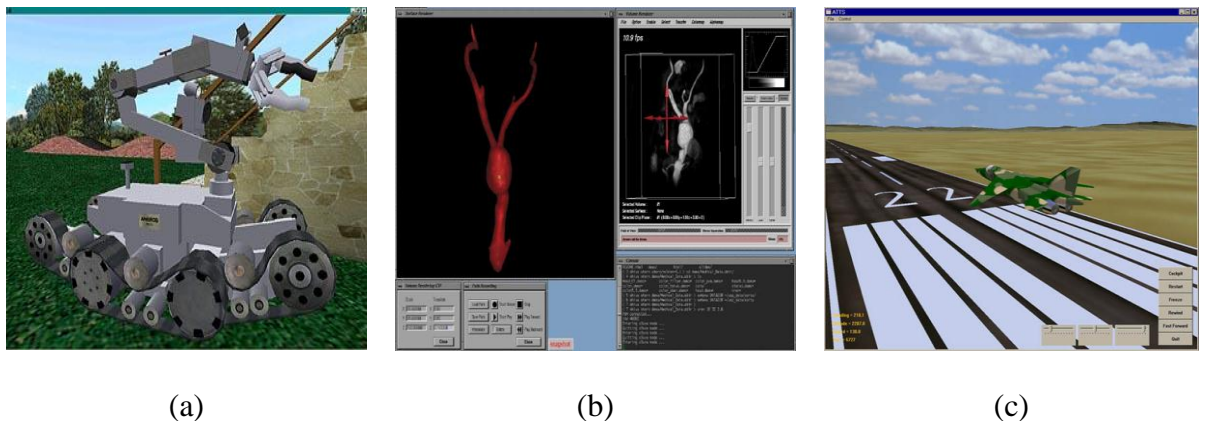


Figure 2.21: Example screenshots on applications develop using WTK. (a) Battelle Memorial's Bomb Disposal Simulation (b) ViewTech's Medview was built using WorldToolKit Technology (c) Smith Industries Flight Recorder Simulation

Incorporates the philosophy of OpenVR in WorldToolKit has enabled the software to be portable across platforms, including SGI, Sun, Windows, and Linux. WorldToolKit R10 is optimized to make full use of the unique capabilities of each platform to deliver the fastest graphics possible. In addition, WorldToolKit R10 supports a wide variety of input and output devices, and also allows user to incorporate existing C code-such as device drivers, file readers, and drawing routines into their application.

2.9 Conclusion

First of all, this chapter has briefly explained the techniques and types of minimally invasive surgery that had been practiced. One types of minimally invasive surgery, laparoscopic cholecystectomy had been discussed in detail.

Gallbladder removal by laparoscopic surgery is an exciting development because it offers so much to the patient. The surgeon carefully evaluates each case and discusses it with the patient. While problems can occur with the procedure, they are unusual. In most instances, patients experience excellent results and resume their normal activities very quickly.

Traditional methods of laparoscopic training have many disadvantages in term of risk to patient, cost, ethical problem and lack of realism. Clearly, these problems have produced the opportunity to develop the virtual surgical simulator. Hence, the concept of virtual reality in medicine was discussed including the area of components, advantages and disadvantages. The identification of the benefit of simulation in surgical training had led to the development of several researches on virtual reality that had been conducted in universities and others research centers.

Consequently, the reviews are also looking into the area of some famous virtual reality surgical simulator such as Laparoscopic Cholecystectomy Surgical Simulator, The LapSim System and Virtual Arthroscopic Knee Surgery Simulator.

Finally, the developments tools that are involved in this research were discussed.

Chapter 3 Methodology

3.1 Introduction

Basically, methodology involves a series of methods or processes that can be carried out to produce a systematic and orderliness application by achieving the defined objectives previously.

In this research, mainly are considering the surgical task that can perform in laparoscopic cholecystectomy. Thus, it is crucial to understand the steps and methods that are involved in this procedure such as collision detection, cutting and grabbing in virtual environment. Since this study is using WorldToolKit (WTK) library to build the prototype, hence most of the methods will be implemented by using the functions in WTK library.

First, defining the hardware and software configuration is important to provide information to the user on loading the image into the virtual environment and handling the virtual environment in term of movement on viewpoint and orientation. By doing this, the user is able to view the image in different perspectives that cannot be done in real laparoscopic cholecystectomy.

Next, types and architecture of laparoscopic surgical tools will be discussed. This is important to determine the method to manipulate the surgical tools in virtual environment.

As mentioned in the objective previously, this research does not consider the haptic feedback. Hence, collision management is important to provide a proper guideline

for user to know the interaction between the surgical tools, tissues and organs. Collision detection used to determine the moment when the user touches the surface of the organ by using the surgical tools. The response of this touching can be seen clearly through the changing of the color on the surgical tools, audio feedback and the warning provided by the system before user can proceed to the next task.

The surgeon can starts perform the cutting and grabbing on the gallbladder after clearly defining the collision detection and response. Surely, the methods of performing those tasks will be discussed at the coming section.

3.2 Hardware Configuration

In order to achieve one of the objectives defined previously, the amount of hardware interfaces that were used in this study had been reduced to a minimum. This is due to, most of the features of the simulator are implemented by using software. On the current system, the host computer that had been used is a standard entry-level of Pentium product with 261Mb RAM and AC'97 sound card.

Interaction between the user and the system is primarily through the use of the keyboard and mouse. During the simulation, the user can use the keyboard to control the virtual surgical instruments with which the user interacts with the anatomy being modeled. In real operation, laparoscopic surgical tools are inserted through different trocars which are located in either side of the laparoscope. To simulate these two different surgical tools, the surgeon can alternate between the different keys on the keyboard in order to control the instruments.

Besides providing the alternative to choose the instruments, the keyboard is also used to perform commands such as performing surgical tasks, viewing the wire frame of the model, and checking for frame speed and others. All the tasks that can be performed by the keyboard are provided at the menu on the screen.

Meanwhile, the mouse is mainly used to control the viewpoint and orientation of the scenes in virtual environment.

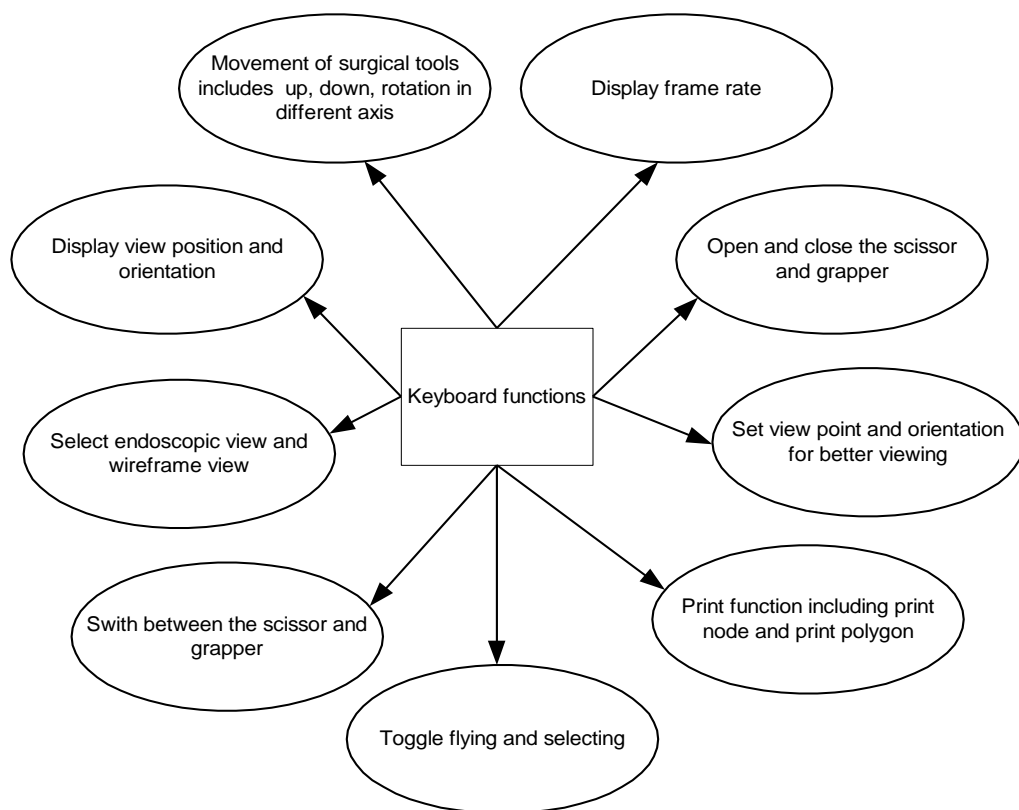


Figure 3.1: Types of functions performed by keyboard

3.3 Software Configuration

In order to develop the simulation, the Virtual Reality toolkit used is the Microsoft Visual C++ based WorldToolK(WTK) from Sense 8, running under Window NT.

As mentioned in literature review, WTK is a powerful commercial software that is able to provide cross platform development environment for high performance, real time 3D graphic application. Objects can have real world properties and behaviors.

The toolkit provides more than 1000 functions written in C for most of the high level tasks of handling the peripheral devices, manipulating objects, detecting collision and others in addition in allowing access to lower level facilities of OpenGL.

3.3.1 WTK description

Simulation manager is the heart of all WTK applications. It will automatically maintain the objects that are created in the universe. Universe is the ‘container’ for all objects in the simulation. It includes geometries, sensors, lights, viewpoints and other objects types. Only one universe for every simulation. The simulation loop entered by calling *WTuniverse_go* and the loop is broken with a call to *WTuniverse_stop*.

The data flow diagram for the software operation which also becomes the flow of this prototype is shown in figure below.

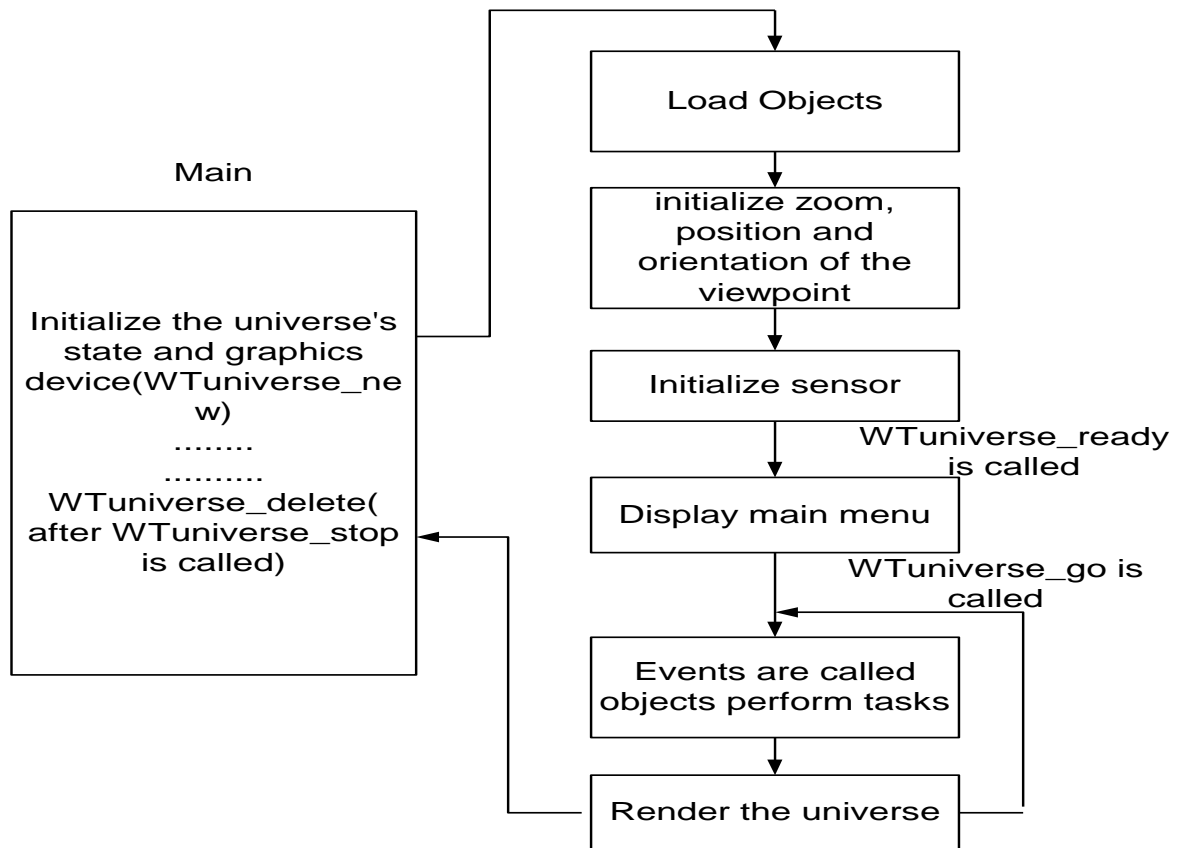


Figure 3.2: Data flow diagram for the operation using WTK software

In a WTK application, an user can use the function *WTuniverse_new* to create the universe. This function is the first WTK call in the program and only can be called once in the application. This function is used to initialize the universe's state and the graphics devices that are used to view the simulation. These include the selection of the suitable types of display whether with no window or have window such as single window (*WTdisplay_default*), two windows (*WTdisplay_stereo*), and CrystalEyes glasses window (*WTdisplay_crystaleyes*). This function can also set the characteristics of the window or windows created such as window with border (*WTwindow_default*), no border (*WTwindow_noborder*) and stereo window (*WTwindow_stereo*).

Next, we can set the background of the window and lighting information. There are four types of lighting in WTK (ambient light, directed light, point light and spot light). In this study, directed light is chosen due to this lighting can provide the effect of sunlight that has direction but no position.

After defined the window, background and lighting, objects are ready to load into the universe. These include objects created using different software package such as 3D studio MAX, which have to import into the simulation, and simple objects created using WTK functions. In our study, human torso and the virtual tools will be loaded into the virtual environment.

After that, the position and orientation of the viewpoint in this system should be initialized. This defines the position and orientation from which all of the objects associated with a simulation are rendered and projected to the computer screen. In this way, the objects can be viewed from certain angle to provide the capability to allow the simulation of procedures to control the view of the relevant part of the human model.

In this simulation, all the defined actions that can be performed by the user through keyboard key will be listed in a menu. This menu will be displayed on the screen after initialize the *WTuniverse_ready*. This function prepares the application for entry in to the main simulation loop.

Next, the activities of the simulation can be controlled and defined by the function *Wtuniverse_setaction*. Several tasks have been defined in this prototype such as reading from a mouse, checking keyboard inputs from the user and detection of collision of the virtual objects and tools.

