A Multi-Configuration Display Methodology Incorporating Reflection for Real-Time Haptic-Interactive Virtual Environments

Mohammad F. Obeid
Old Dominion University

Follow this and additional works at: https://digitalcommons.odu.edu/msve_etds

Part of the Computational Engineering Commons, Computer Engineering Commons, and the Computer Sciences Commons

Recommended Citation
https://digitalcommons.odu.edu/msve_etds/105

This Thesis is brought to you for free and open access by the Computational Modeling & Simulation Engineering at ODU Digital Commons. It has been accepted for inclusion in Computational Modeling & Simulation Engineering Theses & Dissertations by an authorized administrator of ODU Digital Commons. For more information, please contact digitalcommons@odu.edu.
A MULTI-CONFIGURATION DISPLAY METHODOLOGY INCORPORATING REFLECTION FOR REAL-TIME HAPTIC-INTERACTIVE VIRTUAL ENVIRONMENTS

by

Mohammad F. Obeid
B.S. June 2011, German-Jordanian University

A Thesis Submitted to the Faculty of Old Dominion University in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

MODELING AND SIMULATION

OLD DOMINION UNIVERSITY
August 2013

Approved by:

Frederic D. McKenzie (Director)

Yuzhong Shen (Member)

Gene Hou (Member)
ABSTRACT

A MULTI-CONFIGURATION DISPLAY METHODOLOGY INCORPORATING REFLECTION FOR REAL-TIME HAPTIC-INTERACTIVE VIRTUAL ENVIRONMENTS

Mohammad F. Obeid
Old Dominion University, 2013
Director: Frederic D. McKenzie

There is a natural need for real-time mirror reflection in many interactive virtual reality applications to achieve visuo-haptic collocation through optical reflection. Its use, however, calls for adjustments to the original model since a mirror reverses image characteristics. This thesis introduces a design and implementation of a generic configuration component for virtual environments that adjusts image characteristics and haptic interfaces to accommodate various display setups while preserving the correct orientation and properties of the original models in the graphical scene. Four different but related approaches were developed for image correction and are compared and contrasted against primary criteria. The haptic interface is configured by adjusting the position and quaternion parameters of a Phantom haptic device’s behavior using transformation matrices to account for the different world spaces created when using the various display setups. Furthermore, a method is implemented for synchronously running multiple viewports on separate displays for the same virtual environment. This work accomplishes an important step in the development of a surgical simulator for the Nuss Procedure surgery performed to correct Pectus Excavatum, a congenital deformity of the anterior chest wall, and can be expanded to other similar applications and setups. A Sensable Phantom haptic device controls the motion of the tool used in the simulated surgery. The results are validated and tested for consistency.
This thesis is dedicated to my loving wife Mariam whose unconditional support, encouragement and devotion made all the hardships easier.
With the same amount of appreciation and love, this work is dedicated to my beloved family including my father (Brigadier General) Dr. Fayez Obeid, my mother Suzan, my brothers Obada and Katadah, and my sisters Dema, Dana, Rama and Noor who were all there for me to make me laugh at any time and to support me with prayer whenever I needed it.
Alhamdulillah.
ACKNOWLEDGMENTS

I would like to express my deep appreciation and sincere gratitude to my optimistic advisor Dr. Frederic McKenzie for his continuous support, ongoing guidance and infinite patience. His invaluable assistance has helped me through many of the obstacles that I faced and without mentioning him, this work will be incomplete.

I would like to extend my thanks to Dr. Yuzhong Shen and Dr. Gene Hou for being on my committee and for their treasured insights and advice.

I would also like to mention my family, friends and all those who have, directly or indirectly, helped me complete this work.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LIST OF TABLES</th>
<th>ix</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF FIGURES</td>
<td>x</td>
</tr>
</tbody>
</table>

## Chapter

1. **INTRODUCTION**
   - 1.1 Pectus Excavatum ........................................ 1
   - 1.2 Surgical Simulator ........................................ 2
   - 1.3 Research Question ......................................... 4
   - 1.4 Chapter Organization ...................................... 7

2. **BACKGROUND OF THE STUDY**
   - 2.1 Surgical Simulation ....................................... 8
   - 2.2 Pectus Excavatum ........................................... 11
     - 2.2.1 Nuss Procedure ........................................ 12
   - 2.3 Nuss Procedure Surgical Simulator (NPSS) .......... 14
   - 2.4 Visuo-Haptic Collocation in Virtual Environments .. 14
     - 2.4.1 Haptic Devices ........................................ 15
     - 2.4.2 Mirror Reflection ...................................... 16
     - 2.4.3 Collocation ............................................. 18
   - 2.5 Graphics Architecture Background ................... 19
     - 2.5.1 Viewing Pipeline ...................................... 20
     - 2.5.2 Transformation Matrices ............................. 23
   - 2.6 Software ...................................................... 23
     - 2.6.1 3DVIA Virtools ........................................ 24
     - 2.6.2 Unity3D .................................................. 24
     - 2.6.3 MiddleVR ............................................... 25

3. **RELATED WORK** ............................................. 26
   - 3.1 Virtual Reality in Haptic-Interactive Surgical Simulation ............................. 27
   - 3.2 Haptics and Displays in Virtual Reality ....................................................... 29
   - 3.3 Visuo-Haptic Systems Collocation ............................................................... 31
     - 3.3.1 Mirror-based Displays and Haptics ....................................................... 32
     - 3.3.2 Optical Reflection Collocation .............................................................. 34
     - 3.3.3 Collocation via Video Image Integration ............................................... 42
4. METHODOLOGY                      ................................................................. 45
   4.1 Overview .......................................................... 45
   4.2 Design .............................................................. 46
   4.3 Software ........................................................... 49
      4.3.1 3DVIA Virtools .............................................. 49
      4.3.2 Unity3D ..................................................... 49
      4.3.3 MiddleYR for Unity ........................................ 50
   4.4 Environment and Components ............................... 50
      4.4.1 Adapted Coordinate System ............................. 51
      4.4.2 Virtual Environment Display Setup ................. 52
   4.5 Visuo-Haptic Coordinate Systems Configuration .... 54
      4.5.1 Investigation of Coordinate Systems ............... 54
      4.5.2 Comparison of Translations ............................ 56
      4.5.3 Comparison of Rotations ................................ 58
   4.6 Visuo-Haptic Graphical Configuration .................... 59
      4.6.1 Method 1: Manipulation of Camera Position and Orientation 60
      4.6.2 Method 2: 3D Object Polygon Inversion ............ 63
      4.6.3 Method 3: Render to Texture and Plane Inversion .. 65
      4.6.4 Method 4: Customized Projection of the Graphical Scene 67
         4.6.4.1 Projection Matrix Configuration ................. 67
         4.6.4.2 Back-facing Inversion .............................. 69
   4.7 Code Implementation .............................................. 71
   4.8 MiddleVR Implementation ...................................... 70
      4.8.1 Requirements .............................................. 71
         4.8.1.1 Operating System ................................. 72
         4.8.1.2 Devices ............................................... 72
         4.8.1.3 Unity3D .............................................. 73
      4.8.2 Configuration .............................................. 73
      4.8.3 Importing to Unity Scene ............................... 75
      4.8.4 Viewports and Displays ................................. 76
   4.9 Comparison Design ............................................... 77
      4.9.1 Environment for Testing ............................... 78
      4.9.2 Symmetry Condition ..................................... 79
      4.9.3 Model Preservation ...................................... 80
      4.9.4 Resolution and Clarity .................................. 80
      4.9.5 Speed and Performance ............................... 81

5. RESULTS ................................................................. 82
   5.1 Visuo-Haptic Coordinate Systems Configuration (Task a) ............... 82
   5.2 Visuo-Haptic Graphical Configuration (Task b) ..................... 84
      5.2.1 Method 1 Implementation: Manipulation of Camera Position and Orientation 85
<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Commercial providers of Haptic devices.</td>
<td>16</td>
</tr>
<tr>
<td>2. Vertices and Polygon count for the 3D model components.</td>
<td>78</td>
</tr>
<tr>
<td>3. Evaluation of mirror-image correction methods.</td>
<td>93</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Pectus Excavatum deformity (left) and the Nuss Procedure (right) [8]</td>
</tr>
<tr>
<td>2.</td>
<td>Nuss Procedure Surgical Simulator apparatus</td>
</tr>
<tr>
<td>4.</td>
<td>Thoracosopes with different sizes and angulations [24]</td>
</tr>
<tr>
<td>5.</td>
<td>CT scan in transverse plane (left) and an X-ray image in the sagittal plane (right) for a PE condition [27]</td>
</tr>
<tr>
<td>6.</td>
<td>Nuss Procedure pathway making using Introducer (left) guided by a thoracoscope (right) [30]</td>
</tr>
<tr>
<td>7.</td>
<td>X-ray image of the steel bar in a patient’s chest</td>
</tr>
<tr>
<td>8.</td>
<td>Commercial haptic devices: Falcon, Omni and Phantom (from left to right) [35]</td>
</tr>
<tr>
<td>9.</td>
<td>Demonstration of the Law of Reflection [40]</td>
</tr>
<tr>
<td>10.</td>
<td>An image of a quarter (left) and its horizontally mirrored image (right)</td>
</tr>
<tr>
<td>11.</td>
<td>Visual obstruction elimination in a visuo-haptic scene before (left) and after (right) [42], © [2013] IEEE</td>
</tr>
<tr>
<td>12.</td>
<td>Graphics Viewing Pipeline</td>
</tr>
<tr>
<td>13.</td>
<td>Model frame (left) and world frame (right)</td>
</tr>
<tr>
<td>14.</td>
<td>Parallel (left) and perspective (right) viewing frustums [44]</td>
</tr>
<tr>
<td>16.</td>
<td>Algorithm for determining drilling force magnitude and direction (left) and setup of for petrous bone surgery simulator (right) [55]</td>
</tr>
<tr>
<td>17.</td>
<td>Work Station setup by Schmandt [60]</td>
</tr>
<tr>
<td>18.</td>
<td>Virtual reality scene before (left) and after (right) implementing occlusion effects [63], © [2005] IEEE</td>
</tr>
<tr>
<td>21.</td>
<td>Construction of Virtual back (left) and the Virtual Haptic Back apparatus (right) [66]</td>
</tr>
<tr>
<td>22.</td>
<td>improved virtual back model [67]</td>
</tr>
<tr>
<td>23.</td>
<td>Virtual Haptic Back setup using two mirrors (left) and using a single mirror (right) [13], © [2006] IEEE</td>
</tr>
<tr>
<td>24.</td>
<td>Target location task using a HMD (left) and using a semi-transparent mirror (right) [74], © [2007] IEEE</td>
</tr>
<tr>
<td>25.</td>
<td>Virtual Haptic Back implementing HMD [68]</td>
</tr>
<tr>
<td>26.</td>
<td>Typical Virtual Environment structure with configuration component</td>
</tr>
<tr>
<td>27.</td>
<td>Schematic of a simulator that uses mirror-based projector display</td>
</tr>
</tbody>
</table>
XI

28. Right Handed Rule ....................................................................................................... 52
29. Position and orientation of the Mirror Display camera. ........................................ 53
30. Position and orientation of the Throacoscopic Display camera .................................. 53
31. Coordinate system of Virtools (left) and of Unity3D (right) ........................................ 55
32. Coordinate system of Phantom haptic device [75] ...................................................... 56
33. The positive directions of the coordinate system axes of the scene (left) and those when displayed on the mirror (right). ........................................................................... 58
34. Yaw, pitch and roll rotations of the haptic device [75] ................................................ 59
35. Original view of the patient’s body in the scene (left) and its corresponding image in the mirror i.e. “Baseline Reflectance” error (right). ........................................ 61
36. View of the patient’s body in the scene after rotating the camera (left) and its corresponding image in the mirror i.e. “Horizontal Inversion” error (right) ............... 62
37. Letter R as a part of the texture on the right side of the body (left) and the way it became after flipping its polygons in the Y-direction (right). ........................................ 64
38. Camera's scene rendered as a texture to a plane. ......................................................... 66
39. Triangle ABC and its image mirrored in the Y direction. ........................................... 70
40. Creating a Mouse Tracker in MiddleVR – Devices window ........................................ 74
41. Configuring the tracker created in MiddleVR – 3D Nodes window. ............................. 75
42. Configuring displays and viewports in MiddleVR – Viewports window. ..................... 77
43. View of the patient’s body in the scene replicating the Baseline Reflectance error (left) and its corresponding image in the mirror (right) ........................................ 85
44. Image of body after flipping its polygons in the Y-direction (left) and its corresponding image in the mirror ................................................................. 86
45. Additional camera providing an orthogonal view of the render-textured and inverted plane ............................................................................................................... 87
46. Corresponding image in the mirror of an orthogonal projection of the textured plane ..................................................................................................................... 88
47. Image in Mirror Display using (P) before solving the Back-facing polygon problem. .................................................................................................................. 89
48. Image in Mirror Display using (P) after solving the Back-facing polygon problem. .................................................................................................................. 89
49. Schematic of NPSS structure with configuration component .................................... 90
50. View of Thoracoscopic Display (top left) and Mirror Display (top right) .................. 95
51. Synchronos run of both displays in real-time after configuring two parallel viewports ...................................................................................................................... 95
CHAPTER I
INTRODUCTION

With the development of simulation technology, great emphasis is placed on incorporating virtual reality (VR) and using special devices to increase realism and immersion as well as effectiveness and purpose fulfillment. Such a device or system was developed to simulate the Nuss Procedure surgery for repairing Pectus Excavatum. Pectus Excavatum (PE) is a congenital chest wall deformity that affects children and young adults exhibiting a sunken or funnel anterior chest wall [1].

During this procedure, the highest risk considered is related to puncturing the heart as a pathway is made over the heart. This pathway is used when inserting the corrective steel bar. Thus, the need to develop a simulator that can be used as a platform to enhance that skill for trainees, as well as other significant skills related to this procedure, has driven the development of the surgical simulator. Additionally, not only is thoracoscopy commonly used in such surgeries, but it also has become a routine part of this minimally invasive procedure [2].

Therefore, the surgical simulator should necessarily provide the surgeon with both external and internal (thoracoscopic) views in real-time during the simulated surgery.

1.1 Pectus Excavatum

Pectus Excavatum (Figure 1 left) is a common chest wall deformity in children with which psychosocial as well as physiological effects are associated. Not only does it cause social anxiety and embarrassment, but in severe cases it may impact the function of

---

1 IEEE Computer Society publication format is used for citations and captions of figures and tables.
the cardiac and respiratory system as well as cause pain. It takes the form of a funneled chest with a decreased anterior-posterior dimension of the chest over the sternum [3, 4]. This anomaly occurs in approximately 1 in 400 children and is more common in boys than girls [5]. Several techniques for treating PE have been developed. An increasingly adopted one is the Nuss Procedure. In this procedure, two small incisions are made on either side of the chest to insert a pre-concaved steel bar from the side of the chest. The bar is placed and secured beneath the funneled area of the sternum for support and to push out the sunken part of the ribcage [6, 7]. Figure 1 (right) shows the Nuss Procedure steps for bar insertion (a) and bar flipping (b).

Figure 1. Pectus Excavatum deformity (left) and the Nuss Procedure (right) [8].

1.2 Surgical Simulator

The Nuss Procedure has become increasingly popular, and more surgeons are adopting it. To ensure error-free training for surgeons on this procedure, a surgical simulator can be of tremendous benefit.
To realize this idea, a real-time surgical simulator that utilizes patient specific data was developed as a training aid for the Nuss Procedure surgery. A 3D model of the human chest consisting of the outer surface of the skin as well as the ribcage, lungs and heart has already been built. The model’s behavior was coded and connected to a Phantom® haptic device where the motion of the device’s stylus is interpreted as the motion of the surgical tool appearing in the model [7]. As part of the surgical simulator development, a custom aluminum frame was built as a base on which a 6DOF Phantom haptic device, a 3D monitor, and a semitransparent mirror were installed. Figure 2 shows a picture of the surgical apparatus.

![Figure 2. Nuss Procedure Surgical Simulator apparatus.](image)

During the simulated surgery, the user can observe the patient lying down by looking at the model’s reflection in the mirror or can observe the thoracoscopic view of
the patient’s internal organs displayed on a separate monitor. This is all coupled with the surgical tool’s consistent motion in both views as the user moves the haptic device.

The setup integrates a graphical model with a haptic interface. Such integration calls for a methodology to achieve collocation (consistency) between graphical display (patient avatar) and haptic interface. Solutions that exist in the literature include collocation based on optical reflection, which is the use of a semi-transparent mirror reflection to provide a floating graphical display that the user can work underneath, and another is based on video image integration using cameras to display the virtual scene on a Head Mounted Display.

In the optical reflection solution, a half-silvered mirror is used to reflect the image of the graphical model from the monitor. The user can reach underneath the mirror and grasp the haptic device. Many famous commercial products use this setup including the Reachin Display system [9], SenseGraphics platforms [10] and Immersive Touch [11]. This approach is the one implemented for the surgical simulator in the presented work.

1.3 Research Question

During the development of the Nuss Procedure Surgical Simulator, an obstacle was faced regarding the mirror-based display and the user’s interaction with the virtual environment. What was explained thus far leads now to substantiating the existing problem which is the driver of this work. The following is a brief explanation of the problem faced as well as the solution developed.

Fundamental to the Nuss Procedure surgery is the need to insert surgical instruments and the steel bar starting from the right side of the chest and then traversing
to the left side, carefully avoiding the patient’s heart. This is where the need arose to develop a multi-configuration display convertor in a haptic device-controlled virtual environment to quickly and easily switch between the original view and a mirror reflection. This is the contribution this work aims to realize. Another component that this work implements is having both an external patient view as well as the thoracoscopic view available synchronously in real-time as the simulated surgery is performed.

A mirror can be a very useful tool, but it has a behavior that can lead to a confusing perception of the image. Depending on where the image is situated, a mirror image is inverted either across a vertical or a horizontal axis. In both cases, the image will be disrupted and may not be usable as is within the same application [12]. As mirror reflection is commonly used in interactive virtual environments, an approach that adjusts the image to account for the mirror’s behavior can be of great use.

To achieve seamless mirror use across multiple configurations, a methodology needs to be developed to compensate for the anomalies of mirror reflection. Such a methodology must account for the Cartesian differences so that the world coordinate system of the mirror view and the normal view will be aligned with the world coordinate system of the haptic device to allow for consistent and correct motion.

Ji et al. at Ohio University faced similar problems during the development of the Virtual Haptic Back Project as the image of the virtual back is flipped up and down. They approached that problem by using two mirrors in one attempt and by using screen adjustment software in another. They also had the problem of the Cartesian differences which required coordinate transformations between the HMD, the transmitter and the 3D
object [13]. Many other researchers have come across the same issue, making image impairment caused by mirror reflection a common problem [14-18].

When the interactive simulator is running, surgical instruments can be controlled using the haptic device stylus, and to illustrate the patient lying down, the model’s projection is perceived on the mirror. The capability of viewing the model on the monitor is also required for ease of use in running the simulator. However, displaying the model on the monitor would need a different configuration and orientation than when it is projected on and reflected off the mirror. Therefore, a solution is required to be able to move in between these two environments as well as being able to run them both simultaneously.

The presented thesis will mainly tackle three main tasks. Throughout this work, the following three tasks (a, b and c) will be referred to several times:

a) Achieving consistent coordinate system alignment between the two types of displays (Mirror display and thoracoscopic display) and the haptic device.
b) Correcting the flipped image displayed on the mirror to result in a correctly-oriented graphical scene.
c) Running both displays at the same time each on a separate viewport in real-time.

By accomplishing these three tasks, a generic configuration component can be constructed to be used with any similar application to make the necessary adjustments to the image of the graphical scene as well as to adjust the coordinate system and behavior of the haptic device. This work will provide the methodology of developing such a component which can be integrated into any similar virtual environment application.
1.4 Chapter Organization

There are six chapters in this thesis: Introduction, Background, Related Work, Methodology, Results, and Conclusions and Recommendations. A general description of the project and its motivation as well as an explanation of the problem and solution are introduced in Introduction; Background introduces the terminology, ideas, and publications related to this project; Related Work covers and analyzes some research done by others that is related to this work; the actual steps of how the problems are solved are given in Methodology; Results shows the experimental results of using the methods proposed to solve the problem. Finally, Conclusions and Recommendations gives a summary of the accomplished objectives and discusses possible improvements and expansions in the future.
CHAPTER 2
BACKGROUND OF THE STUDY

To build a knowledge infrastructure for the succeeding chapters and to better understand the topics to be discussed, the theoretical background is now introduced. This chapter presents a thorough description of the main aspects in the literature that this thesis is concerned with and defines the main themes of research that the rest of this work will focus on and refer to.

2.1 Surgical Simulation

Simulation in general can be defined as an exercise that enables the user to reproduce a specific phenomenon from a real system. Surgical simulation is an instructional strategy utilized to transfer required surgical skills and procedures by involving the learners in scenarios that emulate actual realistic situations. Surgical simulation and planning can be utilized to minimize surgical risks. It is generally used to train medical students on specific procedures to achieve eye-hand coordination and the ability to learn the primary steps of the procedure hands-on. In addition, surgical simulation allows for a repetitive performance of a specific task to enhance the trainee’s skills, speed and efficiency.

There are several types of surgical simulation. One type uses physical synthetic models and box trainers. In this type, plastic or rubber objects are used to simulate organs and pathologies on which the trainee performs the procedure by manipulating these
simulated (synthetic) tissues and parts. In box trainers, the actual surgical instrument and optical devices are usually used to reflect a realistic surgical sensation.

Another type of surgical simulation is using live animal models. This type provides a realistic non-patient environment where the trainee can develop simple skills required in the operating room. Animals range in anatomical differences and similarities to humans which makes some animals better simulation candidates than others.

Another type of surgical simulation is the use of human cadavers to provide the trainee with a detailed understanding of human anatomy. In addition to the benefits in anatomical teaching, human cadavers are used to train procedures including laparoscopy, endoscopy and others [19, 20].

Another type of surgical simulation is utilizing computer technology to develop simulations of surgical procedures. Virtual reality (VR) is being increasingly used in surgical simulation where computer-based models are coupled with dynamic interactions to provide a specific simulation interface. The advantage of using VR for surgical simulation is that it provides a platform for collecting objective metrics such as completion time, number of errors and other measures for speed and efficiency.

Realistic tactile feedback is being increasingly used in virtual reality surgical simulation to increase fidelity where one or two haptic devices are employed. Such a virtual surgical simulation is what Choi et al. [21] have developed for the phacoemulsification procedures of the cataract surgery where two haptic devices are used. Features such as surface cutting and tissue deformation were made available and the procedure-related tasks can be performed through the use of two 6DOF haptic devices (one for each hand) incorporating the virtual environment displayed on the monitor.
The human eye and its components were constructed as 3D meshes and algorithms for tissue deformation, surface mesh cutting, and volume sculpting were constructed to simulate the phacoemulsification procedure. When running the simulation, the user experiences the response and deformation of the tissues upon contact by adapting a mass-spring model. Figure 3 shows the simulator’s setup.

![Figure 3. Setup of Cataract Surgery Simulator [21].](image)

Laparoscopic and endoscopic surgeries are becoming the main emphasis of VR surgical simulator development. Although the skills needed for these procedures can be taught during an operation setup, simulation allows trainees to practice these skills before entering the operating room [22, 23].

Thoracoscopy is a similar medical procedure involving internal examination, biopsy or disease resection within the pleural cavity and the thoracic cavity. It was pioneered by Hans Christian, a Swedish internist in 1910. Today, thoracoscopy is performed using specialized thoracosopes. A thoracoscope (Figure 4) includes a light
source and a lens for viewing and is connected to a monitor to provide video-assisted surgery.

Figure 4. Thoracoscopes with different sizes and angulations [24].

2.2 Pectus Excavatum

Pectus Excavatum (PE) (a Latin term meaning hollowed chest) is a congenital deformity in which several ribs as well as the sternum grow abnormally causing depression of the anterior chest wall. In this condition, the sternum articulates posteriorly toward the spine (Figure 5). This deformity is often observed at birth and can develop with growth [25]. PE is the most common chest wall anomaly in children with an incidence of 1 in 400 children and is more common in boys than girls. PE patients often experience limited cardiac and respiratory function as well as social anxiety and psychological stress [1, 26].
2.2.1 Nuss Procedure

Several techniques have been developed and adapted for PE correction including the Ravitch technique, the Robicsek technique and the Vacuum bell. The minimally invasive repair of Pectus Excavatum (MIRPE), often referred to as the Nuss Procedure, was developed by Dr. Donald Nuss of the Children’s Hospital of the King’s Daughters, Norfolk, VA. This technique has recently become more and more popular as it requires no cartilage incision or sternal osteotomy. It was performed in 1987 for the first time, and in 1998, a 10-year experience of the technique was reported in the *Journal of Pediatric Surgery* [25, 28, 29].