Finally, the universe is rendered which will update the display of the objects in the display window. Calling the function *WTuniverse_stop* will finally stop the simulation loop. However, *WTuniverse_delete* also needs to free all of the objects in the universe including cleaning up and closing the WTK display. This function always is the last call in the program.

3.4 Methods

In this section, the different methods of implementation will be discussed.

3.4.1 Graphical Objects

In WTK, every 3D object that loads to the virtual environment is known as geometry. Examples of geometries are block, cylinder, balls, and houses. Geometry is the composition of 3D points (vertices) and the surfaces (polygons). Along with material properties such as color and texture, users have great freedom to create any shape and objects.

However, users not only can create geometries as their object, they also can import objects from others sources. Since the functions of the WTK to create objects is very basic (only allow to create block, cylinder, cone, sphere, hemisphere, rectangle, truncated cone and text3D), the graphical object that used in this project is created by using other modeling software package known as 3D Studio Max and then imported into the WTK virtual environment.

Mainly, the objects that were needed in this project are gallbladder that included in human anatomy and surgical tools such as knife, scissors and grasper. All of this

objects need to save as '3ds' file in order to allow WTK to import the file into the virtual environment before performing any simulation.

For the purpose of display, the simulated surface of the model is built from a set of polygons that consists of small triangles that can enhance the realistic graphic rendering of the instrument. Each polygon consists of three interconnected vertex. Although the resulting models are not as detailed as those in volumetric images, they are still accurate enough to perform the gallbladder removal procedure and to identify and to overcome many problems before entering to the real operating room.

3.4.2 View Point and Orientation

When one creates a universe with *WTuniverse_new*, default viewpoint and orientation was automatically created as shown in figure below. However, additional viewpoints such as 'bird eye view', 'rear view' can be constructed and switched between them.

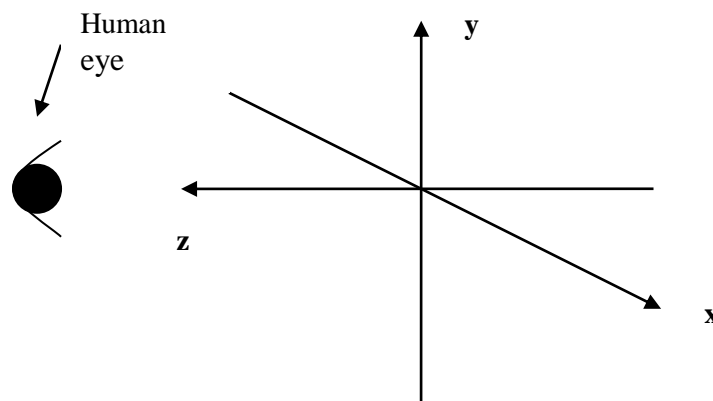


Figure 3.3: Default human eye viewpoint

The position and orientation of a viewpoint always can be set through function calls *WTviewpoint_setposition* and *WTviewpoint_setorientation*. This is useful to set the viewpoint and orientation that is suitable in the application to let the user to have a different perspective of view for the human organs.

Besides, controlling a viewpoint's position and orientation using a sensor that is attached to it such as translation and rotation can be done. For example, a mouse sensor object can be used to construct and attached a viewpoint (*WTmotionlink_new*) in order to perform a movement. Rotation of viewpoint involved in this study that use *WTviewpoint_rotate* to enable the user to have different perception on the anatomy model.

In order to provide a better and clearer view of the objects in the window, zooming (*WTwindow_zoomviewpoint*) in and out on the objects in the window is allowed.

3.4.3 Manipulating laparoscopic tools

Basically, manipulating laparoscopic tools involve two sections. First we need to consider the movement of entire surgical tools in the simulation window. Then, the movement of the part of the surgical tools such as opening and closing of the blade at the grasper before and after performing certain tasks. In this application, the position of the human body and organs and all the types of movement were controlled by the keyboard.

3.4.3.1 Manipulation of entire surgical tools in simulation window

This prototype provides facility to the user to manipulate of entire surgical tools in simulation window. User can perform rotation in x, y and z-axis on the selected surgical tools. Meanwhile, translation also can be done for all direction to allow the user to move the surgical tools towards or backward the target.

Types of movement	Direction	WTK related function
Rotation	x-axis y-axis z-axis	WTnode_axisrotation (WTnode *node, int axis, float angle, int frame);
Translation	Up Down Left Right	WTnode_translate(WTnode *node, WTp3 pos, int frame);

Table 3.1 Summary of the movement of surgical tools

3.4.3.2 Manipulation of part of the surgical tools

Currently, there are two types of surgical tools, scissors and grasper involved in this laparoscopic cholecystectomy. Scissors are used to cut the cystic duct in order to remove the gallbladder from the human body. Meanwhile, grasper is used to grab particular skin of cystic duct before performing next tasks. Besides, a small knife also used to demonstrate different styles of cutting procedure than scissors on the skin. In order to switch among the tools, user allows alternating and moving the surgical tools by inputting the appropriate key.

In order to move the blade to perform cutting and grapping, the architecture of the scissors or grasper that is used in this study is divided into three main parts as shown below:

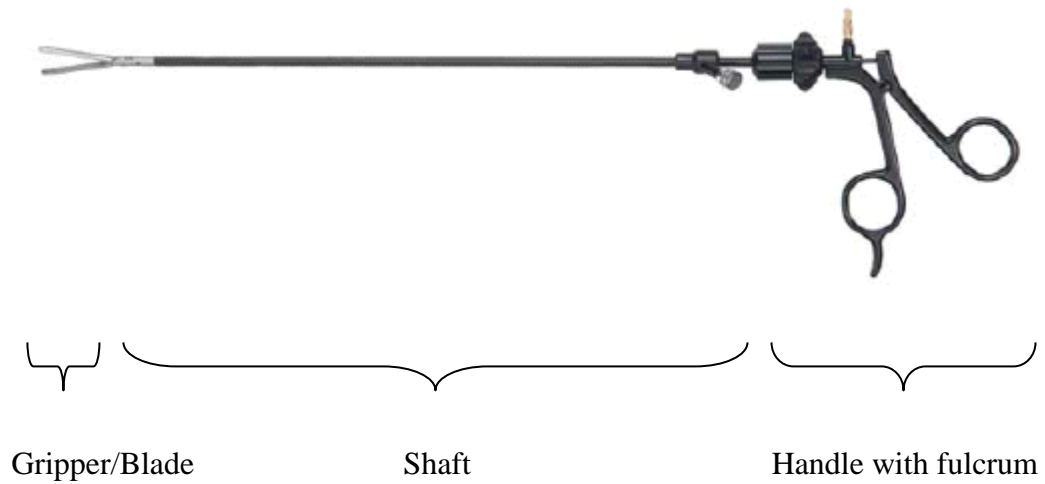


Figure 3.4: Architecture of Laparoscopic grasper/scissor

In this simulation, four small graphical models, which include shaft, and two grippers/blades, will be created and attached together to become a scissors. This technique is used for grasper too.



Figure 3.5: Some example of laparoscopic tools. (a) Prep Scissor (b) Curved Grasper

In WTK, a group node needs to be created by calling *WTgroupnode_new* while loading the surgical tools. This is because the surgical tools were formed by more than one small geometry nodes. Hence, it always need to attach them together to the scene graph to form a group node by calling the function *WTmovnode_attach* (*WTnode *parent, WTnode *child, int attachmentnum*).

3.4.3.3 Effects on tools manipulation

Complexity of the virtual environment will effect the tool's manipulation. Simple test need to be done to observe the tool motion in the VR environment. Currently, the virtual environment consists of human torso with all the organs include the gallbladder which are the main organs in the human torso.

Comparing frame rate achieved when different objects are loaded to the environment holds testing that need to be done. Besides, the motion of the surgical tools and the interaction between tools and target will be observed on different frame rate.

3.4.4 Collision Management

Collision management mainly consists of collision detection and collision response. Collision detection determines the occurrence of contact between the tools and the organ geometry while collision response computes proper response of models such as deformation fields and interaction forces. Both are highly dependent on modeling of organs and surgical tools. This includes three-dimensional representation of organs geometry, visual texture, and condition of tool tissue interaction as well as mechanical properties of soft tissues. (Jung Kim 2002)

In laparoscopic cholecystectomy surgical procedures, the surgeon manipulates and manoeuvres a laparoscopic tool to perform cutting and grabbing. During the procedure the tool will be inevitably be in contact with the cystic duct in gallbladder before perform the cutting to retrieve the gallbladder. Therefore, simulation of touching is one very important aspect to be looked at and considered in the VR simulation.

3.4.4.1 Collision Detection

In a simplistic form to implement this contact feature, it initially involves the detection of contact between two objects. In order to accomplish this, Bounding Boxes Intersection Tests method was used.

Before explaining the method used, the node path of the cystic duct and virtual tools need to be defined first. According to WTK, a node path is a mathematical entity that allows user to distinguish between multiple occurrences of the same node due to instancing. This allows indicating a specific occurrence of a node in the scene graph. Instancing means that there may be only one objects loaded into memory but we can make as many references to it's as needed. The following function can be called to create the node path for particular object:

```
WTnodepath_new(WTnode *node, WTnode *ancestor, int occurrence_of  
_the_node);
```

Next, a bounding box intersection test can be performed between the specified node path and the numbered occurrence of the specified node and its sub tree by calling the functions *WTnodepath_intersectnode(WTnodepath *nodepath, WTnode *node, int occurrence_of_the_node);* .

This function will return null if both nodes and node path do not intersect. If they do intersect, then this function traverse down the specified sub tree in search of the node whose bounding box of the specified node path to that node is created and a pointer to it is returned.

Besides, a Boolean function, *WTnodepath_intersectpoly(WTnodepath *nodepath1, WTnodepath *nodepath2)*; also can be called in the prototype to tests the intersection of any polygons in two node paths and their sub trees. In this study both node paths are specified on the cystic duct and the blade for the virtual surgical tools. This function will return true if an intersection is detected between the blade and the cystic duct. Otherwise, false will be returned.

3.4.4.2 Collision Response

Some methods have been used to provide a significant response to the user to determine that the collisions have been detected.

3.4.4.2.1 Change of blade color

In order to provide the response on the collision that happens in between organs and surgical tools. The color of the particular blade in the surgical tools will change accordingly. The color of the blade on grasper and scissors are different for first collision and second collision. This is useful to provide the response to the user regarding the status of the collision whether is first collision or second collision or no collision at all before performing next tasks.

3.4.4.2.2 Change of Skin Color

Besides the changes of the blade color while collision are detected, the intersected polygon on the cystic duct also being change. This is important to signify the user on the exact position that had been intersected. The color of the intersected polygon can be restored if the collision is not more detected.

3.4.4.2.3 Message Prompt

The response also supported by message prompt. Message will also prompt on the screen while user performing the touching. This is important and always able to acknowledge the user regarding the current status of the collision.

The information that can be provided to the user through the message prompt including the detection of the surgical tools are contacting the cystic duct, total degree of the blade that contacting the duct for scissor and grasper, possibility of performing the cutting for scissors.

3.4.4.2.4 Audio Response

Besides the visual display, the response will also be supported by audio. The proper sounds are embedded while the collision is happened. Appropriate sound can be played by using the function *WTsound_play(WTsound *sound)*. Sounds are display every time the user touches the skin.

Different kinds of sounds are embedded for different kinds of surgical tools. Each sound triggers an alert to the user when there is more than one collision happened for different types of tools.

3.4.5 Grabbing

After determine the collision with appropriate responses, laparoscopic surgical procedures will start by performing grabbing. In a real laparoscopic surgery, a surgeon will manipulate the tools inside a patient body and perform an intended task. In this procedure, the surgeon has to grab part of the cystic duct before performing the cutting.

In the earliest chapter, a simple grabbing method (Salleh 2001) has being proposed. First, the intersection between the virtual scissor and the virtual objects is detected. Next, the nearest intersected vertex to the tip of the virtual scissors need to be determined as shown in figure below which is in yellow color. This centre vertex will become the center of the stretching point.

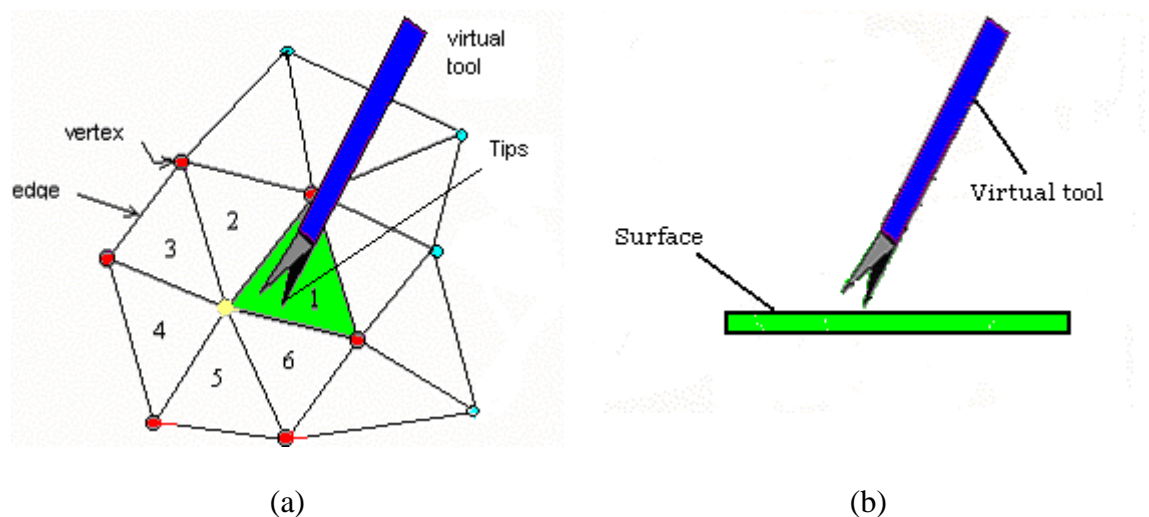


Figure 3.6: (a) A polygon (green color) of the object intersected with virtual tools.

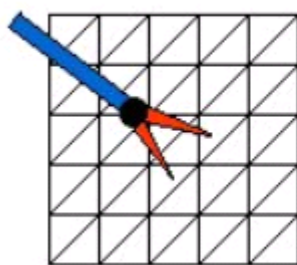
(b) Sides view before grabbing

When stretching the skin, the center vertex will follow the position of the tip of the virtual tool. The surrounding that includes first level and second level vertexes will

also be set to move toward the stretching point but with lesser displacement as compared to the center vertex.