The surgery is performed by making two small incisions on either side of the chest. An Introducer is then inserted from the right side and pushed along posterior to the sternum and ribs and anterior to the heart and lungs to make a pathway to the other side of the chest (Figure 6 left). A third incision is made to insert a thoracoscope to provide an
internal view and guide the procedure (Figure 6 right). A pre-concaved stainless steel bar is then inserted under the sternum through the pathway. The bar is then flipped causing the sternum to pop out, and the bar is fixed in place and supported by a stabilizer and PDS sutures (Figure 7). After a period of two to four years, the bar is removed from the patient’s chest.

Figure 6. Nuss Procedure pathway making using Introducer (left) guidded by a thoracoscope (right) [30].
2.3 Nuss Procedure Surgical Simulator (NPSS)

As the Nuss Procedure has become increasingly popular and more surgeons are adapting it, a need to ensure adequate training for surgeons on this procedure was triggered. A Nuss Procedure Surgical Simulator (NPSS) was developed to provide that training platform [7, 31, 32]. The simulator utilizes patient specific data to generate a 3D model of the human chest and the associated organs including the ribcage, lungs and others. The setup consists of an aluminum frame on which a 6DOF Phantom haptic device, a 3D monitor, and a semitransparent mirror were installed in addition to a separate monitor (Figure 2). This thesis’s contribution is a significant step in the development of the Nuss Procedure Surgical Simulator (NPSS).

2.4 Visuo-Haptic Collocation in Virtual Environments

In modern technology, instead of manipulating a haptic device in one space and looking at the graphical model on a separately located monitor, visual and haptic displays
are combined together (collocated) in the same spatial location to increase realism and fidelity. A visuo-haptic system can therefore be defined as a system that allows the user to see and touch a virtual object in the same space. The NPSS uses such a system; therefore, the main concepts involved with the system will be explained.

2.4.1 Haptic Devices

The word “haptic” comes from the Greek word *haptesthai* which means “to touch.” Haptic technology, or haptics, is a tactile feedback technology that provides touching sensation by applying forces or vibrations. Haptics are used in virtual environments extensively to take the user’s experience beyond the visual aspect enabling him/her to control, interact with and manipulate virtual objects using stylus-based haptic rendering. In other words, the user’s interface with the virtual world is through a stylus, representing a specific form of tool. A haptic device (Figure 8) is a mechanical three-dimensional input device structured with tracking encoders and force feedback to integrate the sense of touch to the user’s experience [33, 34].

Figure 8. Commercial haptic devices: Falcon, Omni and Phantom (from left to right) [35].
Commercially sold haptic devices track the user's motion in space and provide force and/or torque feedback according to that motion. Such commercial providers are listed in Table 1 as well as the devices they offer. In the NPSS, a Phantom Premium 1.5 High Force (6DOF) haptic device is used.

<table>
<thead>
<tr>
<th>Company</th>
<th>Devices</th>
<th>PS DOF</th>
<th>FF DOF</th>
<th>Price $x1000</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensable Technologies</td>
<td>Omni</td>
<td>6</td>
<td>3</td>
<td>2.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Desktop</td>
<td>6</td>
<td>3</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Premium 1.5</td>
<td>6</td>
<td>3, 6</td>
<td>27, 54.5</td>
<td>[35]</td>
</tr>
<tr>
<td></td>
<td>Premium 1.5 HF</td>
<td>6</td>
<td>3, 6</td>
<td>32.5, 59.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Premium 3.0</td>
<td>6</td>
<td>3, 6</td>
<td>60.5, 79.5</td>
<td></td>
</tr>
<tr>
<td>Novint</td>
<td>Falcon</td>
<td>3</td>
<td>3</td>
<td>0.25</td>
<td>[36]</td>
</tr>
<tr>
<td>Force Dimension</td>
<td>Omega 3, 6, 7</td>
<td>3, 6, 7</td>
<td>3</td>
<td>23, 29, 37</td>
<td>[36]</td>
</tr>
<tr>
<td></td>
<td>Delta 3, 6</td>
<td>3, 6</td>
<td>3, 6</td>
<td>35, 64</td>
<td>[37]</td>
</tr>
<tr>
<td></td>
<td>Sigma 7</td>
<td>7</td>
<td>7</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td>Mimic</td>
<td>Mantis duo</td>
<td>7</td>
<td>3</td>
<td>38</td>
<td>[38]</td>
</tr>
<tr>
<td>Immersion Corp</td>
<td>CyberForce</td>
<td>6</td>
<td>3</td>
<td>58</td>
<td>[39]</td>
</tr>
<tr>
<td></td>
<td>CyberGrasp</td>
<td>5</td>
<td>5</td>
<td>58</td>
<td></td>
</tr>
</tbody>
</table>

PS DOF: Degrees of freedom of position sensing, does not include force.
FF DOF: Degrees of Freedom of force feedback

2.4.2 Mirror Reflection

When a ray of light strikes a mirror, it is reflected. Reflection involves a change in direction of the light ray. The direction of the reflected ray is expressed by the two angles that the ray makes with the normal line drawn on the mirror’s surface. The angle of incidence is the angle between the normal line and the incoming ray, and the angle of reflection is the angle between the normal line and the reflected ray. According to the law of reflection (Figure 9), these angles are equal [40].
When using a mirror in a display setup, the same process explained about light rays takes effect. When a display's image is projected on a mirror, the image is flipped in the axis of projection about the normal of the mirror plane. Depending on the orientation, the image will either be inverted in the horizontal or vertical direction [12]. Figure 10 shows an image of a quarter (left) and its mirror image reflected across a horizontal axis (right) respectively.

In the NPSS setup, the LCD’s image is projected on the semi-transparent mirror causing the image to flip horizontally. Taking into account the angles of incidence and
reflection, the LCD is angulated in a way that allows the full image to appear on the mirror.

2.4.3 Collocation

When incorporating a virtual environment with haptic interactions, the issue of consistency between the visual and haptic (visuo-haptic) interfaces is crucial as otherwise the realism factor will be negatively affected as well as causing incorrect interactions.

The idea of touching and seeing objects in the same space was introduced and pioneered by Yokokohji et al. [41] as they developed what they called a What-You-See-Is-What-You-Feel (WYSIWYF) Display. Their solution was primarily based on the idea of extracting an image of the user’s real hand and integrating it with the virtual world both visually and haptically. Collocation can therefore be achieved through a video image integration solution where a video camera is used to capture the movement of the hand, and the image of the hand is extracted and integrated with the virtual scene using a virtual reality device with tracking capability such as the Head Mounted Display (HMD).

This technique was adapted by many for visuo-haptic collocation based on video image integration. One example is the most recent work of Cosco et al. [42] where they developed a visuo-haptic mixed reality setup with association of a proposed technique for eliminating visual obstruction in the visuo-haptic scene (Figure 11).
According to the literature, visuo-haptic collocation can be achieved through multiple ways other than video image integration: through optical reflection, where the virtual image partially occludes the real scene; through stereo monitors; through immersive projection systems[15, 43].

In the optical reflection solution, a half-silvered mirror is used to reflect the image of the graphical monitor allowing the user to see the reflected image of the graphical scene as well as his/her hand through the mirror in the same frame. The NPSS implements this solution to achieve visuo-haptic collocation and to allow the user to move the surgical tool in the virtual scene as he/she physically moves his/her hand in the same view.

2.5 Graphics Architecture Background

In graphics architecture, the viewing pipeline is a group of processes concerned with transforming objects to be displayed from a specific viewpoint and removing
surfaces that cannot be seen from this viewpoint. This section will discuss this topic and introduce the transformation matrices used to complete that process.

2.5.1 Viewing Pipeline

The input to the viewing pipeline is a list of geometric primitives representing the objects in addition to camera position and illumination models (Figure 12). In a typical rendering environment, each object is defined in its own local coordinate system, also called model frame, where the origin is a specific point on the object itself like a center of a sphere or upper right corner of a cube. This coordinate system is created in the modeling stage where the geometric primitives are drawn and the object is constructed (Figure 13 left).

![Figure 12. Graphics Viewing Pipeline.](image)

After the objects are created, they all need to be defined in the overall scene. Therefore, the first stage of the viewing pipeline is to convert each object from its local coordinate system into the world coordinate system, also called world frame. This is called model transformation (Figure 13 right).
In the modeling transformation stage, a transformation is applied to each object individually to convert its coordinates to a common space. For instance, a translation of $[5, 4, 9]$ and a clockwise rotation around the Y-axis by 24 degrees are applied to an object to convert its coordinates. Such a transformation will impact each vertex of the object resulting in a new coordinate representation for it. Doing that to all objects will result in a scene composed of all the objects defined in common coordinate system which allows for consistent bulk transformation of the whole scene.

The camera parameters are inputs to the graphics pipeline. In the next stage, these parameters include the position and orientation of the camera as well as the viewing plane. The setup of the camera and the definition of its position allows for defining a camera coordinate system, also called camera frame. Another parameter to define is the viewing volume (viewing frustum). To define the viewing volume, the camera can be configured to either perspective or parallel projection.

In parallel (orthogonal) projection (Figure 14 left), the polygons are projected onto the viewing plane using parallel lines called projectors. When using this setup, as a
line of fixed length moves away from the camera it will be drawn with the same size. Therefore, parallel projection does not provide a sense of depth. In perspective projection (Figure 14 right), polygons are projected onto the viewing plane using projectors that connect each point from the scene to a single point referred to as the center of projection. Using this setup, a line of a fixed length will be drawn shorter as it moves away from the camera. Therefore, perspective projection provides a sense of depth.

![Figure 14. Parallel (left) and perspective (right) viewing frustums [44].](image)

After the camera has been configured, the world frame is converted to the camera frame using what is called view transformations. Doing that results in a coordinate definition of the scene in the coordinate system of the camera.

After modeling transformation and viewing transformation, the third stage of the pipeline is clipping the scene to a view volume. In this stage, the objects are clipped according to the defined viewing volume. and any portions of the scene that fall outside the viewing volume will be removed and not displayed.
The fourth stage of the pipeline is the projection. In this stage, the coordinate system of the 3D scene is converted to a coordinate system of the 2D projection plane resulting in the image being displayed on the screen. This is done by using what is called *projection transformations*. After that, the 2D image is mapped to the viewport by scaling it to fit the display device [45-48].

2.5.2 Transformation Matrices

As mentioned before, the first step involves using model transformation to convert each object’s local coordinate system into the world coordinate system, i.e. from the model frame to the world frame. The second step involves using viewing transformation to convert world coordinates into camera coordinates, i.e. from world frame to camera frame. These two steps are done using matrix multiplication to result in a *model-view matrix* which represents the scene in the coordinate system of the camera.

In the next step, projection transformation is used to convert from the camera frame to the screen. This is done by multiplying the *model-view matrix* by the *projection matrix* resulting in the 2D representation of the scene on the projection plane [45-48].

2.6 Software

Three main software packages were utilized for the development of this work. They are introduced here, and a brief explanation of each is provided.
2.6.1 3DVIA Virtools

3DVIA Virtools®, a part of Dassault Systemes’ [49], is a game engine platform that allows the user to build interactions, animations and navigations in a virtual environment by using Building Blocks from the Virtools library, by modifying the built-in building blocks or by creating new ones. The Virtools Software Development Kit (SDK) can be used to modify and create building blocks.

Many gaming, animation and VR industries use Virtools to model behaviors in virtual environments. 3D models can be constructed in a 3D modeling software and then imported by using a third party plug-in. After all behaviors and interactions have been modeled, the project can be exported as a web file that runs in most web browsers including Firefox, Internet Explorer and Google Chrome.

2.6.2 Unity3D

Unity or Unity3D® is a game engine developed by Unity Technologies [50]. It is an integrated development tool used for creating interactive 3D software, like games or simulations.

Applications developed in Unity are capable of running many platforms including Windows, Mac, Wii, or iPhone. Coding within Unity is done with the Unity Editor, built on Mono, which is an open source implementation of the .NET Framework. Furthermore, it supports files created in most popular 3D modeling applications like 3DS Max, Maya, Blender, and Cinema 4D. Unity3D is utilized by many developers because of all the features it has such as:
• Rendering: the graphics engine uses Direct3d, OpenGL, OpenGL ES and other APIs and supports bump mapping, reflection mapping, parallax mapping and many more.

• Scripting: Unity allows the user to build a script on Mono using JavaScript, C# and others.

• Asset Store: since 2010, Unity’s users can make use of its Asset Store that provides thousands of 3D models, textures and materials, sound effects, tutorials and more.

2.6.3 MiddleVR

MiddleVR® is a generic immersive virtual reality plugin developed by “i’m in VR” [51]. It adds features to the virtual environment supporting interaction devices such as 3D trackers, active and passive stereoscopy, and clustering.

MiddleVR for Unity is a customized plugin that allows users of Unity3D to use trackers to control the cameras in addition to the ability to build multiple viewports on multiple displays.
CHAPTER 3
RELATED WORK

When approaching the underlying issues of this work, several areas of research are to be investigated to create an understanding of what the researchers and scholars have already accomplished in similar applications.

The mirror display for virtual environments which is the aim of this work is a component of the Nuss Procedure Surgical Simulator (NPSS). Therefore, studies and articles regarding the use of virtual reality in haptic-based surgical simulation might provide some helpful insights.

When running the simulator, the user is to maneuver the surgical tools using the haptic device which provides a touching sensation when in contact with the model in the virtual environment. It would, therefore, be of benefit to examine studies about the use of haptics in combination with displays.

As mirror reflection has been utilized in previous virtual reality applications, describing information from researchers' work in this area will enlighten the development path of this thesis. Particularly beneficial would be those that implement the combination of mirror reflection with haptic devices in virtual environments as that would be a great platform for gathering guidelines and extracting insights. The focus would be on research concerning the issue of co-locating a visuo-haptic display: that is, collocation of a visual graphical scene with a haptic interface.

This chapter introduces an exploration of these areas of research to show the work of some researchers who have come across similar issues addressed by this thesis as well
as some of the solution attempts for approaching the tasks explained earlier. Furthermore, this review of existing literature will help to frame this thesis's contribution.

3.1 Virtual Reality in Haptic-Interactive Surgical Simulation

In 1999, Burdea et al. argued that current training for digital rectal examination for detecting prostate malignancies lacks adequacy and its skills have a very small opportunity for improvement as the patients that the medical students can practice on are very few. Therefore, Burdea et al. developed a virtual reality-based simulator for the diagnosis of four types of prostate cancer [52].

As shown in Figure 15, the simulator is composed of a phantom haptic device to which the tip of the user's index finger is attached. Forces are applied as the user moves within the range of motion that is available. To replicate the procedure, the user places his index finger in a specified hole within a motion restriction board behind which the haptic device is placed.

The 3D model of the human anatomy was purchased from Viewpoint DataLabs as well as a 3D model of a human right hand. In the virtual environment, the 3D models are rendered, and the motion of the index finger is displayed as the trainee moves the haptic arm. The prostate and its malignancies were modeled with a hemisphere and polygon meshes.

Initial human factors test results showed that both medical and nonmedical students were able to touch and diagnose the cases but scored lower results than the testing group that performed the DRE on rubber physical models.
Although their surgical simulation application involves haptic interaction, they did not indicate a solution for the problem of coordinate system differences between the haptic interface and the graphical scene (Task a).

Petersik et al. (2002) [54, 55] have developed a simulator for petrous bone surgery where they introduce haptic collision detection based on multiple contact rather than point contact. This would facilitate drilling into bone which is optimally realized through volume collision.

In their work, they model the petrous bone as voxels (volume elements) which have a size of 0.33 mm$^3$. Through the algorithm that they developed, the sphere shaped tool, which simulates the drill, comes in contact with the voxel-based bone model and collision is detected on multiple points of the tool’s surface generating force feedback through the haptic device according to the magnitude and direction calculated from the interaction between the normal vectors of the points on the tool’s tip and the surface of the bone model. This is illustrated in Figure 16 (left).
As shown in Figure 16 (right), their simulator implements a mirror-based display set up. The user looks into the mirror perceiving the model’s reflection while using the haptic device below the mirror. They also use stereoscopic viewing to enhance realism. Their work is now a commercial product for various surgical and dental simulators [56].

Since their application uses a mirror to display the graphical scene, it can be assumed that they have solved the problem of “Task b” that deals with correcting the mirrored image. However, they did not indicate how they achieved that. In addition, their work did not solve the problem of “Task a” and “Task c” relating to coordinate systems differences and running two separate displays simultaneously, respectively.

![Figure 16. Algorithm for determining drilling force magnitude and direction (left) and setup of for petrous bone surgery simulator (right) [55].](image)

### 3.2 Haptics and Displays in Virtual Reality

Brooks et al. introduced the first haptic display for scientific visualization [57]. Their project was originally initiated in 1967 and aimed to develop a haptic display for
molecular forces. The display was to be used by chemists to investigate molecular docking positions for drugs.

The system’s development life-cycle started with a 2D system (GROPE I) that produced some results and initial explorations. This system was then integrated with E-3 Argonne Remote Manipulators (ARM’s) to produce a system with 3 translation degrees of freedom (GROPE II). However, the available computer at that time wasn’t able to produce real-time forces for sophisticated models, so that development had to hold at that point until a more capable machine was available. In 1986 the computing requirements were fulfilled, and the system was further expanded into a 3-forces/3-torques (6D) system (GROPE III). Experiments were made in each of the three stages involving a different procedure performed by different participants, and results were collected.

One of the primary goals of this project was to investigate the influence of having haptic feedback on enhancing the performance and learning experience of the user. The participants in the study were, therefore, asked to dock a molecular drug into a protein molecule binding site once with force feedback enabled and another time with it disabled. Overall results showed that participants were able to complete the task in much shorter time when the visual display was coupled with haptic feedback.

Their work solves the issue of coordinate system differences between the haptic interface and the graphical scene (Task a) but does not indicate a solution to the other tasks.

Stredney *et al.* (1998) [58] provided an overview of integrating visual displays with haptics. As they were involved in two projects previously to develop a virtual environment for functional endoscopic sinus surgery (FESS), until 1998 their work was
focused merely on developing systems that provide a realistic environment for learning FESS which lacked interactions with surfaces. In 1998, they proposed a method of integrating their methods with a haptic device to allow the user to perform complicated surgical techniques in a simulated environment. Using data sets from the Visible Human Project (male) provided by the National Library of Medicine [59], the regional anatomy was constructed and integrated into a simulation environment developed under Performer and OpenGL. This graphical display of the anatomical model provides haptic feedback using a haptic device by Immersion Corp.

Their work uses interaction with graphical display, but they did not indicate a solution to the problem of coordinate system differences between the haptic interface and the graphical scene (Task a).

3.3 Visuo-Haptic Systems Collocation

When a virtual environment is coupled with the use of a haptic device, a prerequisite for a high fidelity virtual reality system is a seamless alignment between the visual and haptic interfaces. In order to achieve that, a collocation can be implemented by using either of two methods. The first method is to use optical reflection of the visual and haptic displays, and the other one is using video image integration to construct a space where both the graphical as well as the haptic interfaces are present.

Both methods have been implemented by researchers in various applications, but since optical reflection is used in the Nuss Procedure surgical simulator setup, it will be more deeply explored.
3.3.1 Mirror-based Displays and Haptics

Schmandt from the Architecture Machine Group at Massachusetts Institute of Technology was the first to introduce a mirror-based display system as a reach-in workspace in 1983 [60]. In his work, the stereoscopic display is viewed reflected from a half-silvered mirror under which the user (wearing PLZT shutter glasses) can reach and manipulate the objects in the image using a "wand" that implements a magnetic digitizer. Figure 17 shows the work space constructed.

He indicated the need for calculations and transformation matrices to align the tool's coordinate system with the coordinate system of the image reflected off the mirror as well as the need to compensate for the image's disparity in the x/y plane. However, no detailed investigation was made, and this problem wasn't the main component of this work as the virtual object that the user manipulates is merely a paintbrush in space.

![Figure 17. Work Station setup by Schmandt [60].](image)
The team at CSIRO Australia [61] introduced a software kit for modeling complex reflecting systems in 2003. They aimed to develop a system combining haptic interaction, 3D computer graphics and auditory interfaces in order to support the user with tasks such as design and mentoring. They integrated the ARToolKit [62] with the Reachin Core Technology API [9] to result in an interface that can be used for collaborations between face-to-face and remote users. Although Adcock et al. used mirror reflection and combined it with a haptic device and tracker. Their main focus was creating a collaborative-friendly augmented reality environment as well as integrating the toolkit and the tracker to achieve real world video and computer graphics.

In their mirror-based display system, Mulder and the research team in Amsterdam [63] incorporated realistic occlusion effects at points of interactions between physical and virtual objects in a co-location display system in 2005. In this system, as the user looks through the semi-transparent mirror, occlusion is performed either by omitting the rendering (not rendering) parts of the virtual objects governed by the physical objects that occlude the view or by blocking light (light blocking device) and omitting pixels in order to block view of the physical objects when looking through virtual objects that have translucent material rendering. Figure 18 shows the virtual reality scene before (left) and after (right) occlusion effects. In his work, Mulder presented an algorithm to perform this operation as well as introduce a prototype that functions but with poor transparency as well as blurriness issues.
All mentioned research thus far did not tackle the issue of using two displays at the same time with one being a mirror display (Task c). Even though some of these applications use mirror reflection in their setup, the issue of mirror inversion was not addressed.

3.3.2 Optical Reflection Collocation

The Electronic Visualization Laboratory at the University of Illinois at Chicago has developed many virtual reality systems including the CAVE [64], ImmersaDesk [65] and many more. The research team at this facility, Johnson et al., have developed a Personal Augmented Reality Immersive System (PARIS) that was motivated by the need to have virtual reality immersive system where the user’s hands do not occlude the virtual scene but can instead be a part of it [14].

The PARIS® (Figure 19) utilizes a projector to display the graphical image on a semi-transparent mirror and maneuvers the tool using the haptic device underneath it. Since a projector is used, the authors solve the problem of the reversed image by simply flipping the image originally using the configuration options of the projector. This option
is not possible in regular monitors and LCDs, so this solution can only be realized using a projector. Although this method solves the issue of “Task b”, it can only be implemented using projectors. Also, this work does not provide a solution for the other tasks of this thesis regarding mirror reflection correction and multiple display capabilities (Tasks a and c).

Figure 19. The PARIS® [14].

Luciano et al. (2005) [15, 16] developed a system that utilizes optical reflection with incorporation of head and hand tracking mechanism in a stereoscopic haptic-interactive virtual environment. Their system, ImmersiveTouchTM (Figure 20), is now a commercial product that is used for surgical education and surgical planning applications [11].

As they develop their system, Luciano et al. evaluate the existing systems out there that implement optical reflection for visual-haptic collocation. In their evaluation, they explicitly state the problem of horizontal image inversion when using a mirror.
They explain that in the Reachin Display system [9], the reflected image on the mirror is flipped and cannot be used for most of the application. Similarly, the SenseGraphics Display system [10] has the same problem as the image is flipped and will only work for symmetric objects. The Nuss Procedure surgical simulator has a Reachin display setup, so the very same obstacle is faced.

In their system, Luciano et al. resolved this issue by modifying the CRT display hardware components. They reversed the wires of the horizontal deflector yoke of the monitor resulting in a horizontally reversed image which causes its mirror reflection to be in the right orientation. This is a proposed solution for “Task a”, but this work does not have a multiple display setup, so “Task c” was not addressed.

Figure 20. ImmersiveTouch™ Platform [11].

In 2003 Williams et al. of Ohio University initiated a project to develop a virtual reality haptic-interactive model of the human back to be used for training of osteopathic
physicians, massage therapists and medical students in palpation diagnosis [66]. The main motivation of the project was to put together an increased-realism training platform where the trainees were able to distinguish and resolve minor differences and dysfunction locations in tissue and back structure not only visually but also by close-to-realistic touching sensations. The other aim of the project was to examine the influence and effectiveness of haptics in virtual environments by collecting feedback and evaluations from trainees regarding different training aspects and criteria.

As illustrated in Figure 21 (left), a volunteer’s back was measured to generate digital data used to build the virtual model of the back. Geometric shapes representing main skeletal landmarks such as vertebras and ribs were added with colliding properties and combined with the data collected to visualize the human back, and the overall model was then constructed to detect collisions with the virtual fingertips and send spring force feedback upon contact through the Phantom devices (Figure 21 right).