However, in this prototype, a new improved algorithm is proposed based on previous algorithm. The previous algorithm unable to grab the skin if the blades of the virtual scissor remain open. The blade must close and then only can determine the tip's coordinate and perform the deformation using the algorithm mentioned. The vertex will deform according to the tip point. Currently, the tip point is determined by the index that had being fixed at the middle of the gripper. The intersection is not determined through the blade of the gripper.

A new proposed algorithm including collision detection must be able to be detected between the surgical tools and the skin, then it depends on the decision of the user whether he/she wants to perform grabbing. If the defined key is pressed for performing the grabbing, then the collision polygons between the blades and the skin will be captured.



The blade must touch more than one point.

Figure 3.7: Determine the vertexes for the polygon intersected

Based on these polygons, each of the centre vertexes is defined. By using these centre vertexes, the distance between the two tip points for the blade will be determined. This distance can be used to find the real pivot point that had been

mentioned on the algorithm. This pivot point will be the mid of the total distance of the two tip point. If the mid point for the distance is not fall on the exact vertex, then the nearest vertex will become the centre vertex. The neighbor vertex will be deformed according to this vertex.

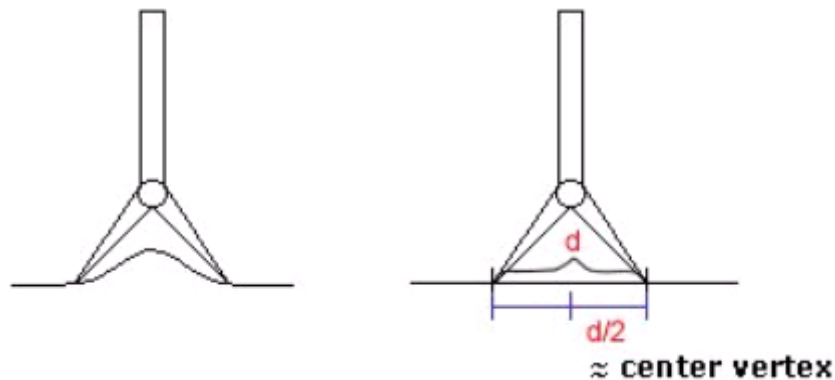


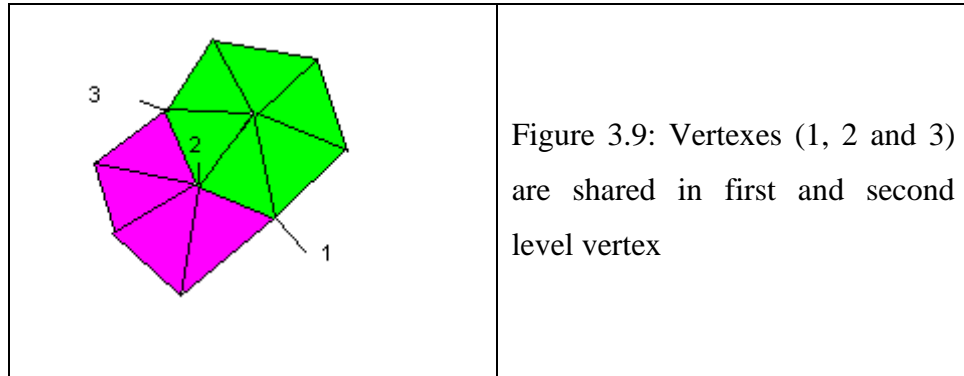
Figure 3.8: Finding the center vertex.

While the user pulling the skin, this pivot point will be deformed to peak point as usual but the part being touch by the blade will also deform but not as much as pivot point. Hence, visually it looks more realistic.

In order to avoid the conflict between touching and grabbing, message prompt will inform the user on the current status of the detection. When user touches on the first time is considered as touching. If user chooses the key for grabbing, then user can pull and perform the grabbing, else if no key is pressed, and user just pulls off from the skin, then there is no grabbing performed.

While performing the grabbing, surrounding vertex will be deformed according to this centre vertex. Hence, surrounding vertexes had to be determined especially the conflict vertexes between first level and second level vertexes. This is due to some vertexes (vertex 1, 2, 3) shared in first level and second level vertex as shown in

figure below. Hence, algorithm needs to be performed to compare the vertexes that involve in first level while searching second level vertex. Redundant vertex ids and positions are ignored.



The method used to perform grabbing is can be simplified as follows:

If (collision detection is found and grabbing's key is pressed and the degree of the blade $< \Pi$),

Find surface polygon intersect with grasper's left blade nodepath

Get polygon id

Find surface polygon intersect with grasper's right blade nodepath

Get second polygon id

Find the distance of the two tip's point

*Distance = distance * 0.5*

Compare the half distance and get new polygon id

Search centre vertex

Search and Save all vertex id

Save vertex position

Search and save first level vertex

Compare first level vertex with centre vertex

If same, ignore else save first level vertex

Search and save second level vertex

Compare second level vertex with first level vertex

If same, ignore else save second level vertex

Set centre vertex id to tip position

*Vector = (vertex position - vertex initial position) * 0.50*

Loop

Add vector to all vertexes(first level and second level) initial position

Save as new vertex position

End loop

- *All the vertices and current positions were stored for returned to their original position.*
- *Limitation- the opening of the blade should not be too big, such as 180 degree. Besides, this grabbing is more suitable for flat skin.*

Stretching simulation will be more realistic if second level vertices and so on are taken into consideration. However, this prototype will only implement the method until second level in order to achieve lower processing.

In WTK, *WTgeometry_beginedit* will be called before geometry of the model such as setting the vertex position and vertex color can be edited. When finish editing, *WTgeometry_endedit* is called to ensure that the WTK properly updates the internal state of the geometry and the entire polygons contained in the geometry.

3.4.6 Cutting

After grabbing the cystic duct, cutting of duct is involved in laparoscopic cholecystectomy. A simple technique of cutting has been simulated on cystic duct before removing the gallbladder.

Before performing the cutting procedure, intersection between scissors node path and model node path have to be determined by using the method that mentioned in collision detection. The intersected polygon will be assigned by the sequence number as polygon id. Function in WTK that can be used to set the polygon id is

*Wtpoly_setid(Wtpoly *poly, short id)*. This first and the last sequence number can always be retrieved out by using *Wtpoly_getid(Wtpoly *poly)* to determine the boundary before cutting the cystic duct.

Cutting of cystic duct can be performed by calling the function *Wtpoly_delete(Wtpoly *poly)*. This function will delete the specified polygon's vertices from the geometry.

A threshold was set to determine the border of deformation and cutting. Cutting procedure will only be performed if it exits the threshold limit. This threshold is determined through the angle of the blades for the scissors. Currently, in this prototype, if the angle for the blade is less than 90 degree ($\pi/2$), cutting procedure will be performed.

Besides using the cutting procedure (Salleh, 2001) mentioned before: The closest vertex of the polygon to the point of contact and all the vertices surrounding the point of contact are considered the peak points. Using the Gaussian formulation, these peak points and all other surrounding vertex were made to deform. In the cutting procedure, polygons having peak deformation were removed from the simulation effectively.

In this study, the procedure had been improved by not immediately deleting the intersected polygon. While collisions are detected, the intersected polygons which follow the blade's node path will be separated into three small triangles instead of being deleted. The separation of polygon can be done by searching for the intersected polygon. Then, the vertexes for each polygon will be determined. Based

on each of the polygon, the middle point is determined too. This new vertex will be used to generate three new polygons by sharing the same middle vertex.

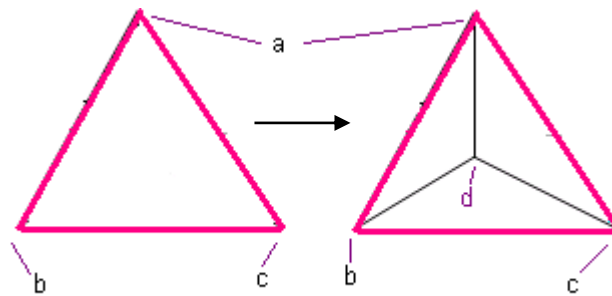


Figure 3.10: Subdivision of a polygon to three small polygons

For example, the triangle that shown in figure above with points a, b and c had been subdivided into three small polygons (adb, bdc, and adc) which d as the middle point.

Current practice for simple cutting algorithm involved polygon deletion. The intersected polygon will be deleted by arranging the polygon id for those are intersected. Obviously, it is not very accurate because the area of skin being deleted is considered bigger than the new improved algorithm.

The improved algorithm will not delete the entire polygon. It is trying to increase the accuracy by not deleting the polygon. This is because in reality cutting does not effect the deletion of the tissue but the tissue skin will squeeze to side or being grab and cut out.

Hence, the new algorithm will practiced as follows:

- When knife or scissor touches the skin, the intersected vertex will deform as similar to grabbing, just the pivot point is the middle of the distance

between two tips. The surrounding vertex will deform according to this tip point.

- Set the threshold for cutting. This threshold determines the range for the degree of the blade to cut the skin or vein.
- After exceed the threshold limit, cutting will performed.
- In the cutting procedure:
 - Find the surface polygon intersect with the tool's blade node path
 - All the intersected polygon will subdivided into 3 small triangle as in figure 3.10
 - Check again the intersected polygon id
 - Assign the sequence polygon id.
 - The polygon will be deleted.
- If the cutting procedure only involved the skin tissue, then just the subdivision of the polygon and the deletion involved instead of deformation on the skin.

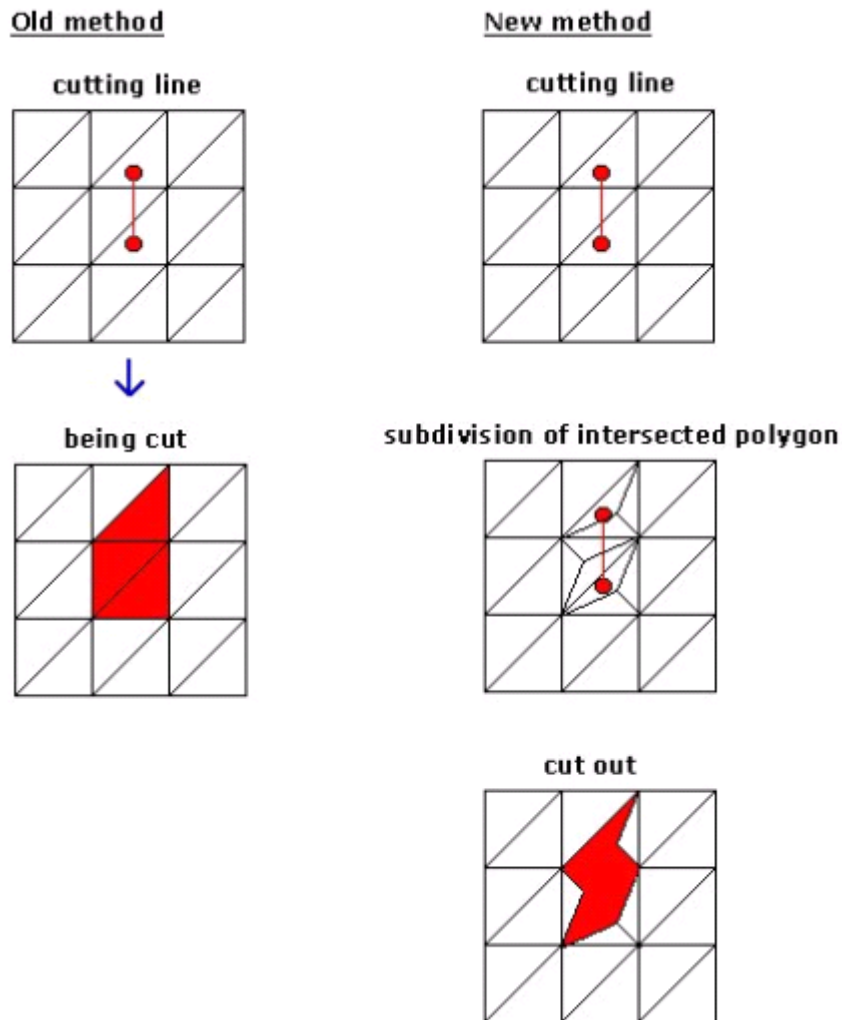


Figure 3.11: Comparison between existing method and the new method

The main objective of this new algorithm is to create a better visual feedback of cutting the cystic duct.

3.5 Conclusion

This chapter has presented the methodologies for several tasks that need to be developed for this study. There are a lot of methodologies that can be used to simulate the tasks involved. However, the simple method had been implemented in this study in order to provide the efficient simulation that able to be supported by the PC-based environment.

Chapter 4 Result and Discussion

4.1 Introduction

In this chapter, the focus will be given mainly to the implementation of the system based on the methodology defined in chapter three. The discussion is not only concerned on the entire flow of the laparoscopic cholecystectomy procedure in the virtual environment but also the details on the core system code that allows the above features and tasks to work effectively.

A variety of measurements were conducted in order to evaluate the effectiveness of the VR laparoscopic cholecystectomy surgery training.

4.2 Results

In this prototype, single window with no specific attributes is created in order to start the simulation. Background color was set to provide a clearer view for the user. Then directed light had been chosen to provide the effect of sunlight that has direction but no position.

Before enabling the simulation, the system need to initializes the position and the orientation of the window and viewpoint. Zooming is also needed to fit the objects into the scenes. In the main body of this prototype, the objects were set to fit into the window. The position of the window, viewpoint and orientation was provided.

Users also have the option to zoom the viewpoint of the given window so that all the geometries in the scene graph are visible by using the function:

```
WTwindow_zoomviewpoint(WTuniverse_getwindows());
```

In this case, the orientation of the viewpoint is preserved.

The entire design of the prototype that had been discussed in section 3.3.1 can be simplified as in figure 4.1.

```
void main(int argc, char *argv[])
{ // Create a default universe
    WTuniverse_new(WTDISPLAY_DEFAULT,WTWINDOW_DEFAULT);
    //Set the background of the scene
    WTuniverse_setbgrgb(50,50,120);
    .....
    // load light and objects
    Light=WTlightnode_newdirected(root);
    .....
    // Zoom window to fit all objects
    WTviewpoint_setposition(WTuniverse_getviewpoint(),ViewPos[0]);
    WTviewpoint_setorientation(WTuniverse_getviewpoint(), ViewOrient[0]);
    WTwindow_setposition(WTuniverse_getwindows(),377,113,672,666);
    .....
    // Setup sensors to control the simulation
    setupSensor();
    // Tie action to the new universe
    WTuniverse_setactions(action);
    // Prepare to enter simulation
    WTuniverse_ready();
    .....
    // Starts simulation loop
    WTuniverse_go();
    // Clean up when the loop stops (after WTuniverse_stop)
    .....
    WTuniverse_delete();
}
```

Figure 4.1: Main body of the prototype

After running the code above, two windows will pop out. One of it will be used to perform the simulation. Meanwhile, another one is the command prompt that will prompt the display menu and others messages to inform the user regarding the current status of the simulation. User is also allowed to view the feedback from the movement and task performed during the simulation.