Figure 21. Construction of Virtual back (left) and the Virtual Haptic Back apparatus (right) [66].
In 2004, the team further developed the Virtual Haptic Back model to include more skeletal landmarks, simulate several back dysfunctions as well as generate an image of the vertebral column using a 3D scanner. This is shown in Figure 22. Furthermore, they implemented it as a training tool and reported initial evaluation results to support its validity as a training instrument [67].

Figure 22. improved virtual back model [67].

Before 2006, the Virtual Haptic Back (VHB) project was run using a monitor and two Phantom haptic devices where the displayed image of the back was in two dimensions. In addition, there was a substantial distance between that graphical image and the points in space where the user feels haptic feedback from contact with the model. To add more realism and user immersion, Ji et al. (2006) [13] further developed the viewing experience by adding 3D stereo effects and eliminating this separation between the graphical and the haptic models. They proposed and evaluated two viewing methods for the purpose of visual-haptic display collocation: a Head Mounted Display (HMD),
which will be discussed in the next section, and a Mirror Viewing System (MVS). The latter method is most similar to the work of this thesis.

In the MVS method, the authors used a semitransparent mirror to reflect the graphical model from the monitor while the user reached underneath the mirror to maneuver the haptic devices. The authors here faced the notorious problem of the image being reversed because of the mirror.

They proposed two ways to resolve this issue. The first one is to use two mirrors; the second mirror re-reverses the reflected image from the monitor to place it in the right orientation. This method, however, needs space and results in the (reflected) image being 52 centimeters away from the user’s eyes because of using two mirrors as shown in Figure 23 (left).

The second solution they introduced was to use OpenGL programming to initially reverse the image so that when it is reflected off a single mirror it will be back to its right orientation as shown in Figure 23 (right). This latter approach of originally reversing the image in programming is more suitable to the conditions of the surgical simulator as using two mirrors will be problematic and will result in image distortion and difficulty of usage. The authors, however, do not include any details about their OpenGL programming nor any references with that regard.

After further digging, it was found that Wei Ji (the primary author of this paper) did his master’s thesis about implementing and evaluating these two viewing methods of the Virtual Haptic Back. In his thesis [68], he doesn’t refer to OpenGL programming at all for solving this problem but instead mentions the usage of the software Keystone from NVIDIA for screen adjustments and explains that this software was used to originally flip
the image of the model before being reflected off the mirror. Unfortunately, Keystone software has many limitations; it is not compatible with Windows vista and beyond to mention just one [69].

In the NPPS, the graphical model of the patient needs to be produced for two views, only one of them uses a mirror and the other one (thoracoscopic view) is displayed on a monitor and both views must be of the same model in real-time. Therefore, even if the image was originally flipped it cannot be used for both views. As the author’s application doesn’t have that requirement, their proposed OpenGL programming approach will be further discussed as an attempt to yield a suitable implementation for the surgical simulator.

Figure 23. Virtual Haptic Back setup using two mirrors (left) and using a single mirror (right) [13], © [2006] IEEE.

In a more recent study, Wang et al. (2008) [17] also addressed viewing options for applications combining haptics with virtual reality and the issue of collocation of both interfaces to increase the “feel” aspect of the user as well as enhance the fidelity of the
virtual environment. As optical reflection was one of the methods used to achieve that, they used a mirror-based display set up for their research.

In their work, they emphasize the need for consistency between the haptic and visual display and explain that the literature lacks methods to quantitatively measure registration accuracy. They define registration accuracy as the distance between the point on the virtual avatar and that on the haptic interface according to the human’s eye coordinate system. The higher the registration accuracy, the more realism is achieved and the more successful the simulation is.

They proposed a mathematical quantification of the registration accuracy and validate it by evaluating the performance of a prototype that they set up. In order to achieve that, they used a high precision measuring apparatus, i.e., FARO arm [70], to compare the accuracy error between haptic and visual displays. Visual consistency experiments were used to evaluate registration accuracy. An object is constructed with both physical and virtual components and is used to produce quantitative validation results.

In 2011, Wang et al. [18] presented further development of the method. They defined spatial collocation accuracy as the distance between the seen tip positions of the virtual and the real tools.

A measurement approach that specifies visuo-haptic spatial collocation error was constructed. They also introduced spatial calibration methods and accuracy verification techniques. Minimizing the spatial collocation error was achieved by deriving a mathematical model based on a signal transformation analysis from which calibration was performed.
The set up that they experimented on showed an increased spatial collocation between the haptic and visual display, and the measurements taken of the final position of the virtual tool after moving the real tool indicated a very small error.

3.3.3 Collocation via Video Image Integration

Yokokohji et al. (1996) [71] were the first to introduce the concept of a What You See Is What You Feel (WYSIWYF) display which integrates visual and haptic interfaces. Their approach utilizes chroma key compositing to produce a live video image of the user's hand motion. This video image is then blended into the user's virtual scene using a head mounted display. Their prototype in 1996 allows the user to simply manipulate a virtual cube with high accuracy.

Later in the same year, they added a fast tracker to improve the frame rate and implemented the prototype for a skill training experiment [41, 72, 73].

Sandor et al. [74] (2007) introduced an experiment to compare using half-mirrors for optical reflection based collocation with an HMD system. In their work, they indicated the advantages of using a HMD in a simple task of target location. In addition, they indicated that using half-mirrors reduces performance of the user/operator. They also introduced an interactive painting application using a HMD.

After performing the experiment of implementing a target locating task using a HMD in one setup and a mirror-based display in another, they concluded that the task completion time with the HMD set up was faster. Figure 24 shows the experiment implemented once using a HMD (left) and another using a mirror-based display (right). They explained the long duration in task completion for the mirror-based display setup
was because of the lack of a head tracking mechanism, difficulty of perceiving the scene’s background caused by the limited level of transparency that the mirror provides, and lastly because the user’s hand was not completely visible.

Figure 24. Target location task using a HMD (left) and using a semi-transparent mirror (right) [74], © [2007] IEEE.

In their Virtual Haptic Back project, Ji et al. [13] evaluated the use of another method for visuo-haptic collocation which used the Head Mounted Display to integrate the real and the virtual scenes.

In their HMD method, they used a head motion tracker to change the display on the HMD according to the head’s motion. The tracker’s sensor was placed on the head (affected by the head’s movement) and the transmitter was placed (fixed) at some surface. However, combining the tracker with the HMD and the model of the back is not straightforward as there are coordinate system differences. The system’s coordinate systems are those of the world, user’s eyes, the graphical model of the back, and the transmitter and sensor. The sensor placed on the user’s head has the same coordinate system change as the user’s eyes as they move synchronously. The authors here
mentioned that conventional coordinate transformation methods can be used to align the coordinate system of the user’s eyes with that of the graphical model (Figure 25).

This problem is similar to the problem faced in the development of the NPSS where the coordinate systems of the world, the haptic device, the graphical model’s thoracoscopic view on the monitor and the graphical model’s external view displayed on the mirror are all to be aligned in real-time. The authors, however, did not provide detailed calculations as to how that was performed and how they addressed the issue of aligning more than two coordinate systems. Furthermore, the authors did not have the problem of correcting two different graphical display systems with reflection in real-time (Task c).

Figure 25. Virtual Haptic Back implementing HMD [68].
The work presented in this thesis introduces a methodology for developing and executing a multi-configuration display component that can be utilized for virtual environment applications. The term multi-configuration refers to the ability of this display to view the scene in a mirror-reflected setup as well as a normal setup or both at the same time. This function of the display takes into account Human Computer Interfaces utilizing haptic feedback while seamlessly updating the views.

In this chapter, methods to achieve the research goals are presented as well as the approaches and techniques that were undertaken to develop a graphical-haptic display that uses mirror reflection collocation and displays the corrected mirrored graphical scene in real-time.

4.1 Overview

This chapter provides at the outset a general outline and description of the design developed for a multi-configuration display component. Description of the software packages utilized and implemented plug-ins is then provided. The remainder of the chapter will discuss the development methods for achieving the three main goals in this work.

The first goal addresses the issue of coordinate system differences between the different interfaces used. The user's physical world, the graphical scene displayed on the monitor and the graphical scene displayed on the mirror, as well as the haptic device all
have a different coordinate system. Aligning the different coordinate systems is therefore an essential step toward achieving full functionality of the surgical simulator.

The second goal addresses the use of a mirror as one of the viewports of the surgical simulators. This necessitates optimizing a method for correction of the mirrored image of the patient's external view while displaying a separate un-mirrored of the patient’s internal view on a different monitor. Four different yet related methods were developed and implemented to achieve mirror correction. However, some of them lack efficiency and model preservation requirements. Ultimately, an optimum method was developed that addresses all requirements and can be considered the solution for such application. As the four methods differ in their outcomes, lastly presented is a comparison between the four where they are evaluated against relevant criteria to show the attributes missing in some and to describe the advantages of others.

The third and last goal addressed is the ability to run multiple displays at the same time each as a separate viewport with independent configuration.

4.2 Design

A virtual environment application can be composed of multiple components that are all integrated to provide the correct interface for the application. In addition to the virtual environment scene that is constructed and displayed on a monitor, the components might include input devices to build interactions for the virtual scene such as a joystick or a tracker. If components are to be included that might alter properties of the image or interface, a configuration component will be needed. Figure 26 shows a schematic of a
simple yet typical virtual environment structure and illustrates how a configuration component will be integrated in the setup produce satisfying behavior.

![Typical Virtual Environment structure with configuration component.](image)

In this work, a generic configuration component is designed to be used in such applications to provide the capability to configure the virtual environment to produce consistent and correct behavior. This component takes into account the different types of display methods such as monitor, projector or mirror-based display. It also takes into account the use of a haptic device that is rendered with translational and rotational behavior. The two configuration paradigms here are the image reflection, in case a mirror was used, as well as the alignment of different coordinate systems for the haptic device and the used display(s).

After developing and implementing solutions for the three tasks of this work mentioned earlier, the solutions will be integrated in this component which in turn will be coupled with the virtual environment to implement those solutions as appropriate. The
component will, therefore, provide the user with a customized configuration capability to allow for mirrored-image correction and coordinate systems alignment.

A virtual environment application, a surgical simulator for instance, might be composed of the virtual environment, an Omni haptic and one viewport for display by using a mirror-based projector display. The designed component will be integrated into the structure of the application to account for the used components. As shown in Figure 27, the Display Architecture will be composed of one viewport using a projector and the image will be flipped in the direction of reflection for the scene, the X-direction for instance. For the Haptic Alignment configuration, the Omni haptic device might require an inversion of the Y-rotation only for instance.

Figure 27. Schematic of a simulator that uses mirror-based projector display.
4.3 Software

The software packages and modeling tools that were involved in this work are briefly described here. Initially, the development platform of the simulator was 3DVIA Virtools. Then, with the expansion of the model and the need for more enhanced features, a more capable platform was needed which was the reason for migrating to Unity3D.

4.3.1 3DVIA Virtools

The majority of the development of this work was done using 3DVIA Virtools 5.0 software. It has been the development platform for the Nuss Procedure surgical simulator from the start, so parts of this work were done there. Several objectives were accomplished in Virtools. A significant objective addressed was the problem of Cartesian System difference. This issue was investigated and resolved using the Virtools Platform. Subsequently, the development of the surgical simulator was entirely relocated making Unity3D the new platform.

4.3.2 Unity3D

As the need for a more capable and supported software arose, Unity3D game engine was adopted and is currently the primary development platform for the Nuss Procedure surgical simulator. Unity3D is a high performance development environment with limitless capabilities. The 3D objects of the patient’s torso and the surgical tool were imported, and required behaviors were modeled. Unity3D was utilized for the major objective of this work involving the development of four different methods for mirror inversion correction.
4.3.3 MiddleVR for Unity

As mentioned before, when the Nuss Procedure surgery is performed, the patient’s thorax is viewed using a thoracoscope displaying the patient's interior on a monitor in addition to viewing the body externally. To realize this aspect in the simulator, the two separate cameras in the scene should each display on a separate monitor.

MiddleVR for Unity is a commercial plugin that performs this operation. This plugin was purchased and was utilized in this work to achieve the two views required.

4.4 Environment and Components

The setup of the surgical simulator is composed of the virtual environment and the physical setup. The physical setup includes the Reachin platform (LCD, semi-transparent mirror and haptic device) as well as another separate monitor. The virtual environment is where all the interactions are constructed and it contains the 3D objects of the patient’s body, internal organs and the surgical tool. In the external view, the user sees only the patient’s torso and surgical tool whereas in the thoracoscopic view the user sees the ribcage, thoracic cavity, lungs, diaphragm, the pericardial sac as well as the surgical tool.