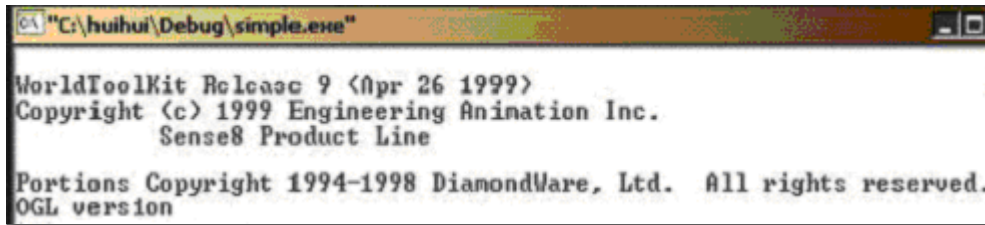


Figure 4.2 Command Window

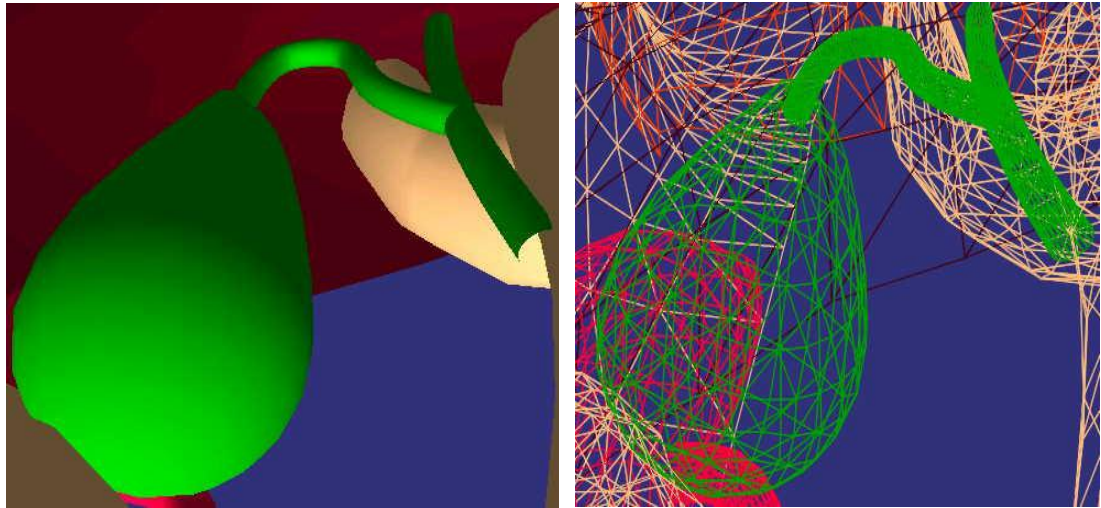
4.3 Implementation on Graphical Objects

In order to provide a meaningful educational experience, a surgical training simulation must depict the relevant anatomy accurately. Since this study is about laparoscopic cholecystectomy, gallbladder with cystic duct becomes main concern for this simulation.

The geometry of the gallbladder was obtained by using 3D Studio Max software packages and imported into WTK environment by using the function below.

```
NodeHumanBody=WTmovnode_load(root, "organs.3ds", 1.0);
```

As mentioned in section 3.4.1, the simulated surface of the model is built from a set of polygons that consists of small triangles. These small polygons can be rendered in wire frame format as shown in figure 4.3(b).



(a) Original display

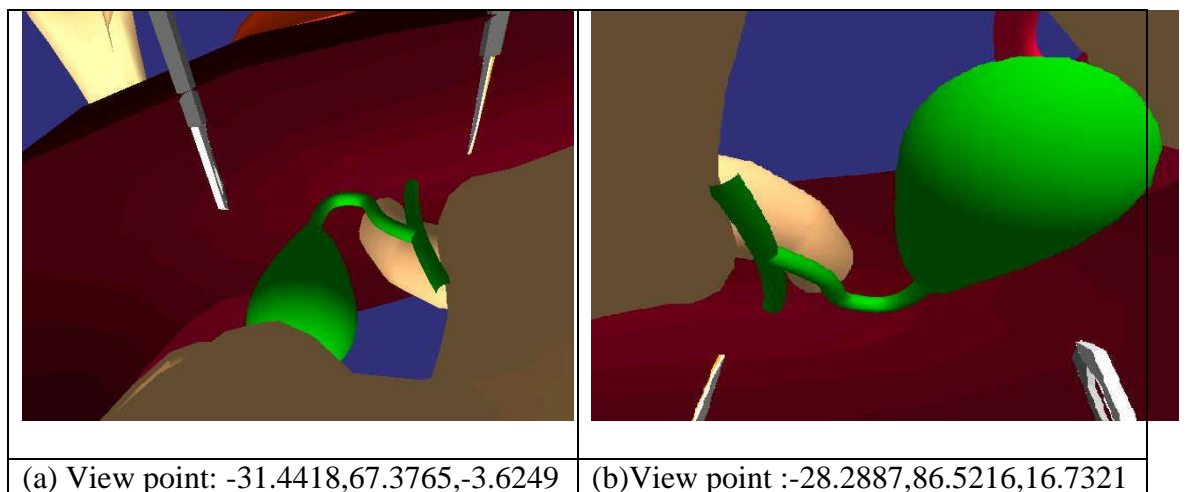
(b) Wire frame

Figure 4.3 Gallbladder display

Even though wire frame views of objects can be drawn very rapidly, they are difficult to interpret, particularly if several objects in a scene overlap. Hence, realism is greatly enhanced when several faces of the objects are filled with colors and surfaces that should be hidden are removed.

4.4 Implementation on View Point and Orientation

Viewpoint and orientation is important to provide a different perspective to the users before users are able to perform surgical tasks.



(a) View point: -31.4418,67.3765,-3.6249

(b)View point :-28.2887,86.5216,16.7321


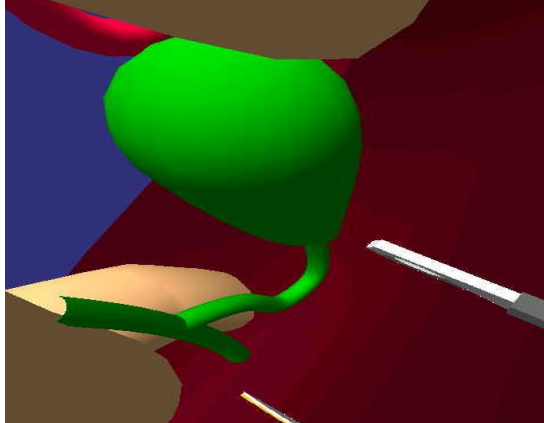
Orientation: 0.3908, 0.0869, -0.0532, 0.9148	Orientation: 0.0619,-0.3213, -0.9443, 0.0340
	
(c)View point: -30.0436,90.5429,21.7095 Orientation: 0.3425, -0.0164, -0.3451, 0.8737	(d)View point: -24.7251, 83.2301, 9.4832 Orientation: -0.0829, 0.2344, 0.8523, 0.4603

Figure 4.4: Different viewpoint and orientation for the gallbladder

In this study, users have the great freedom to view the gallbladder in different viewpoint and orientation. Besides controlling the viewpoint through the keyboard, it also can be controlled by the mouse. This can be done by creating a mouse sensor and attached it to the viewpoint making it possible to move the viewpoint by using the mouse.

```
void setupSensor ( void )
{
    .....
    // Create the mouse sensor
    mouse = WTmouse_new();
    .....
    //Create motionlink with viewpoint using mouse
    Wtmotionlink_new(mouse, WTuniverse_getviewpoints(), WTSOURCE_SENSOR,
        WTTARGET_VIEWPOINT);

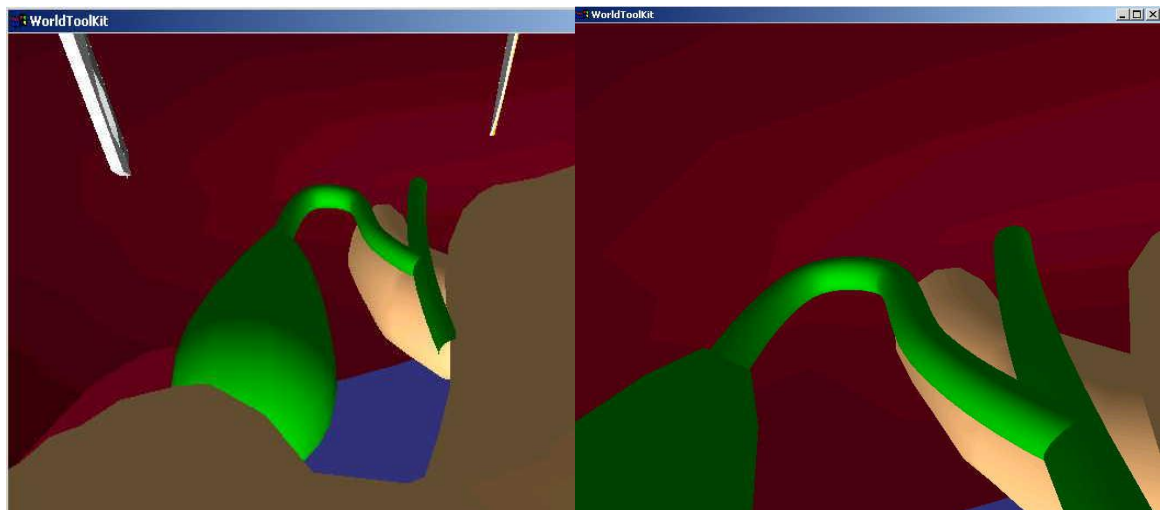
    // Get a pointer to the viewpoint
    view = WTuniverse_getviewpoint();
    .....
    // Attach the sensor to the viewpoint
    WTviewpoint_addsensor(view, mouse);
}
```

Figure 4.5: Sample codes for controlling viewpoint by using mouse sensor

In order to attach the mouse to the viewpoint, motion link also need to be created to connect the mouse of position and orientation information with the viewpoint that moves to correspond with that changing set of information.

Once the motion link is created, position and orientation records from the motion link source (mouse) will automatically cause the corresponding translation and rotation of the motion's link target (viewpoint).

When attaching a mouse sensor to the viewpoint, users also have the flexibility to zoom the particular part of the scene such as cystic duct of the gallbladder as shown in figure 4.6.



(a) Before zooming

(b) After zooming

Figure 4.6: Zooming on cystic duct

4.5 Implementation on Manipulating Laparoscopic tools

The initial model of laparoscopic cholecystectomy consists of two surgical tools that shown in figure 4.7. Grasper is mainly used to grab the surface of the cystic duct. Meanwhile, the scissor is obviously used to cut the cystic duct.

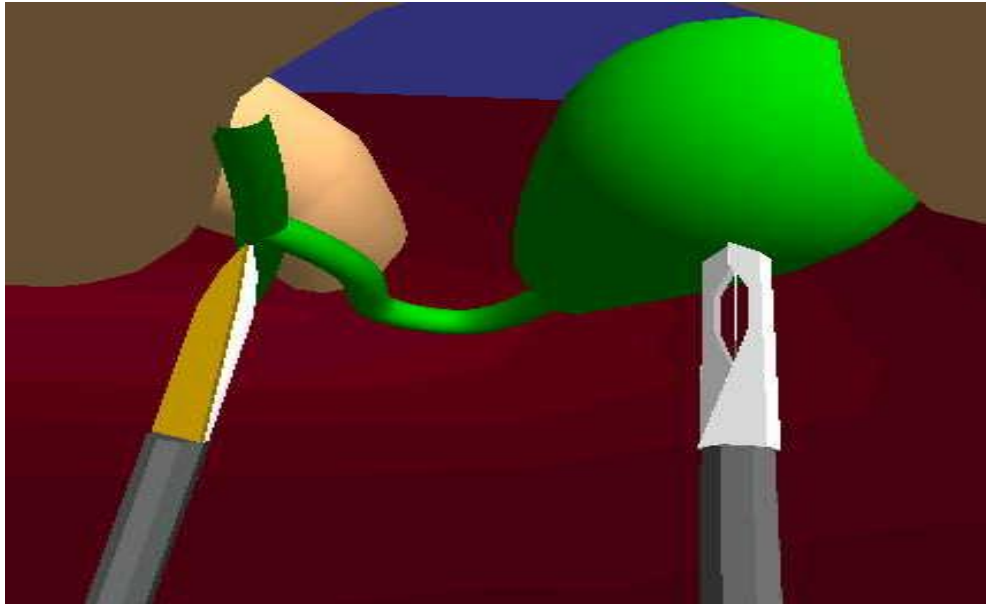
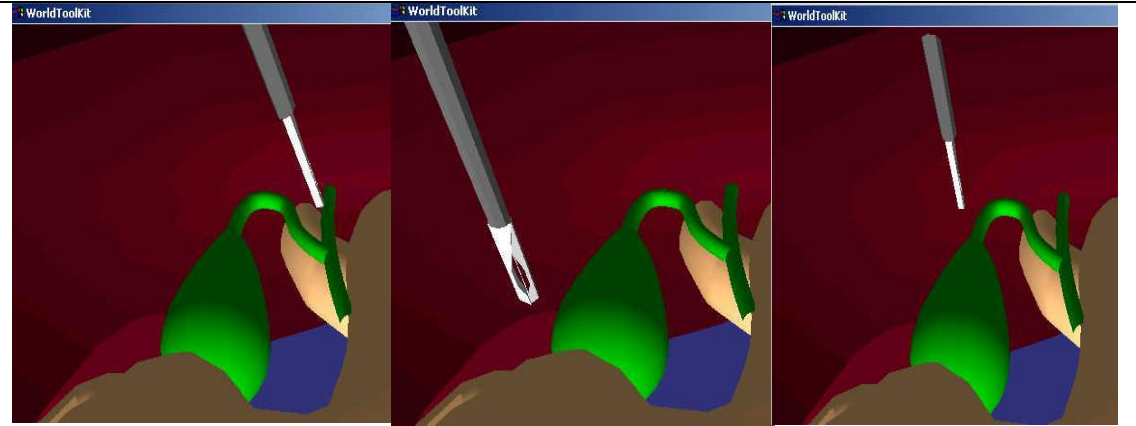


Figure 4.7 Surgical tools involved in laparoscopic cholecystectomy

4.5.1 Implementation on Manipulation of entire surgical tools in simulation window

The surgical tools can perform the movement in the scene with great flexibility. The movements include the rotation and translation. These movements can be done for both tools separately or together.