In the Nuss Procedure, to avoid puncturing or damaging the heart, the surgeon makes an incision on the patient’s right side of the chest to create a pathway for introducing the steel bar. Therefore, as was requested by the surgeons, it is essential that the 3D model of the patient in the surgical simulator is in that specific position with his/her right side facing the surgeon. When creating the virtual environment, it is not difficult to position the camera in a way to display the correct view of the patient’s chest.
on the monitor, but when considering the view through the mirror reflection, the image is falsely altered.

4.4.1 Adapted Coordinate System

An important step for targeting the issues of this work is to have a standard coordinate system to refer to in order to construct a methodology to evaluate the different systems involved in the simulator setup.

A right-handed coordinate system is a 3-dimensional coordinate system that satisfies the right-hand rule. The right-hand rule is a conventional rule that specifies the positive directions of the axes by assigning the positive X-axis and positive Y-axis to the right hand thumb and index finger respectively creating a plane, and then the positive Z-axis will be in a direction perpendicular to that plane assigned to the middle finger. The left-handed coordinate system works the same only using the left hand. Figure 28 demonstrates how the right hand rule is used to specify the positive direction of the axes in a coordinate system. For either of the two conventions and by merely looking at the coordinate system from different angles, one can specify which axes point to the right direction and which one points to the up direction depending on the angle of view. That does not change the coordinate system configuration, however.

This work uses the left-handed coordinate system. Any system with different coordinate systems throughout this work will be defined and compared to that when investigating this aspect.
4.4.2 Virtual Environment Display Setup

As mentioned before, the surgical simulator is to be capable of supporting two different displays, one on the monitor and another when using the mirror. Throughout this work, the term “Thoracoscopic Display” will refer to graphical scene provided by the camera positioned to project the view of the patient’s thorax and internal organs and display it on a small monitor. Similarly, the term “Mirror Display” will refer to the graphical scene provided by the camera positioned to display the view of the patient’s torso from the outside on an LCD to be reflected off the mirror.

To be observed correctly, the 3D model of the torso was positioned horizontally facing upwards; aligned with the X-axis with the patient’s head pointing to the negative X-direction. Figure 29 shows the point in space where the 3D model was placed with respect to the positive directions of the coordinate system.
To provide the Thoracoscopic Display, a camera was positioned in space inside the torso’s right side and was rotated downward to show the exact view that the surgeon experiences when inserting the thoracoscope while performing the procedure as shown in Figure 30.

These two cameras will be the primary ones that provide the two different displays and will be the part of the simulator on which this work is based. Throughout
this work, any configurations or changes to the two displays will be a result of manipulating these two cameras.

4.5 Visuo-Haptic Coordinate Systems Configuration

When integrating different systems with different Cartesian coordinate systems, a process of alignment has to take place. This is the case when a virtual environment and a haptic device are coupled. In order for the motion to be consistent, the two coordinate systems need to be transformed into a common one. When adding a mirror to the setup, an added step is to be considered as mirror reflection creates a different coordinate system by inverting one of the axes.

This issue can be resolved using transformation matrices. An investigation was conducted to develop two standardized sets of matrices that account for the configuration difference between the haptic device world space and the graphical scene world space of Virtools in one case and between the haptic device world space and the world space of the graphical scene’s reflection on the mirror in another.

4.5.1 Investigation of Coordinate Systems

As mentioned before, the left-handed coordinate system is implemented in this work with the positive X-axis pointing to the right, positive Y-axis pointing up and the positive Z-axis pointing away from the user. When examining the other systems, it was found that Virtools also uses a left-handed coordinate system with the positive X-axis pointing to the right, positive Y-axis pointing up and the positive Z-axis pointing away from the user into the screen. In addition, it was found that Unity3D also uses a left-
handed coordinate system similar to Virtools. Figure 31 left and right show the coordinate systems of Virtools and Unity3D.

![Figure 31. Coordinate system of Virtools (left) and of Unity3D (right).](image)

However, when studying the PHANTOM Haptic device, it was found that it uses a right-handed coordinate system with the positive X-axis pointing to the right parallel to the front plate, positive Y-axis pointing up and the positive Z-axis pointing toward the user and away from the device as shown in Figure 32.

![Figure 31. Coordinate system of Virtools (left) and of Unity3D (right).](image)

When building interactions in Virtools for the haptic device, the motion of the haptic device is computed in the form of position and quaternions parameters. The x-y-z components of the position parameter are updated as the tool moves in space in three dimensions whereas the x-y-z components of the quaternion parameter are updated as the device reads pitch, yaw or roll rotations from the user motion.
4.5.2 Comparison of Translations

It was essential to identify and confirm the configuration of the haptic device's coordinate system and how it differs from the Virtools coordinate system. Additionally, it should also be compared to the world space of the graphical scene's reflection in the mirror.

Before developing the configuration for the conversion matrices, the haptic device's source code was studied and adjusted. In the device's code, the position and quaternion configurations were reset to their original state and the code was updated to make sure that its motion is consistent with the world space given in its manual as Figure 32 shows.

When considering the Thoracoscopic Display, merely placing the 3D model so that the patient's head is pointing in the negative X-direction assures consistent motion with the haptic device. This way, when comparing the haptic device's coordinate system with Virtools's coordinate system on the monitor an alignment is found; thus, when the haptic device's stylus moves horizontally to the right, the surgical tool will move to the
right also, whereas it would have been inverted if the 3D model was positioned in a different way.

However, when considering the Mirror Display this is not the case. As the haptic device’s coordinate system is compared with the coordinate system of the scene’s image in the mirror (Figure 33), it is observed that the positive Z direction is inverted. This means that, when the haptic device’s stylus moves forward towards the patient’s body, the surgical tool in the graphical scene displayed in the mirror will move outward towards the surgeon. A construction of conversion matrices is needed for Virtools to control this behavior converting the configuration of the Cartesian coordinate system using transformation matrices for the tool’s position configurations.

As indicated before, the first parameter of the Phantom haptic device is the position vector of the tool. This takes the form of an X-Y-Z vector which determines the three dimensional linear motion of the tool in space. In order to correct the tool’s motion, the X-Y-Z components are to be multiplied by a matrix consisting of three elements which can be either 1 or -1 according to the investigation made earlier. Thus, for the position configuration, the transformation matrices should perform the following:

- For the Thoracoscopic Display, the matrices multiplication should not invert any of the axes.
- For the “Mirror Display”, the matrix multiplication should invert the Z-axis.
4.5.3 Comparison of Rotations

To be able to configure the rotation parameter of the tool, an investigation was done by running the device and moving the 3 rotations (yaw, pitch and roll) one at a time to assign each one to its correct component among the first three components of the quaternion (X, Y and Z) in Virtools. The investigation showed that the Yaw rotation of the tool is controlled by the Y quaternion component, the Pitch rotation is controlled by the X quaternion component and the Roll rotation of the tool is controlled by the Z quaternion component (Figure 34). The investigation also showed that the yaw rotation (Y-component) was inverted in the Monitor View and that the Roll rotation (Z-component) was inverted in the Mirror Display. The correction model needs to account for these configuration differences in the quaternions as well.

After completing the investigations and determining the underlying gaps, the configuration matrices were developed.
As discussed before, the second parameter of the Phantom haptic device motion configuration is the rotational parameter. This parameter is in a form of a quaternion. Each of the first three components of this quaternion (the X, Y and Z components) controls a specific rotation of the tool. With the use of the investigation made earlier, determining the conversion matrix was possible. The transformation matrices for the quaternion configuration should perform the following:

- For the Thoracoscopic Display, the matrices multiplication should invert the Y-component (Yaw rotation) only.
- For the Mirror Display, the matrices multiplication should invert the Z-component (Roll rotation) only.

4.6 Visuo-Haptic Graphical Configuration

After determining the difference in world space between the haptic device and Virtools and constructing the required transformation matrices for alignment, the next and most essential step in achieving correct visuo-haptic collocation is to resolve the display error caused mirror reflectance.
As mentioned previously, assigning the correct position and orientation of the thoracoscopic camera is trivial, but using the same camera position and orientation for the Mirror Display will not provide the right view as the image will be flipped.

In this section, four attempted methods for resolving this issue are introduced. After all simulator-related development activities were moved to Unity3D, further modifications to the visuo-haptic display, including the development of these four methods, were accomplished using Unity3D.

Four different methods were developed because each one had some limitations, and the initial ones were found to satisfy only some of the required features but not all. Ultimately, all needed functionality was achieved, and all requirements were met.

The primary properties of the Mirror Display are derived from the customized needs of the Nuss Procedure surgical simulator. This display is to provide consistent visuo-haptic collocation by displaying the graphical scene reflected in the right orientation. This display also has to accommodate another view that is integrated to the same scene in real-time with no reflection and is displayed on a separate monitor.

4.6.1 Method 1: Manipulation of Camera Position and Orientation

Initially, to observe the existing error, the camera was positioned in the desired position and was initiated to project for the Mirror Display. As the mirror was observed, the patient’s body appeared hanging horizontally in the air facing downwards instead of lying down and facing upward. This view of the image will be referred to as the Baseline Reflectance error (Figure 35 right).
Therefore, the second attempted approach was to position the Mirror Display camera in the same point in space but to rotate it 180 degrees or, in other words, inverting the up direction of the camera. This showed the patient’s body in the right position lying down and facing upward, but caused the body to be inverted sideways causing the patient’s head to be on the right side and the chest’s left side to face the spectator or, in this case, the surgeon. This will be referred to as the Horizontal Inversion error (Figure 36 right). It was obvious that merely rotating the camera did not suffice in providing the correct view.

A very similar and helpful analogy is an emergency vehicle. The word "Ambulance" is written on the front of the vehicle from right to left and the letters are inverted horizontally. A driver in another vehicle, however, will perceive the word’s reflection in his/her rear view mirror exactly in the correct orientation and the right order. This basically implies that when the word “Ambulance” was originally written correctly on the emergency vehicle it will be perceived inverted sideways. However, when the word is originally written exactly the same way it appears in the error (inverted
sideways), the result will be the correct word reflected off the mirror. Therefore, the goal is to position a camera in a way to make the 3D model appear originally just like the Baseline Reflectance error (Figure 35 right) and fix the camera in that position.

Figure 36. View of the patient’s body in the scene after rotating the camera (left) and its corresponding image in the mirror i.e. “Horizontal Inversion” error (right).

To achieve that view, the camera had to be placed on the other side of the patient’s body (the left side) and then had to be rotated 180 degrees. This camera position showed the body originally hanging in the air horizontally and facing downward (Baseline Reflectance error).

This method has a limitation. Although the user perceives the patient in the correct position with the chest's side facing the user, it is actually the left side of the patient's chest. The placement of the camera in that position projects the left side of the avatar's torso to be the right side. Therefore, even though it may look correct, this method can be used only for applications with 3D objects that have symmetric right and left sides, so this error will not be of relevance.
4.6.2 Method 2: 3D Object Polygon Inversion

As mentioned in the previous section, the method of Camera Position and Orientation will only work for symmetric-sided models. If the application includes interactions with the object, then this method will produce an error where the user will think that he/she is interacting with the right side of the object whereas in fact it is the left side.

Since the application of interest is surgery simulation and it involves haptic interaction and model deformation, that method does not solve the problem. The application, therefore, requires that the right side is physically the one facing the user and not just displayed so, especially since the patient's body is obviously asymmetric.

After specifying this setback, another approach was attempted. The same concept is still carried on, to create a view that imitates the Baseline Reflectance error in the original view so that the model will look correct in the mirror. However, the limitation of the previous method is that the user was facing only what seems to be the right side while it is actually the left one. Thus to try to solve that, a practical idea would be to flip the model itself so that the user will in fact face the right side of the patient's body. To do so, the scale of the patient's 3D model and all associated organs is inverted in the direction of reflection by flipping its axis. This will cause an inversion of all vertices in that direction causing the right and left sides to be swapped.

To better explain the approach, the letter $R$ was placed on the right side of the original model as a part of its texture to show how the polygons move from one side to another. Figure 37 (left) shows how the letter appears on the torso’s right side in the
scene. The scale of the body was then inverted in the Y-direction so that all the vertices on each side will be swapped as in Figure 37 (right).

Figure 37. Letter R as a part of the texture on the right side of the body (left) and the way it became after flipping its polygons in the Y-direction (right).