User can always alternate the key to choose the appropriate tools.



(a) Rotate in x-axis

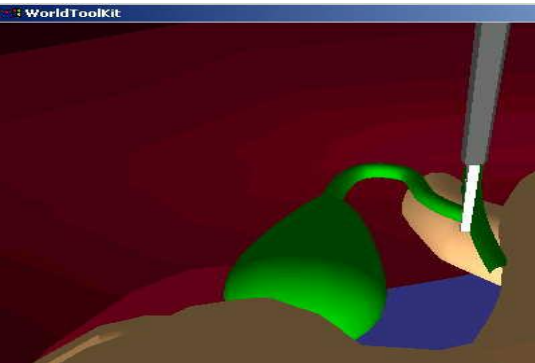
(b) Rotate in y-axis

(c) Rotate in z-axis

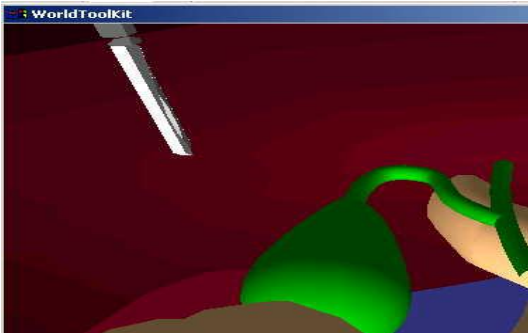
Figure 4.8: Rotation of surgical tools



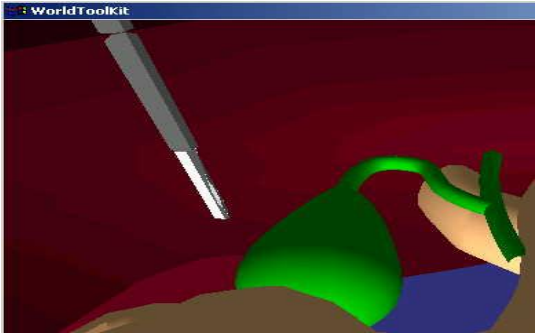
(a) Move to left



(b) Move to right



(c) Move up



(d) Move down

Figure 4.9: Translation of surgical tools

4.5.2 Implementation on Manipulation of part of surgical tools

In WTK, the surgical tool was formed from a group of nodes or objects. Hence, group node need to be created by calling the function *WTgroupnode_new(root)*. Next, all the parts of the tool need to be loaded into the environment by using *WTmovnode_load*. After determine the right position, the entire geometry nodes will be attached to the relevant parent together to form a tool. *WTmovnode_attach* is used to perform this function.

For example, scissor consists of two main parts: shaft with blade and another small blade. First, all parts of the scissor need to be loaded into the environment. By defining the right position of the pivot point of the scissors, the blade is attached to the shaft in the appropriate geometry position. Sample code can be view in figure 4.10. Same concept was applied in creating the grasper.

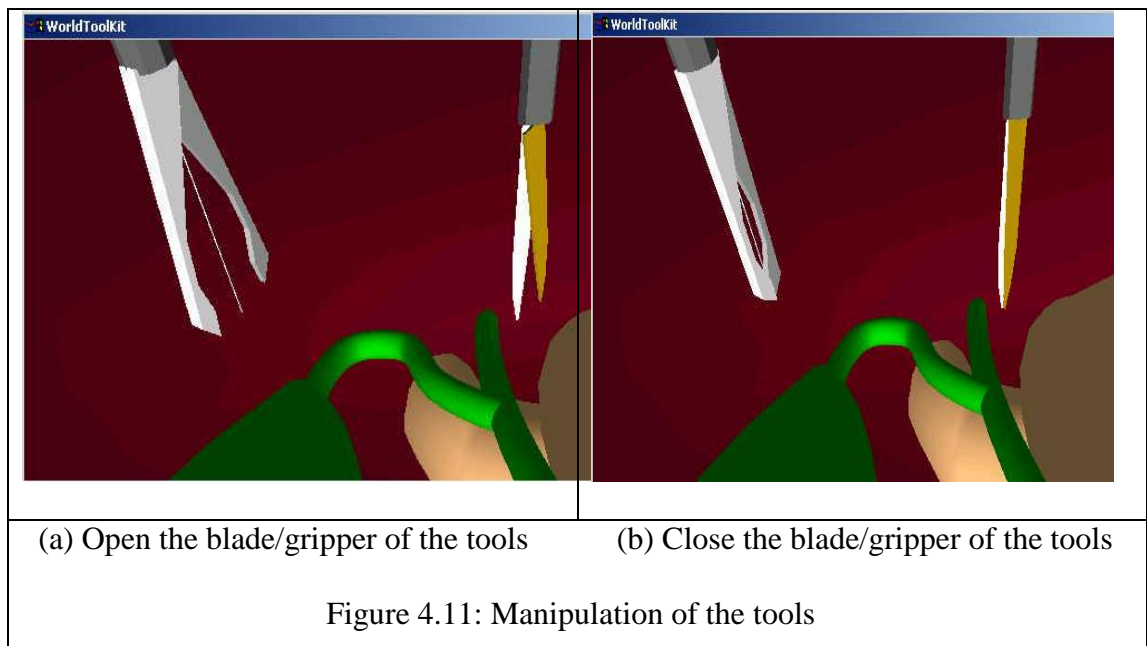
```
//Create a group node
    Scis2.ScisGroup=WTgroupnode_new(root);
// Load all the parts of the scissor
    Scis2.S1=WTmovnode_load(NULL, "Gunting1.3ds", 0.5);
    Scis2.S3=WTmovnode_load(NULL, "Gunting3.3ds", 0.5);
    .....
// correct the pivot point of the scissor handle
    pivot[X]=0; pivot[Y]=22; pivot[Z]=0;
    WTp3_invert(pivot, invpivot);
    WTgeometry_translate(WTnode_getgeometry(Scis2.S3), invpivot);
    WTnode_translate(Scis2.S3, pivot, WTFRAME_LOCAL);
    .....
//Attach the entire parts of the scissor to each other
    WTmovnode_attach(Scis2.B1, Scis2.B2, 0);
    WTmovnode_attach(Scis2.B2, Scis2.B3, 0);
    .....
```

Figure 4.10: Sample codes in creating a scissor (attaching the entire parts of the scissors to each other)

Since the tools are formed from each small part of the objects, user can manipulate part of the tools such as open and close the blade for the scissor or gripper for the grasper. For example, the code below allows the scissor to open the blade in the local frame with 0.03 radian each time in z-axis.

```
WNode_axisrotation(Scis2.S3, Z, 0.03, WTFRAME_LOCAL);
```

The snap screens for manipulation of the tools before performing the surgical tasks are shown in figure 4.11.



4.5.3 Effects on tool manipulation

In WTK, simple test had been done to observe the tool motion in the VR environment by comparing the frame rate. *WTuniverse_framerate()* was called to observe the frame rate of two obvious cases. Firstly, frame rate was taken when entire human torso with all the organs (include gallbladder and cystic duct) and the surgical tools were loaded to the environment for laparoscopic cholecystectomy.

Secondly, the frame rate was taken when only necessary objects are loaded to the environment to perform the surgical task.

Removing the unwanted organs that are not needed from the simulation can reduce the complexity of the environment. The condition whereby only the objects needed in the surgical procedures (gallbladder, scissor and grasper) are left into the virtual environment is preferred in order to provide ‘smoothly’ environment. Removing unnecessary node can be done by calling the function *WTnode_remove (node)*. The result can be observed at the table 4.1.

Test	Load all the objects	Load the objects needed only
1.	13.315579	53.475937
2.	12.376238	54.545456
3.	13.315579	53.475937
4.	13.315579	53.475937
5.	13.315579	54.446461
6.	13.256739	54.545456
7.	13.315579	53.571430
8.	13.198416	55.452866
9.	13.256739	54.446461
10.	13.192612	54.446461
Average	13.185864	54.188240

Table 4.1 Comparing frame rate for objects loaded

Initially, when all the objects and the virtual tools were running on the simulation, the average frame rate achieved was 13.185864 frames per seconds. This frame rate determines the total of 23700 polygons is running on the virtual environment. At this frame rate, the motion of the virtual tools was a little bit jerky. The delay of the tools motion can be seen clearly.

Meanwhile, if only the gallbladder and the virtual tools that needed are loaded into the environment with total of 2076 polygons, the average frame rate was increased to 54.188240 frames per second. With this rate, the movement of the tools is very

smooth compare to previous movement. The user will not be able to notice the delay. The interaction with the gallbladder was also smooth.

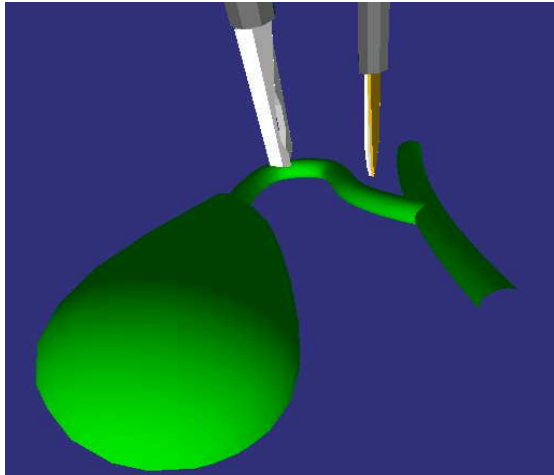


Figure 4.12: Only surgical tools and gallbladder are loaded into the virtual environment

From the study by R. A. Robb (1996), a surgical simulation needs to generate smoothly animated images and the suggested frame rate is 30 frames per second to sustain the illusion of reality. From the results of the simple testing that had been done, can be concluded that the complexity of the virtual environment can determine the effect to the tool's manipulation. However, that is not the only factor that needs to be considered. There are other considerations such as screen resolution, types of colors, types of computer processing system and functions that is used in virtual environment that will also determine the frame rate of the virtual environment. (ANASTOPOULOS 1997)

4.6 Implementation on Collision Management

Collision management is a major concern in this study. This is because it is very important to determine the detection of contact between two objects before being able to perform the next tasks. Collision response will provide appropriate feedback

to the user, regarding the status of the collision. In order to accomplish this, a proper steps of implementation need to be considered.

As mentioned in section 3.4.4.1, the node path of the cystic duct and virtual tools need to be defined first. The following function can be called to create the node path for a particular object:

```
//load the grippers  
Scis1.S2=WTmovnode_load(null,"Grip2.3ds",0.5);  
Scis2.S3=WTmovnode_load(null,"Grip3,3ds",0.5);  
-----  
-----  
//Attach the blade to the appropriate part of the scissor  
WTmovnode_attach(Scis1.B4, Scis1.S2,0);  
WTmovnode_attach(Scis1.B4, Scis1.S3,0);  
-----  
-----  
  
//Create the node path for both of the blade  
path1=WTnodepath_new(Scis1.S2,Root,0);  
path2=WTnodepath_new(Scis1.S3,Root,0);  
-----  
(The node path for the skin had been created previously with same function call  
(PathManipulate=WTnodepath_new(node, root, 0);))
```

Figure 4.13: Sample code for the creation of gripper with node path that ready for collision.

Next, a bounding box intersection test was performed between the gripper's /scissors node path and the cystic duct's node path. If collision was detected, this function will traverse down the specified sub tree in search of the node whose bounding box of the specified node path to that cystic duct node is created and a pointer to it is returned else null will be returned.

```
//check collision detection between surgical tool and cystic duct, the intersection is  
checked for both of the surgical tool's blades  
Laparoscope.touch1=WTnodepath_intersectpoly(path1, PathManipulate);  
Laparoscope.touch2=WTnodepath_intersectpoly(path2, PathManipulate);
```

```

// get the key pressed from the user and set the threshold for the grabbing

if (possibleGrab && Laparoscope.step<1/2) {
    // find the exact intersected poly
    .....
    poly=PolyManipulate;
    poly2=PolyManipulate;
    find=WTpoly_intersectnode(poly, PathManipulate, PathScis);
    find=WTpoly_intersectnode(poly, PathManipulate, PathScis);

    while (!find) {
        // find the next polygon if not found
        poly=WTpoly_next(poly);
        find=WTpoly_intersectnode(poly, PathManipulate, path1);
        poly2=WTpoly_next(poly2);
        find2=WTpoly_intersectnode(poly2, PathManipulate, path2);

        .....
    }
    else {
        // if found the intersected polygon, get the polygon
        poly3 =WTgeometry_id2poly(GeomManpulate,
WTpoly_getid(poly));
        // set the color of the intersected polygon for the left blade
        WTpoly_setrgb(poly3,255,0,0);
        poly4 =WTgeometry_id2poly(GeomManpulate,
WTpoly_getid(poly2));
        // set the color of the intersected polygon for the right blade
        WTpoly_setrgb(poly4,255,0,0);
        .....
    }
}

```

Figure 4.14: Sample codes for collision detection

After the collisions are detected, the appropriate responses are needed to provide the feedback to the user. In this study, collision responses were designed in three aspects. First, the color of the surgical tools via blade or gripper will be changed when collision was detected. This able to provide a significant visual feedbacks. Three colors (white, red and dark yellow) are involved in the collision responses to determine different status. Before collision happens, the blade or gripper is in white color. While the collision happened, the color of the blade / gripper changed from

white to red color. If the scissor or grasper was released from collision, the color will change again from red to dark yellow. The color will alter from dark yellow to red color when collision happens. Snap screens shown in figure 4.17 are examples of displaying the color changing on the blade or gripper developed using sample codes shown in figure 4.15.

```
if (Laparoscope.touch)  
  
    //Change the color of the scissor to red when intersection is detected  
    WTgeometry_setrgb(ChangeColorGeom1, 255, 0, 0);  
else  
    //Change the color of the scissor to orange when the intersection is not detected  
    WTgeometry_setrgb(ChangeColorGeom1, 255, 200, 0);
```

Figure 4.15: Sample codes on changing the color for the surgical tools

Previously, the mentioned algorithm will determine the intersection through the middle index and not through the blade. Hence, intersection is not detected even the blade touch the skin because the middle index is not touching on the skin. Due to this concern, improvement had been done. Current practice involved is by detecting the pivot point of two intersected polygon by the blade (this is performed when the middle pivot point is being calculated), then the polygon intersected by the blade also being colored. This is to ensure that the exact polygon being intersected. The sample codes are shown in the figure 4.14.