After doing that, the camera was situated in a way to provide the Baseline Reflectance error just like in the first method. This provided the Mirror View desired.

This method solves the problem created by the previous method. However, it has another problem of its own. Inverting the polygons that compose the model will cause every part to switch from left to right. For instance, the left lung will appear in the right side of the body. As a thoracoscopic view is also involved, this method does not provide a sufficient solution for the application of the surgical simulator nor for many other applications. Another method will, therefore, have to be established that preserves all properties of the model itself but only influences the viewport.
4.6.3 Method 3: Render to Texture and Plane Inversion

In the previous section, the solution explained affects the 3D model and inverts the polygons that compose the objects in the axis of mirror inversion so that the right and left sides of the patient's body will be swapped.

This, however, cannot be used in the application of the surgical simulator as the internal organs and components built, the pericardial sac for instance, have to stay intact and have to be in their correct places anatomically. The optimal solution would be one that does not impact the location of the 3D models as this would significantly reduce realism.

The idea that this third method introduces is utilizing a render to texture approach. Render to Texture is a technique for creating a specific type of dynamic texture that is updated at runtime. The idea is to render a scene just as usual, but instead of displaying it on the screen, it is converted into a 2D texture to become an entity that can be used in the scene itself. Associated applications for this technique include in-game cameras and post-processing. In OpenGL, this can be done by rendering the scene to a Frame-buffer and then creating the texture which will contain the RGB output of the rendered scene. In other words, using this technique enables the user to read the pixels from the camera and create a 2D texture out of them.

In the application of the simulator, this is done by placing the camera in its correct position and orientation to view the patient in the right way and then convert this display that this camera renders into a 2D texture. A thin 3D plane will then be created, and it will be textured with that 2D texture created from the camera. When the simulation is
running, this plane's texture will be a real-time view of the patient's right side provided by that camera.

Using Unity Pro, a render to texture capability is directly provided. After creating the camera, a Render Texture can be created from the Create menu. In the camera's properties, one of the camera's parameters is called Target. The created Render Texture is assigned as the target for that parameter. A material is then created, and the material's component will be assigned as the created Render Texture. Then, the created material will be added to the object that is to display the rendered scene which, in the case of the surgical simulator, is the thin plane that was created in the scene. Figure 38 shows the camera positioned to view the patient's body and how its view is rendered as a texture of the plane created.

![Figure 38. Camera's scene rendered as a texture to a plane.](image)

After all that is prepared, the same concept introduced in the first method is still carried on. The Baseline Reflectance error has to be created in the original view so that
the model will look correct in the mirror. To achieve that, the plane's scale is inverted in
the direction of reflection by flipping its Y-axis to create the Baseline Reflectance error.
An additional camera is created in the scene and positioned to provide an orthogonal
view of the flipped plane. This camera will project that view onto the mirror resulting
with the desired output for the Mirror Display. In this method, the render to texture
feature in Unity3D actually reads the pixels from the camera and assigns them to a target
texture map not affecting the model in anyway.

4.6.4 Method 4: Customized Projection of the Graphical Scene

Although the previous method solves the issue of using optical reflection and
corrects the reflected image through rendering the camera view to a texture, it can be
further optimized. In that method, as the pixels are read from the correctly-positioned
camera, they are used as a texture and another separate camera displays the view of that
plane. This, however, involves an extra, unnecessary step.

This extra step was found to add additional computation time since the Mirror
View will be displaying a window of another camera’s view. In addition, it is found that
this extra step reduces clarity and resolution since this view is displaying an orthogonal
view looking at a perspective view. However, a better way can be yet implemented to
reduce the complexity and steps required.

4.6.4.1 Projection Matrix Configuration

In the graphics pipeline, after the geometric primitives are modeled and
transformed into a common world coordinate frame (world space) and illumination has
been set up, viewing transformation takes place. In viewing transformation, the objects are transformed from the world coordinate system to the view reference (camera) coordinate system. The product of these rendering stages is called the Model-View Matrix. After that, a clipping operation takes place to define the viewing frustum and primitives outside this frustum are not rendered.

The next stage is to project the view onto the projection plane. This stage transforms the points from the 3D coordinate system of the camera into the 2D coordinate system of the image space of the screen. The projection transformation is performed by multiplying the Model-View Matrix by the Projection Matrix ($P$).

In the case of the surgical simulator, the required view is obtained by positioning the camera above the patient’s body to the right. As mentioned in method one, using that position of the camera results in the Baseline Reflectance error which is having the body flipped in the Y-axis, when the projection plane is observed. Inspired by this, the problem can be solved by constructing a customized projection operation to control the direction of displaying when transforming from the camera coordinate system to the coordinate system of the screen.

Going back to the graphics pipeline, this projection is performed by multiplying the Model-View Matrix by the Projection Matrix. So building a customized Projection Matrix will grant access to manipulating the projection as desired.

Since the mirror reverses the view in the Y-direction, the Y-component of the Projection Matrix will be the target for this customization. To obtain a new customized Projection Matrix ($\bar{P}$), the original Projection Matrix ($P$) will be multiplied by a 4x4 matrix that has a negative Y-component as follows:
4.6.4.2 Back-facing Inversion

When this new customized projection matrix is implemented in the scene, another reverse-related problem arises. This will be explained in this section as well as how it was resolved.

In OpenGL, any portions of an object that are outside of the viewing frustum are removed from the scene. In addition, all polygons that are occluded by other oblique polygons are not rendered. Similarly, back-facing polygons are not visible in the scene and only front-facing polygons are visible. By convention, all polygons whose vertices appear on the screen in a counterclockwise order are called front-facing. In a simple 2D environment, Figure 39 shows an image of a triangle $ABC$ and its mirrored image in the Y-direction. In graphics convention, triangle $ABC$ is considered front-facing as the vertices $ABC$ appear in a counterclockwise order. However, when it is reflected in the Y direction, $ABC$ becomes a clockwise order of the vertices and therefore the triangle $ABC$ will become a back-facing polygon. When rendering such a polygon, it will not be visible, and instead, all the scene components that this polygon was blocking will come to view.
This is what occurred when using the new customized Projection Matrix for the Mirror Display as it now projects all the polygons as back-facing and therefore not visible. An operation, therefore, was needed in order to make this method work.

In OpenGL, the rule of choosing front-facing polygons can be changed. Using the function `glFrontFace()`, the user can specify the front-facing polygons are determined by assigning the mode to be counterclockwise (`GL_CCW`) or clockwise (`GL_CW`). Also, the command `glCullFace()` can be used to specify which polygons should be discarded. The user can specify the mode to be `GL_FRONT`, `GL_BACK` or `GL_FRONT_AND_BACK` to make culling affect front-facing, back-facing, or all polygons respectively.

This concept was implemented in the application of the surgical simulator, and the problem of the back-facing polygons was resolved using these commands.

### 4.7 MiddleVR Implementation

One of the requirements for the Nuss Procedure surgical simulator is to have two separate displays to display both the thoracoscopic scene on one monitor as well as the
external scene on the mirror. MiddleVR for Unity is a plugin that adds many capabilities to the engine. Of its many features, two are particularly of interest to this application. The first one is that it allows using multiple Viewports by assigning each camera in the graphical scene to a separate display and the second one is that it enables many input devices such as joysticks, trackers and gamepads to be used to move the cameras around. This software was acquired and utilized to fulfill these purposes.

4.8 Code Implementation

Method 4 for visuo-haptic graphical collocation is implemented by configuring a customized projection matrix for displaying the scene in the Mirror Display. To implement this solution, C# code was written in which the projection matrix used for the camera assigned for the Mirror Display is called. This projection matrix was then multiplied by a predefined 4x4 matrix that inverts the Y-component of any 4x4 matrix to result in the new Customized Projection Matrix ($\tilde{P}$) explained earlier. The resulting image is reversed in the Y-direction as in the Baseline Reflectance error causing the view in the mirror to be error-free. Then, to solve the back-facing polygon issue, commands were implemented to change the rule for determining the front-facing polygons for that specific camera. This C# is included below.

```csharp
using UnityEngine;
using System.Collections;

public class MirrorDisplayProj : MonoBehaviour
{
    Matrix4x4 AdjMatrix = Matrix4x4.Scale(new Vector3(1, -1, 1));

    void OnPreCull() {
        camera.ResetWorldToCameraMatrix();
        camera.ResetProjectionMatrix();
        camera.projectionMatrix = camera.projectionMatrix * AdjMatrix;
    }

    void OnPreRender() {
```
GL.SetRevertBackfacing(true);
}

void OnPostRender() {
    GL.SetRevertBackfacing(false);
}

4.8.1 Requirements

For MiddleVR to operate, some requirements have to be considered. The following is a brief explanation of the main requirements.

4.8.1.1 Operating System

MiddleVR requires Windows XP, Vista or 7, 32 or 64 bits with the latest Service Packs and DirectX version. If Windows XP is used, the Microsoft .NET Framework 3.5 [76] has to be installed as well as the Windows Imaging Component [77, 78].

4.8.1.2 Devices

MiddleVR has some pre-specified devices with configured parameters. If any of them is used in the application, some requirements are also needed. The following is an explanation of such devices and their requirements.

- If the Razer Hydra tracker and joysticks is used, the official Razer Hydra has to be installed [79].
- If the Kinect trackers are used, the official Microsoft Kinect SDK 1.0 has to be installed [80].
- If the Motion Analysis trackers are used, Cortex has to be installed [81].
4.8.1.3 Unity3D

Unity3D 3.4 or 3.5 has to be installed to run MiddleVR and Unity3D Pro has to be used if OpenGL quad-buffer is to be used for active stereoscopy.

4.8.2 Configuration

After MiddleVR has been installed and all drivers have been properly downloaded. A simple scene with MiddleVR interactions can be created. For simplicity, this section will provide an explanation on how to configure the mouse as a tracker for the camera and any other input device can be configured the same way. For the case of the surgical simulator, a Wiimote was used as the input device to control the camera’s behavior to simulate a thoracoscope.

After installing MiddleVR and importing the license, the configuration window can be initiated. To create a mouse tracker the following steps can be used (Figure 40):

1. Go into the Devices window;
2. Press the + button to add a device;
3. In the 3D Trackers section, select “Tracker Simulator – Mouse”.


Now that the mouse tracker has been created, a camera in the scene can be assigned to it. This can be done by going to the 3D Nodes window (Figure 41). In this window, the HeadNode can be selected from the hierarchy to display its properties. 3D nodes here represent real world objects in the scene and by configuring them, the user specifies where the cameras are located in the scene and how are they viewing the virtual environment. Other nodes can be added as required.

In the properties, the Tracker parameter can be changed from “Undefined” to “MouseTracker”. By doing so, this HeadNode will be moved according to the 3D tracker’s motion. When this configuration is saved, a configuration file with format .vrx is created and can be used in Unity3D later on.

Predefined buttons in the tracker device (mouse here) can perform specific functions. For the mouse, moving the mouse while pressing the middle button or the Ctrl key on the keyboard moves the HeadNode, translation in all directions can be done by
adding the Alt key and the tracker’s values can be reset by pressing both the left and right mouse buttons at the same time. Other input devices can be configured to perform these functions as desired.

Figure 41. Configuring the tracker created in MiddleVR – *3D Nodes* window.

4.8.3 Importing to Unity Scene

After configuring MiddleVR, a scene can be created in Unity3D, and the MiddleVR package can be imported into the project by going to the dropdown menu “Assets” and following: Assets > Import package > Custom package. After the package has been imported, the VRManager prefab is to be dragged into the scene. In the VRManager properties, the Config File parameter is to be set to the full path of the .vrx configuration file created earlier. When the scene is played now, the camera can be moved around using the mouse.
This provides a simple configuration of the plugin and how it is set up. The user can go back to the configuration tool and perform other changes such as editing the hierarchy, adding cameras or modifying the viewports as explained in the next section.

4.8.4 Viewports and Displays

The other feature in MiddleVR that was implemented in the surgical simulator is using multiple viewports. A viewport in MiddleVR is simply the behavior of the scene’s cameras with respect to the used displays. The layout of these cameras can be configured using the MiddleVR setup in the Viewports window (Figure 42). To do so, simply following these steps will suffice:

1. In MiddleVR configuration window, go to Viewports window.

2. Add the desired number of viewports by clicking on “+ New”. In the case of the surgical simulator, two viewports are required and created.