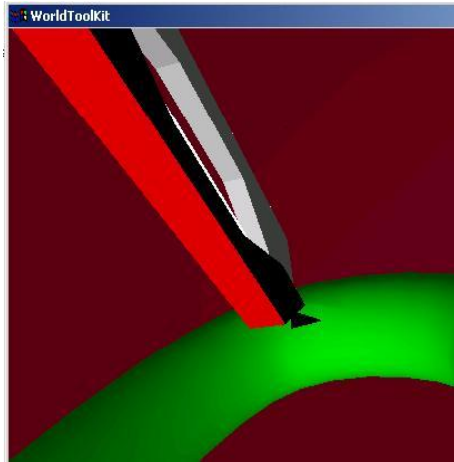
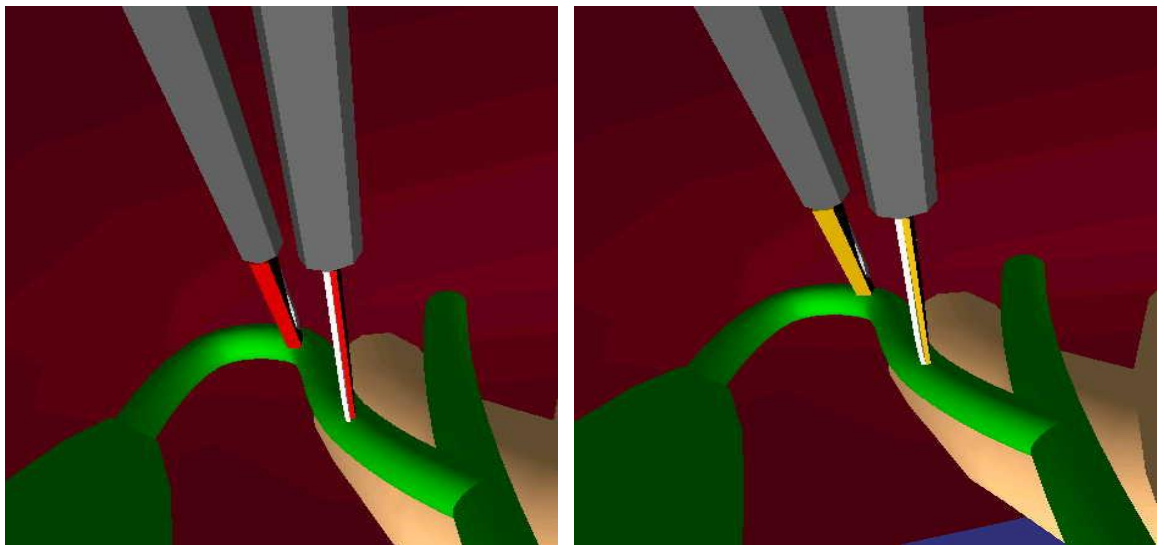


Figure 4.16: The color of the polygon that intersected by blade is changed to black color.

Second, visual feedback also provided through message prompt from command window. The appropriate prompt can always acknowledge the user regarding the current status of the situation. Figure 4.17 shows example of prompt while collision occurred. Currently, the simplest code had being used to show the message prompt such as shown below:

```
WTmessage('Collision detected');
```



(a) Collision response

(b) Response if no collision

Figure 4.17 Collision response on visual feedback

Finally, audio feedback is also provided while collision occurred between the surgical tools and the surface of the cystic duct. Embedding the sound effect in the surgical procedure is very important to alert the user all the time while they are performing the surgical tasks. In order to play the sound, the sound device needs to declare first, and then the proper sound needs to be loaded and play while collision happened.



Figure 4.18: Collision response through message prompt in command window

```
// Open the sound device
    Credev=WTsounddevice_open(WTSOUNDDEVICE_WINMM, 1,
    WTuniverse_getviewpoints());

//Load the appropriate sounds
    .....
    sound[0] =WTsound_load(credev, "Blip.wav");
// Play the sounds
    .....
    WTsound_play(sound[0]);
```

Figure 4.19 Sample codes for sound effect

4.7 Simulation on Grabbing

In order to simulate the grabbing tasks in laparoscopic cholecystectomy, methods used had been discussed in section 3.4.5. However, mainly four steps are involved in grabbing procedures. First, determine the tip point which is the middle of two intersected points.

```

.....
//find the distance between two intersected polygons
distance= WTp3_distance (poly3, poly4)
distance = distance/2;
.....
//determine the new position for the tip point
npoly[x]=poly3[x]+distance;
npoly[y]=poly3[y]+distance;
npoly[z]=poly3[z]+distance;
.....
tippos =npoly;
// set the color for the pivot point
Wtpolysetrgb(tippos, 255,0,0);
.....

```

Figure 4.20: Find the middle pivot point with two intersected polygons

Then, the surrounding vertexes need to be saved into the array for references in step two and three.

```

.....
// get the intersected polygon with the tips of the gripper
verted[0].poly= stpoly;
// get the vertex pointer for the specified polygon
.....
verted[0].vert0= Wtpoly_getvertex(stpoly, 0);
.....
// get the vertex geometry position
Wtgeometry_getvertexposition(GeomManipulate,verted[0].vert0,
verted[0].initpos0);
.....
//get the first polygon of the node
poly= PolyManipulate;
while (poly) {
//get the polygon vertex and compare with the intersected polygon vertex
//save the vertex pointer and raise the flag
if ( (Wtpoly_getvertex(poly,0)==vert) ||
(Wtpoly_getvertex(poly,1)==vert)
|| (Wtpoly_getvertex(poly, 2)==vert)) {
hard_vert[0]= Wtpoly_getvertex(poly, 0);
.....
verted[v_index].poly= poly;
//save the vertex pointer according to the index level and save the
initial vertex position
if (first) {
verted[v_index].firstVert=TRUE;

```



```

        verted[v_index].vert0= hard_vert[0];
        WTgeometry_getvertexposition(GeomManipulate,
        verted[v_index].vert0, verted[v_index].initpos0);
        } else verted[v_index].firstVert=FALSE;
        .....
        v_index++;
    }
    //find next polygon
    poly= WTpoly_next(poly);
}
return(v_index);
}

```

Figure 4.21: Sample codes for saving the intersected polygon and vertex information

After saving all the relevant polygons and vertexes information, grabbing will be performed. This procedure involves the reposition of the vertex that had been saved before. After finding the distance between first level and second level vertex with the tip position, these distances were reduced to half of original distances. Sample code that developed is shown in figure 4.22.

```

.....
{
    //Determine the distance of the vertex with the tip position
    WTgeometry_setvertexposition(GeomManipulate, verted[0].vert0, vertexpos);
    WTp3_subtract(vertexpos, verted[0].initpos0, vector);
    //reduced the distance to half
    WTp3_mults(vector, 0.50f);
    //add the results with the initial position
    WTp3_add(vector, verted[0].initpos1, vertexpos);
    //set the new geometry position
    WTgeometry_setvertexposition(GeomManipulate, verted[0].vert1, vertexpos);
    //repeat the steps to the different level of the vertexes
    for (i=1; i<total;i++) {
        if (verted[i].firstVert) {
            WTp3_add(vector, verted[i].initpos0, vertexpos);
            WTgeometry_setvertexposition(GeomManipulate, verted[i].vert0,
            vertexpos);
        }
        .....
    }
}
}

```

Figure 4.22: Sample codes to set the vertexes to the tip position

Snap screen for the grabbing are shown in figure 4.23. It can be clearly seen that the skin of the cystic duct is grabbed by the grasper and the position of all the skin close to the gripper is recomputed

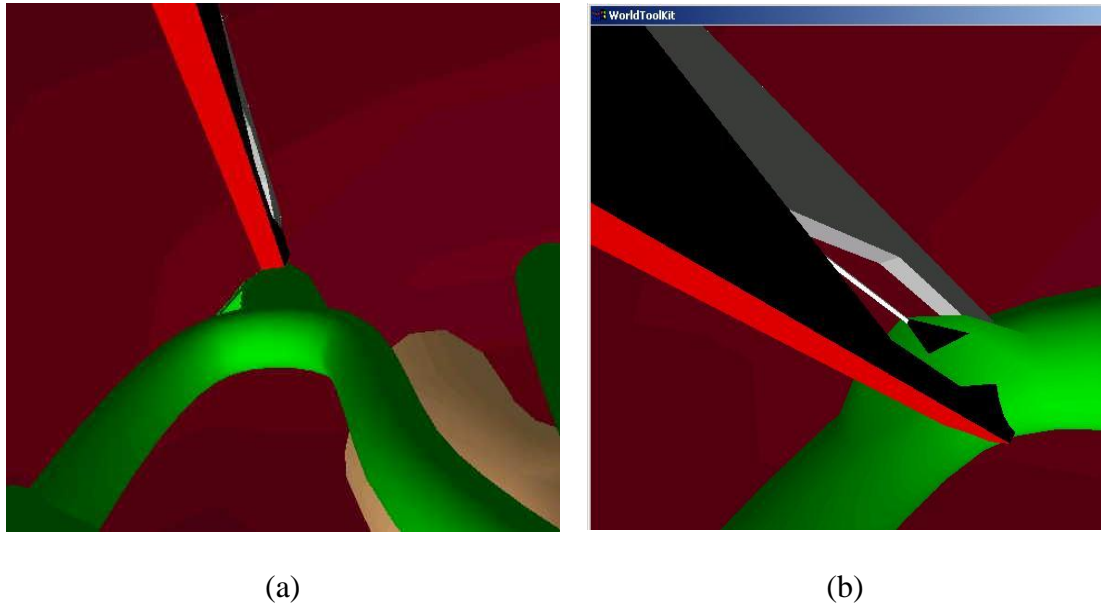


Figure 4.23: Simulation of grabbing on the cystic duct (a), with closer look (b).

Finally, releasing the grabbing should allow all the vertex that set near to the tip position can return to the initial position as before. Hence, another function was developed to set the vertexes to the initial position.

```
.....  
    //set the vertex that near to the tip position back to the initial position  
    WTgeometry_setvertexposition(GeomManipulate, verted[0].vert0,  
    verted[0].initpos0);  
.....  
    //set the vertexes back to initial position  
    for (i=0; i<total;i++) {  
        if (verted[i].firstVert)  
            WTgeometry_setvertexposition(GeomManipulate, verted[i].vert0,  
            verted[i].initpos0);  
    }  
    return;
```

```
}

```

Figure 4.24: Sample codes to set the vertexes back to initial position

4.8 Simulation on Cutting

Simple technique of cutting on cystic duct had been implemented in this study. Before performing the cutting procedure on the cystic duct, some steps had to be carried out in order to determine the correct polygons to be deleted. Steps performed are as below. Sample codes and the output screen are also provided in order to give a clearer understanding on the concept discussed.

First collision detection that had been discussed previously is implemented again for the detection between scissors blade and the cystic duct. After finding the intersected polygons, the mid vertex for these polygons had to be determined in order to produce three small polygons. Hence the algorithm involved are as below:

```
// for each of the intersected polygon
.....
if (find) {
    //get the polygon id
    poly2 = WTgeometry_id2poly(GeomManipulate, WTpoly_getid(poly));
    WTpoly_getnormal(poly, normal);
    .....
    //get the centre point for the polygon
    WTpoly_getcg(poly2, cg)
    .....
    //subdivide the polygons into three vertices
    .....
    //store the vertex for the intersected polygon
    vert0=WTpoly_getvertex(poly2,0);
    vert1=WTpoly_getvertex(poly2,1);
    vert2=WTpoly_getvertex(poly2,2);
    .....
    //start the manipulation of the skin geometry
    for (i=1; i<4;i++){
        npoly[i]= WTgeometry_begin poly(GeomManipulate);
        .....
    }
}

```

```

}
.....
// calling the addvertex function to generate the three sub polygon based on
the vertex found
addvertex(GeomManipulate, npoly[1],cg);
addvertex(GeomManipulate, npoly[1],vert0);
addvertex(GeomManipulate, npoly[1], vert1);
Wtpoly_close (poly1);
Wtpoly_setbothsides(poly1, true);

addvertex(GeomManipulate, npoly[2],cg);
addvertex(GeomManipulate, npoly[2],vert1);
addvertex(GeomManipulate, npoly[2], vert2);
Wtpoly_close (poly2);
Wtpoly_setbothsides(poly2, true);

addvertex(GeomManipulate, npoly[3],cg);
addvertex(GeomManipulate, npoly[3],vert0);
addvertex(GeomManipulate, npoly[3], vert2);
Wtpoly_close (poly3);
Wtpoly_setbothsides(poly3, true);
.....
WTgeometry_close(GeomManipulate);
.....
}

```

Figure 4.25: Sample code for finding the centre point of the polygon and the subdivision of the polygon to three small polygons

Based on the centre point that had been found, a simple function had been called to add to the vertexes for generating new polygon.

```

// function used to add the new vertex to form a new polygon
void addnewvertex(WTgeometry * geom, Wtpoly *poly, WTP3 pos)
{
    WTvertex * vertex;
    vertex=WTgeometry_newvertex(geom., pos);
    Wtpoly_addvertexptr(poly, vertex);
    .....
}

```

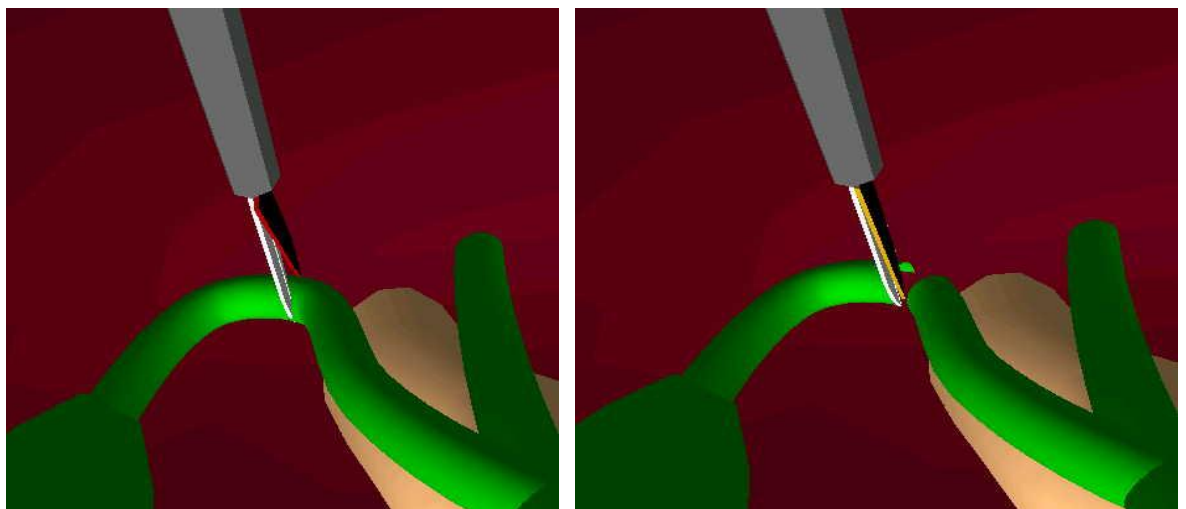
Figure 4.26: Sample codes to add the new vertex to form a new polygon

By using the function above, we can generate three small polygons for each intersected polygon. After the subdivision, collision detection method is called again to find the exact intersected polygon. After that, only the intersected polygon is being deleted.

```
//check for collision detection between the nodepath  
.....  
if (find) {  
    //get the total polygon id  
    k=WTpoly_getid(poly);  
    }  
//based on the total polygon Id, delete all the polygon (cutting)  
  
for (i=k;i<k+100;i++) {  
    poly=WTgeometry_id2poly(GeomManipulate, i);  
    WTpoly_delete(poly);  
}
```

Figure 4.27: Sample code to cut the skin

Snap shots were provided in figure 4.24 to show the cystic duct before and after cutting. The duct was cut smoothly.



(a) Before cutting

(b) After cutting

Figure 4.28: Simulation of cutting

4.9 Validation and Testing

Some validation and testing had to be performed in order to test the accuracy, availability of the application for more user friendly environment and ability to perform the surgical tasks simultaneously.

4.9.1 Performing grabbing and cutting simultaneously

A simple test consisting both grabbing and cutting had been done. The result shows that these procedures can be done simultaneously. However, there are still limitations and will be discussed in the coming chapter.



Figure 4.29: Grabbing and cutting performed on the cystic duct

4.9.2 Testing on the accuracy of the new proposed cutting method with other method

Besides testing on the smoothness of the surgical task, a simple testing is used to check the accuracy of the new improved cutting algorithm. Comparison is done between the old (Salleh, 2001) and new method.

Here, a fixed area simple plane is used to become the sample skin. Two different types of cutting algorithm have being performed on this plane. The accuracy of the methods can be calculated as follows:

- Assume each triangle having the area as 'p' (shown in red color). Hence the total area of the plane is 400p (20 x 20 triangle polygons). Meanwhile, the new triangle polygon (showed as blue color) that subdivided from the old triangle having the area as 1/3p.

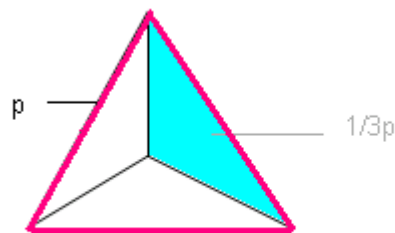


Figure 4.30: Area for the subdivision polygon is 1/3 of the total area, p.

- After perform the cutting using old method,
 - Calculate the number of polygon being deleted, k.
 - Calculate the total area being deleted. (Area each triangle multiply total polygon, $k * p$).
 - Get the total area of the plane after perform the cutting

$$\text{Total area of the plane after cutting, } z = 400p - kp = p(400 - k)$$

- Accuracy of the cutting, c

$$\begin{aligned} c &= z/400p * 100\% \\ &= (400 - k)p/400p * 100\% \\ &= (400 - k)/400 * 100\% \end{aligned}$$

- For new algorithm, repeat the same steps to obtain the total number of polygon been deleted but area of each triangle is 1/3 of the original triangle.

Hence, the area is $\frac{1}{3} * \text{area each triangle} * \text{total polygon}$.

In order to assure the fairness that performed on this two methods,

Based on this algorithm, ten trials been performed to get the relevant data that had been tabulated. And final results had been produced as followed:

No.of Trial	No. of Polygon deleted	Total area for polygons deleted	Total area after polygons deleted	Accuracy (%)
1.	11	11p	389p	97.25
2.	10	10p	390p	97.50
3.	19	19p	381p	95.25
4.	21	21p	379p	94.75
5.	25	25p	375p	93.75
6.	18	18p	382p	95.50
7.	15	15p	385p	96.25
8.	31	31p	361p	90.25
9.	24	24p	376p	94.00
10.	26	26p	374p	93.50

Table 4.2: Results produced by existing method.

No.of Trial	No. of Polygon deleted	Total area for polygons deleted	Total area after polygons deleted	Accuracy (%)
1.	21	7p	393p	98.25
2.	19	6.33p	393.70p	98.43
3.	32	10.67p	389.33p	97.33
4.	42	14p	386p	96.50
5.	50	16.67p	383.33p	95.83
6.	36	12p	388p	97.00
7.	25	8.33p	391.67p	97.92
8.	64	21.33p	378.67p	94.67
9.	42	14p	386p	96.50
10.	50	16.67p	383.33p	95.83

Table 4.3: Results produced by new method

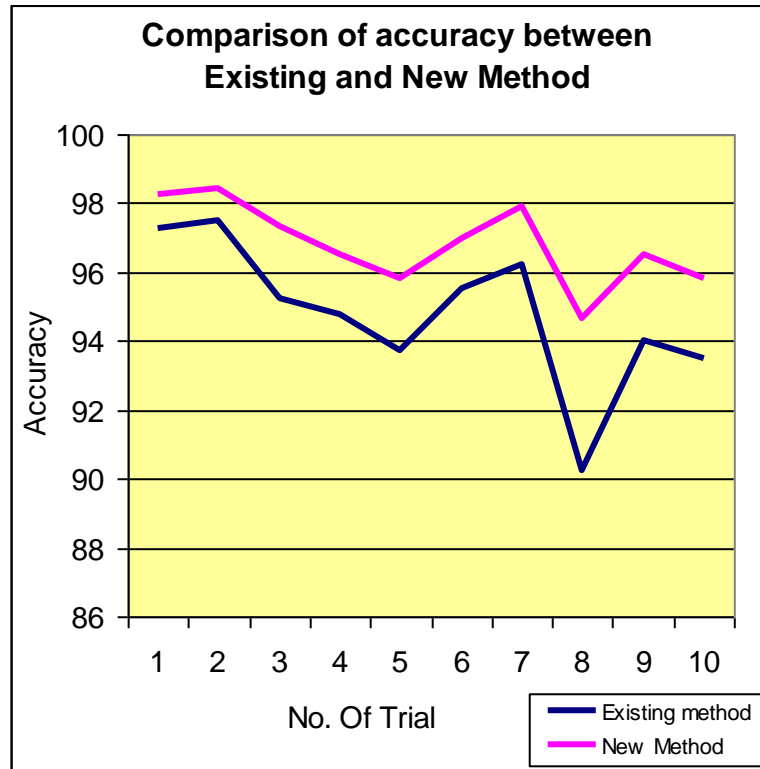


Figure 4.31: Statistic generated based on the comparison of accuracy between existing and new method.

Based on the statistics generated above, found that the new method can achieve higher accuracy as general and the variance is about 2.126%.

4.9.3 Availability of the User Friendly environment

In order to produce a more friendly environment for user to perform the surgical task, Microsoft Foundation Classes (MFC) environment had been introduce to allow the user to perform the surgical task. Example of the environments are shown below:

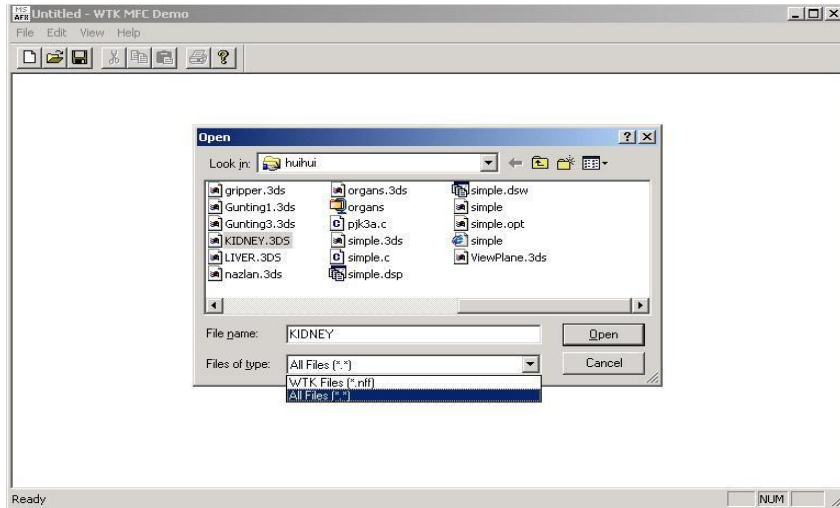


Figure 4.32: Sample WTK MFC environment that allow user to input the relevant organs to perform the surgical tasks

This MFC environment allows user to input the relevant organs and the surgical tool to perform the surgical task. Only the basic functions are currently included for this demo. However, enhancement in this area can be done for future improvement.

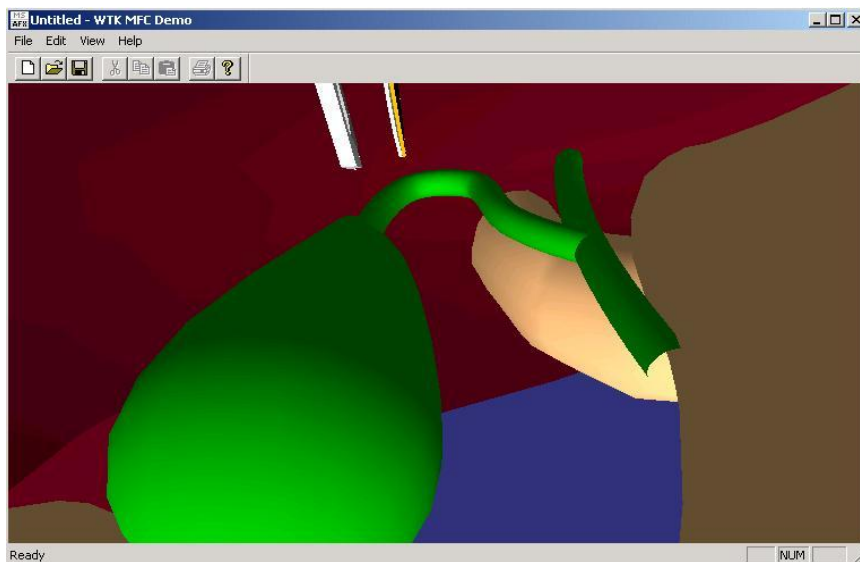


Figure 4.33: Example environment that used to perform laparoscopic cholecystectomy

4.10 Conclusion

As a conclusion, the discussed methods have been implemented in this study. This chapter had been organized to discuss each of the procedures that involved in the laparoscopy. This includes the manipulation of the surgical tools, collision management and surgical tasks involved.

Simple testing also had been conducted to show the effect of the complexity of the environment to the tool's manipulation. Besides, accuracy of the cutting method also included. Most of the codes and snap screens had been presented in this chapter to provide a better references and understanding. However, there are still some deficiencies found during this implementation that will be discussed in the next chapter.

Chapter 5 Conclusion and Future

Enhancements

5.1 Conclusion

Technological advances have made available to medical professionals a wide set of innovative training tools. Among these, VR technology seems to have a great potential to enhance the learning process. It is not only able to improve physician training and performance in term of teaching new advances and procedures in surgery but also access the competence level of surgeons.

The literature review on the laparoscopic techniques and the traditional methods of training has indicated that there are several problems with the current practice of training surgeons in laparoscopic cholecystectomy, which might be solved by using a VR training simulator. In order to develop VR laparoscopic training simulation software, understanding of techniques and methods involved is important in defining the training processes that are to be achieved. Several simulators being developed by researchers had been discussed.

In particular, the main target of this thesis is to concentrate on developing a VR surgical simulation for surgeon's training. WorldToolKit from Sense8 Corporation is used to develop the VR environment. The system used is a low-end PC-based system, which can be afforded by single entity. However, compromise has to be given between high-resolution graphics and speed. If better performance is required, the system can be transported into a high-end system.

Meanwhile, objects include human torso with major organs (gallbladder, stomach and others) and surgical tools were modelled using 3D Studio and imported into the WTK environment. Testing was performed and can conclude that number of polygons in the simulation has to be reduced which means compromising the fidelity of the object in order to achieve a smooth real-time simulation without jerky motion.

The simulation allows the user to grab the tissue and see the effect of the manipulation. The tissue will be deformed when the user pulls the tooling and return to its original status. The techniques of deformation have been implemented by manipulating the vertex. The vertex at the centre of the grabbing spot will follow the tooling, while the surrounding vertices will move to a lesser extent.

Sensory feedback includes the sensation of audio and visual feedback while grabbing/grasping and cutting has been implemented. This becomes a key feature of this thesis.

In the VR simulation, basic skills had been implemented for the surgeon to perform medical operation. These include maneuvering laparoscopic tools, grabbing, touching, and cutting of an object, and also controlling of an endoscope.

5.2 Contribution of this system

The overall output results of the work presented in this study is listed below.

- The major part of this work is to develop the virtual simulation that able to run under low-end PC-based system. Testing was performed and found out that the movement of the tools on the objects was smooth and can achieve up to 54

frames per second if only the gallbladder with cystic duct and surgical tools are loaded into the virtual environment. This already fulfils the requirement to sustain the illusion of the reality.

- Next, the method of collision management includes visual and audio feedback able to provide a sensible way to acknowledge the user regarding the status of the environment while performing the training.
- The method for grabbing motions will deform the tissue in the simulation giving visual feedback sensation as well. Determination of the tip point by using the mid distance of two intersected point with surgical tool's blade is discussed.
- Cutting motion involved also becomes a major contribution from this study. Subdivision of the intersected polygon able to increase the accuracy of the deletion on the intersected polygon.

5.3 Future Enhancements

Although the research is dedicated towards developing a system for the purpose of solving certain shortcomings of the current VR training, deficiencies and limitations were also reported during the evaluation experiments.

Hence, this developed system has also provided a foundation for future enhancement and provides the basis where this thesis can be further improved.

5.3.1 More Comprehensive VR Training

This study only concentrates on the development of VR surgical training simulation that performs the tool manipulations and some simple tasks (grasping and cutting).

However, in reality, performing laparoscopic cholecystectomy still involves many other processes which are not modelled in these simulations such as making holes, inserting cannula, coagulation, ligation of cystic duct and placement of gallbladder in the Pleatman sac for retriever. Hence, it will be a great advantage if the training consists of all the training stages mentioned above.

5.3.2 Realistic Interaction and Behaviour

Many improvements still can be done in virtual organs behaviour. Among the problems is to achieve a highly realistic three-dimensional simulation of human soft tissue behaviour under external stimulation. Imitation of the living soft/hard tissue, having the properties such as elastic (spring), viscous (damper), or plastic deformation can be demonstrated using more advance technique like finite-element method.

Besides, critical problem such as realistic simulation of the interaction between deformable objects and instruments and the manipulation of the virtual objects in real-time also need to be considered in the future.

5.3.3 Realistic Reaction

In this work, a simple method of grasping and cutting has been tested and proved to be relatively effective but still less than perfect impressions of organ interaction and reaction. Currently, a lot of research is being carried out to achieve a realistic organ reaction (such as bleeding) and interaction (such as between surgical instruments and organs). Basdogan and partners (1999) was developing fast algorithms to display the bleeding after the tissue has been cut for laparoscopic

surgery using auxiliary surfaces. This surface flow algorithm can become a great reference for future study in improving the realistic interaction and reaction of the organs in future for this study.

Several virtual models have been proposed to improve the visual impression using techniques such as surface and volume rendering. The use of volume rendering considers the changing of properties inside the organ (not only the surface), to achieve a realistic behaviour and interactivity. A lot of works still needs to be done in this area.

Besides, pulse simulation, irrigation and suction, animation of smoke/steam during procedures can be simulated in future enhancement.

5.3.4 Realistic 3D Models of Human Body/Organs

The models of human body and main organs in this work were modelled manually using the 3D Studio MAX software package. Several problems in terms of actual sizes, shape, locations and complexity of these organs obviously will differ depending on each individual. Methods of using currently available data either from other simulations or data stores need to be explored and developed to permit their import into this virtual environment.

Future works suggest the use of advance medical diagnostic systems such as X-ray Computed Tomography (CT) and Magnetic Resonance Imaging (MRI). These medical devices enable 3D models of the human body to be built with high fidelity. Based on the analysis, diagnosis of the patients and planning for medical operations can be conducted.

5.3.5 Integrating Haptic Feedback

As mentioned previously, this study excludes haptic feedback. Mainly, the user will get the feedback in term of visual or audio. Currently, there are researchers who are able to present their research with haptic feedback. Using specific engine, the sense of force was calculated and fed back to the user allowing him to feel the feedback on his hand during performing the task.

5.3.6 Integrating Tactile Feedback

One of the problems faced by a surgeon who is operating using MIS technique is that the surgeon's finger loses the haptic perception. This is because the surgeon is not directly in contact with the tissue. Besides, the fulcrum at the opening will affect the newly trained surgeons because the direction of motion of the instrument tip opposite to that of the surgeon's hand motion.

In order to solve the problems above, the surgeon can wears the Aesthesia-2000 glove, which is attached at the bottom of the laparoscopic tool. The main purpose is to provide tactile sensation to the surgeon's fingertip. Moving the hand will also move the laparoscopic tool, which can be translated into movements in 3D space of a VR simulation. Opening and closing two of the fingers (i.e. thumb and the index finger) will represent the opening and closing of the gripper at the tip's end.

5.3.7 Pain management

VR systems can aid in pain management by producing sufficient immersion to distract the patient from the pain. This is done by overriding the noxious stimuli

with pleasant sensory input and by modulating pain-gating systems. In this situation, the patient-computer loop must be closed, but the orientation is directed towards control of autonomic responses rather than cognitive interactions. The computer must sense whether the presented stimuli is effective in reducing the pain. It also should be capable of modifying the stimuli routines accordingly.

Bibliography

Aharon, S., Lenglet, C., 2002. *Collision Detection Algorithm for Deformable Objects Using OpenGL*. MICCAI 2002. pp. 211-218

ANASTOPOULOS, G.,1997. *Tele-surgery*. MSc. thesis, Department of Electronic and Electrical Engineering, University of Salford, UK, 1997.

Arvo, J. and Kirk, D., *A Survey of Ray Tracing Acceleration Techniques*. In *An Introduction to Ray Tracing*. pp. 201-262, 1989

Basdogan, C., Ho, C., and Srinivasan, M.A., 1999. *Simulation of Tissue Cutting and Bleeding for Laparoscopy Surgery Using Auxiliary Surfaces*. In: Westwood, J.,et.al. (eds.) *Medicine Meets Virtual Reality*. IOS Press, Amsterdam. pp.38-44

Basdogan, C., Ho, C., and Srinivasan, M.A., 2001. *Virtual Environments for Medical Training: Graphical and Haptic Simulation of Laparoscopic Common Bile Duct Exploration*. IEEE/ASME Transactions on Mechatronics, Vol.6, No.3, September 2001. pp.269-285

Bolton S. *Laparoscope Cholecystctomy*. Available from: <
http://www.laparoscopy.com/pictures/lap_chol.html >. [Accessed 24 February 2004]

Bouma,W. and Vanecek, G., *Collision Detection and Analysis in a Physical Based Simulation*. Proceedings Eurographics workshop on animation and simulation, pp. 191-203, 1991

Bruyns, C., Montgomery, K., *Generalized Interactions Using Virtual Tools within the Spring Framework: Cutting*. Proc Medicine Meets Virtual Reality Conference, MMVR2002, Newport Beach, California, USA, January 23-26, 2002, IOS Press.

Chazelle, B. and Dobkin, D.P., *Intersection of convex objects in two and three dimension*, J.ACM, 34:1-27, 1987

CSIRO, 2002. *Virtual Surgery Across the World*. Media Release: Ref 2002/224. 13 Nov 2002. Available from:<
<http://www.csiro.au/index.asp?type=mediaRelease&id=PrSwedenVirtualSurgery>>. [Accessed 21 February 2004]

Davies, B. W., Campbell, W. B. *Inguinal hernia repair: See one, do one, teach one?*. Annals of the Royal College of Surgeons of England.

Dowens M., Cavusoglu M. C., Gantert M., Way L.W., Tendick F., 1998. *Virtual Environments For Training Critical Skills In Laparoscopy Surgery*. Medicine Meets Virtual Reality, (MMVR'98), San Diego, CA, January 28-31, 1998, IOS Press. pp.316-322

Forest, C., Delingette, H., Ayache, N., 2002. *Cutting Simulation of Manifold Volumetric Meshes*. T.Dohi and R.Kikinis (eds.). MICCAI 2002, pp. 235-244

- Hahn, J.K., Realistic animation of rigid bodies. *Computer Graphics*. pp.299-308, 1988
- Hollands, R.J., Trowbridge, E.A.,1996a. *A Virtual Reality Training tool for the Arthroscopic Treatment of Knee Disabilities*. Proc 1st Euro. Conf. Disability, Virtual Reality and Assoc. Tech., Maidenhead, Berkshire UK, 1996. pp.131-140
- Hollands, R.J., Trowbridge, E.A.,1996b. Virtual Arthroscopic Knee Surgery Simulator. Virtual Reality in Medicine and Biology Group. Available from: <<http://www.shef.ac.uk/~vrmbg/index.html>>. [Accessed 3 February 2004]
- Hubbard P.M, *Interactive collision Detection*. In Proceeding of IEEE Symposium on Research Frontiers in Virtual Reality, October 1993.
- Indelicato D., 1999. *Virtual Reality in Surgical Training*. Darmouth Undergraduate Journal of Science.
- Image Science Group. *LASSO-Laparoscopic Surgery Simulator*. Communication Technology Laboratory, Switzerland. Available from:<<http://www.vision.ee.ethz.ch/projects/Lasso/start.html>>. [Accessed 15 February 2004]
- Jackson Gastroenterology, 2002. *Laparoscopic Cholecystectomy*. Available from: <<http://www.gicare.com/pated/epd0001.htm>>. [Accessed 23 February 2004]
- Jung Kim., 2002. *Virtual Environments for Medical Training: Graphical and Haptic Simulation of Tool-tissue Interactions*. Department of Mechanical Engineering, Massachusetts Institute of Technology. October 2002
- Khalid. A, 2002. *Reorientation of Medical Education for Primary Health Care*. Malaysia, Malasian Medical Association. Available from: <http://www.mma.org/info/l_phc_87.htm> [Accessed 24 February 2004]
- Kuhn, C., 1997. *FORSCHUNGSZENTRUM KARLSRUHE. Karlsruhe Endoscopic Surgery Trainer*. 1997 Jan 16. Available from: <http://iregt1.iai.fzk.de/TRAINER/mic_trainer1.html>. [Accessed 15 February 2004]
- Lambardo J.C., Cani M.P., Neyret F.,1999. *Real Time Collision Detection for Virtual Surgery*. Proceedings Computer Animation, Geneva, Switzerland, 26-29 May 1999. pp. 82-90
- Langreth R., 1994. *Virtual Reality: Head Mounted Distress*. Popular Science 245. November 1994
- Liu, Tendick F., Cleary K. and Kaufmsnn C., 2003. *A Survey of Surgical Simulation: Applications, Technology, and Education*. The MIT Press. Presense. Vol. 12, Issue 6. December 2003

- Mahoney, D.P., 1999. *Getting the Feel of Virtual Surgery*. Computer Graphics World. October 1999.
- Mann, R. W., 1985. *Computer Aided Surgery*. RESNA, 8th Annual Conference, Memphis. 1985
- Mantovani, F., Castelnuovo G., Gaggioli A., Riva G., 2003. *Virtual Reality Training for Health-Care Professionals*. *Cyber Psychology & Behavior*. Volume 6, Number 4, 2003. Marry Ann Liebert, Inc.
- Moline, J., 1998. *Virtual Reality for Health Care: a Survey*. Virtual Reality in Neuro-Psycho-Physiology 1997. IOS Press. Amsterdam, Netherland. Available from: <<http://utenti.lycos.it/dualband/pdf/moline.pdf>> [Accessed 2 April 2004]
- Minimally Invasive Surgical Procedure, 2001 Dec 10. Athena Woman Medical Group. Available from: <http://www.athenawmg.com/laparoscopic_procedures.html>. [Accessed 19 February 2004]
- Mishra R.K., 2003. *Laparoscopy Hospital- History of Minimal Access Surgery*. Available from: <http://www.laparoscopyhospital.com/history_of_laparoscopy.htm>. [Accessed 18 February 2004]
- MIT, 2002. *Laparoscopic Surgery Simulator*. 2002, May 8. MIT Touch Lab. Available from: <<http://touchlab.mit.edu/oldresearch/areas/medsimulation/surgsim.html>>. [Accessed 15 February 2004].
- Naylor, B., Amanatides, and Thibault, W., *Merging bsp trees yield polyhedral modeling results*. In Proc. Of ACM Siggraph, pp 115-124, 1990
- Preparata, F.P. and Shamos, M.I., *Computational Geometry*. Springer-Verlag, New York, 1985
- Primentel, Ken., and Teixeira, A.K., 1995: *Virtual Reality: Through the New looking Glass*. 2nd ed. New York: McGraw-Hill, Inc., 1995
- Richard M. Satava, 1993. *Virtual Reality Surgical Simulator: The First Steps, Surgical Endoscopy*.
- Riva G., 2003 *Application of Virtual Environments in Medicine*. Applied Technology for Neuro-Psychology Lab., Istituto Auxologico, Milan Italy, *Methods Inf Med* 5/2003. pp. 524-534
- ROBB, R.A., 1996. VR Assisted Surgery Planning. *IEEE Engineering in Medicine and Biology Magazine*, Vol. 15 (No 2), March/April 1996, pp. 60-69.

- Rosen J., Hannaford B., 1999. *Force Controlled and Teleoperated Endoscopic Grasper for Minimally Invasive Surgery-Experimental Performance Evaluation*. IEEE Transaction on Biomedical Engineering, Vol.46. October 1999.
- Salleh, R.2001. Minimally Invasive Surgery Training and Tele-Surgery System using VR and Haptic Techniques. Phd Thesis, Telford Research Institute, University of Salford, 2001
- Samet H., *Spatial Data Structures: Quadtree, Octrees and Other Hierarchical Methods*. Addison Wesley, 1989
- SENSE 8 Corporation, 2001. *World Tool Kit Release 10*. Available from <<http://www.sense8.com>>. [Accessed 19 February 2004]
- SENSE 8 Product Line, *WorldToolKit Reference Manual*, Release 9, Engineering Animation, Inc.
- Steffin M, 2001. *Virtual Reality: Overview of its Application to Neurology*. E-Medicine. Available from: <<http://www.emedicine.com/neuro/topic463.htm>> [Accessed 13 February 2004]
- Sourin, A., Tet Sen, H., 1997 *Virtual Orthopaedic Surgery*. Available from : <http://www.ntu.edu.sg/home/eosourina/virtual_surgery.htm>. [Accessed 21 February 2004]
- Surgical Science, 2003. *The LapSim System*. Available from: <http://www.surgical-science.com/_includes/_files/window.cfm>. [Accessed 15 February 2004]
- Suvranu De, Sc.D. Virtual Laparoscopic Surgery Simulation. Department of Mechanical, Aerospace and Nuclear Engineering.
- Szekely G., Brechbuhler C., Dual,J., Enzler, R., Hug J., Hutter R., et al., 2000. Virtual Reality-based simulation of endoscopic Surgery. *Presence*, Vol.9, no.3, June 2000. pp. 310-333
- Versweyveld, L., 1998a. KISMET simulation software forms heart of the Karlsruhe Endoscopic Surgery Trainer. *Virtual Medical Worlds*. 1998 July 6. Available from: <<http://www.hoise.com/vmw/articles/LV-VM-09-98-10.html>>. [Accessed 21 February 2004]
- Versweyveld, L., 1998b. *Endoscopic Surgery Simulator Virgy Interconnects Distant Doctors and Students for Training*. *Virtual Medical Worlds*. 1998 August 11. Available from: <<http://www.hoise.com/vmw/articles/LV-VM-09-98-26.html>>. [Accessed 21 February 2004]
- Voss,G., Hahn, J.K., Muller, W., Lineman, R.W, 1999. *Virtual Cutting of Anatomical Structures*, *Medicine Meets Virtual Reality*, IOS Press, Amsterdam. pp. 381-383

Webster R., Haluck R.S, et al., 2002. *Elastically Deformable 3D Organs for Haptic Surgical Simulation*. Proc Medicine Meets Virtual Reality Conference, MMVR2002, Newport Beach, California, USA, January 23-26, 2002, IOS Press, pp 570-572

Webster R., Haluck R.S., 2003. *A Haptic Surgical Simulator for Laparoscopic Cholecystectomy Using Real Time Deformable Organs*. *IASTED International Conference*. Biomedical Engineering. June 25-27, 2003. Salzburg, Austria. pp.219-222

Weiss D.J., Okamura A.M., 2004. Haptic Rendering of Tissue Cutting with Scissors. Proc Medicine Meets Virtual Reality Conference, MMVR2002, Newport Beach, California, USA, January 15-17, 2004, IOS Press

Zhang, H., Payandeh, S., Dill. J., *Simulation of Progressive Cutting on Surface Mesh Model*. Available from: <
<http://www.ensa.sfu.ca/research/erl/fone/cutting.pdf>>. [Accessed 21 February 2004]