3. In the same time, the user can choose which display (hardware) is associated with which viewport from the System Displays menu. In the case of the surgical simulator, the two displays that appear are those for the computer monitor to support the Thoracoscopic Display and the LCD supporting the Mirror Display. The viewports appear in the configuration space as red rectangles while the available displays appear as blue rectangles. The viewports and displays can be manually aligned as desired here.

4. By clicking on one of the viewports, the user can specify which camera is to be associated with that viewport to project on the assigned display by changing the Camera parameter in the viewport properties. Doing so makes the MiddleVR
Manager apply these changes to the available scene in order for the hardware (displays) to be coupled with the required camera in the assigned viewport.

Figure 42. Configuring displays and viewports in MiddleVR – Viewports window.

For any further configuration issues or to explore more capabilities and options provided by MiddleVR, one can refer to the MiddleVR User Guide [82].

4.9 Comparison Design

The four methods developed for visuo-haptic collocation via optical reflection are dissimilar in many ways. They all can achieve that goal in some way but depending on the applications, some of them are insufficient.

This section provides the design of a platform for a comparison between the four methods and evaluates each one against relevant criteria once constructed within Nuss Procedure surgical simulator. First presented is a description of the environment in which these methods are implemented and assessed to provide a general idea of the complexity
of the models involved as well as the application’s components associated with the required collocation. Then each of the essential criteria is introduced in its own section for a brief description of its associated gauging techniques.

4.9.1 Environment for Testing

The methods developed will be evaluated in the current virtual environment developed for the Nuss Procedure surgical simulator. The virtual environment is modeled in Unity3D (version 3.5.6f4) within which the different 3D objects were constructed. These objects include the surgical tool (Introducer) as well as the patient’s avatar and internal organs.

The patient’s avatar was anatomically modeled to accurately simulate the Pectus Excavatum condition. This complex model of the human body includes the torso, ribcage, thoracic cavity, diaphragm, lungs and pericardial sac adding up to 92,440 vertices and 134,525 polygons altogether. These objects are also textured with high resolution texture- and bump-maps. The surgical tool is composed of 28,086 vertices and 9,362 polygons. Table 2 provides a more detailed count of vertices and polygons of all the models. In addition to that, real-time physics-based modeling is implemented to provide necessary motions such as a beating heart and similar interactions.

Table 2. Vertices and Polygon count for the 3D model components.

<table>
<thead>
<tr>
<th>Part</th>
<th>Vertices</th>
<th>Polygons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Surgical Tool (Introducer)</td>
<td>28086</td>
<td>9362</td>
</tr>
<tr>
<td>Body</td>
<td>28180</td>
<td>56272</td>
</tr>
<tr>
<td>Ribcage</td>
<td>14884</td>
<td>27776</td>
</tr>
<tr>
<td>Thoracic Cavity</td>
<td>29336</td>
<td>58520</td>
</tr>
<tr>
<td>Diaphragm</td>
<td>4178</td>
<td>8356</td>
</tr>
<tr>
<td>Lungs</td>
<td>13005</td>
<td>25990</td>
</tr>
<tr>
<td>Pericardial Sac</td>
<td>2330</td>
<td>3289</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>119999</strong></td>
<td><strong>189565</strong></td>
</tr>
</tbody>
</table>
The simulator is running on an Alienware PC with the following specifications:

- Installed memory (RAM): 16;
- Operating System: Windows 7 64-bit;
- Processor: Intel Core i-73930 3.2 GHz;
- Video Card: NVIDIA GeForce GTK 555.

Another part of the surgical simulator setup is the haptic device. Haptic interactions are modeled to generate force feedback when collision occurs between the Introducer and any part of the body. The device used is a High-force Phantom Premium 1.5 6DOF.

In addition, the MiddleVR plug-in is utilized to provide the two different displays Thoracoscopic and Mirror, each supported by a separate camera. All these components are operating as the simulation is running and the four methods are each implemented separately with the simulator setup for adequate comparison.

4.9.2 Symmetry Condition

In the first criterion to be considered, the question is asked whether or not the implemented method requires a symmetry property for the modeled scene. If the graphical scene was a simple cube, then there is no great emphasis on whether or not the right side of the cube is projected consistently as the right side in the display as well. This consistency is required if the model is more complex and has directional attributes.

The evaluation of this criterion will be describing each method as being *symmetry-dependent* or *symmetry-independent*. Obviously, a method that is not
symmetry-dependent is more suitable for most applications. Such a method will work even if the graphical scene is complicated and has non-geometric-shaped components.

4.9.3 Model Preservation

This aspect looks into whether or not the method implemented has distorted the actual 3D model in any way. In some or perhaps most cases, it is undesirable that the collocation method require the 3D model itself to be altered in any way especially graphical scenes that include simulated scenarios.

This criterion will be evaluated by describing each method as being model-preserving or model-disturbing. A method that is model-preserving will solve the collocation issue and correct the mirrored image without affecting the 3D model's structure itself.

4.9.4 Resolution and Clarity

This aspect questions the clarity of the outcome when the graphical scene is displayed. In this criterion, all the implemented methods will be gauged in how clear the image is when it is projected on the mirror and how visible it is when the simulation is running.

Clarity is a relative measure, and the four methods do not have great differences in that aspect. However, to be able to evaluate this criterion, a specific aspect will be looked into. A method will be described as clear if it displays the original graphical scene directly on the LCD. A method will be described as indirect if the LCD displays the graphical scene indirectly or through looking at another camera's projection which results
in a 2D view of the scene. Lastly, a method will be described as *unclear* if the image is not displayed clearly or if the resolution appears lower than other methods.

4.9.5 Speed and Performance

In this criterion, the computational speed and simulation quality will be investigated. The main measure for this aspect is the number of *frames per second* (FPS) in the simulation. The different methods will be implemented separately and in each case, the speed will be measured and compared and any lags or delays observed in any of the cases will be reported.

To be able to measure its speed and performance, each method will be implemented and the surgical simulation will be run. Then the scene's speed (frames per second) will be recorded in real-time to evaluate each method's speed.
CHAPTER 5
RESULTS

The results of this work have been collected after applying the methods discussed in the previous chapter, resolving the underlying problems. The chapter demonstrates and discusses the outcomes regarding the three main tasks defined early in this work. These tasks are:

a) Achieving consistent coordinate system alignment between the Mirror Display and Thoracoscopic Display (Visuo-Haptic coordinate systems configuration).

b) Correcting image inversion displayed on the mirror (Visuo-Haptic graphical configuration).

c) Running both displays at the same time each on a separate viewport in real-time.
   (Synchronous real-time multiple displays)

Also presented in this chapter are the comparison and evaluation results of the four methods developed for achieving the requirements of “Task b”.

5.1 Visuo-Haptic Coordinate Systems Configuration (Task a)

Two pairs of matrices were developed, a pair for each display. For the Thoracoscopic Display, the position matrix will be denoted as $PM_{Th}$ and the rotation (quaternion) matrix will be denoted as $QM_{Th}$. Similarly, for the Mirror Display, the position matrix will be denoted as $PM_{M}$ and the rotation matrix will be denoted as $QM_{M}$.

Each display’s configuration utilizes a position matrix and a rotation matrix. In the position matrix, the X-Y-Z components of the position vector for the haptic device
were multiplied in a dot product by the constructed matrix to correct the linear motion. The same thing was done for the X-Y-Z components of the quaternion parameter of the haptic device to yield the correct rotational motion.

For the Thoracoscopic Display’s position configuration, the matrices multiplication should not invert any of the axes. For its rotation configuration, however, the matrices multiplication should invert the Y-component (Yaw rotation). This operation is shown in the following:

\[
\begin{align*}
\text{Position Matrix } (PM_{Th}) & = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} & \text{(2)} \\
\text{Position Matrix } (PM_{Th}) & = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} & \text{(3)}
\end{align*}
\]

For the Mirror Display’s position configuration, the matrices multiplication should invert the Z-axis. For its rotation configuration, the matrices multiplication should invert the Z-component (Roll rotation). These operations are shown in the following:

\[
\begin{align*}
\text{Position Matrix } (PM_{Th}) & = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & -1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} & \text{(4)} \\
\text{Position Matrix } (PM_{Th}) & = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & -1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} & \text{(5)}
\end{align*}
\]
In order to execute each configuration separately as needed, input interactions were built in the Virtools platform to link these transformation matrices to their appropriate recipients when required. These matrices were integrated in Virtools and were modeled so that each matrix affects the corresponding vector.

When the Thoracoscopic Display is required only, the corresponding configuration explained above can be applied by pressing the number 1 on the keyboard. Doing that activates the position and rotation transformation matrices of the thoracoscopic configuration and affects the position and quaternion components of the haptic device. Similarly, when the Mirror Display is required only, number 2 is pressed to apply the mirror configuration to the haptic device’s position and quaternion components. However, if a method was implemented that would originally account for the reversed image, the regular thoracoscopic configuration can be applied for both and they can run simultaneously at the same time. This allowed for the actual physical motion of the haptic device to be consistent with the haptic rendering in the graphical scene.

5.2 Visuo-Haptic Graphical Configuration (Task b)

After achieving coordinate system alignment to allow for consistent motion of the surgical tool in the scene with the haptic device’s stylus, the next task was implemented. For the Thoracoscopic Display, merely placing the camera in the correct position to provide an internal view of the patient’s right side and displaying it on the monitor is sufficient. But for the Mirror Display it is not the case as the image has to be processed or adjusted somehow to account for the reflectance problem.
The methods for achieving correct visuo-haptic graphical collocation and resolving the image error caused by mirror reflectance are executed.

5.2.1 Method 1 Implementation: Manipulation of Camera Position and Orientation

In this method, the approach for adjusting the graphical scene in order to account for mirror reflection is done by simply manipulating the position and orientation of the used camera. This was done by positioning the camera on the other side of the torso (left side) and rotating it 180 degrees (flipping the camera’s up-direction). This resulted with an initial view of the scene in the LCD that is similar to the Baseline Reflectance error and when projected from the LCD onto the mirror, it provided the spectator with the correct view (Figure 43 left). When perceiving the mirror now, the patient’s body appeared exactly in the correct orientation lying down facing upward with the chest’s right side facing the surgeon. The camera was fixed in that position and orientation and was named “Mirror Display” (Figure 43 right).

Figure 43. View of the patient’s body in the scene replicating the Baseline Reflectance error (left) and its corresponding image in the mirror (right).
5.2.2 Method 2 Implementation: 3D Object Polygons Inversion

In this method the approach was to change the properties of the objects in the scene. To make these changes visible, the letter R was printed in red as part of the torso’s texture and placed on the right side. After flipping the polygons, the letter R appeared on the side away from the spectator (the torso’s left side). To produce the Baseline Reflectance error, the camera was situated in a way similar to the first method by placing it on the body’s left side and flipping its up-direction (Figure 44 left).

When this image is projected on the mirror, the patient's avatar appears in the right orientation and the user is facing the right side of the patient that has the letter R just like in the original model (Figure 44 right). In the previous method, this would be the left side appearing as the right side, but after inverting the polygons of the model it is the correct side.

Figure 44. Image of body after flipping its polygons in the Y-direction (left) and its corresponding image in the mirror.
5.2.3 Method 3 Implementation: Render to Texture and Plane Inversion

In this method, the render-to-texture capability was used in Unity to create a 2D texture for a thin plane that was created. This plane was then inverted in the y-direction flipping all its polygons and preserving those of the actual object. Then, another camera with orthogonal projection was positioned to view this plane in real-time and its view is projected onto the mirror (Figure 45).

![Figure 45. Additional camera providing an orthogonal view of the render-textured and inverted plane.](image)

Observing the mirror, the patient's right side is seen facing the user as required (Figure 46). This solution does not require any manipulation of the model itself as the inversion affects the external plane that was created and does not impact any of the 3D models in the scene. Furthermore, the user perceives the right side of the patient with no
errors and the camera used did not need to be placed on the left side of the body nor flipped in any way, preserving, with that, the correct position and orientation of the camera.

Figure 46. Corresponding image in the mirror of an orthogonal projection of the textured plane.

5.2.4 Method 4 Implementation: Customized Projection of the Graphical Scene

In this method, programming was used to adjust the projection matrix in order to flip the projection of the 2D image onto the screen. When doing that, however, the back- and front-facing polygons are swapped causing an alteration in the scene’s render. The back-face determination criterion was therefore inverted to compensate for that change (Figure 47).
These commands were programmed in C# code that was added as a component of the Mirror Display camera in the scene so that it affects that camera only. When running this simulation, the problem was solved resulting with a correct view of the patient’s body in the Mirror Display. In this solution, the plane created before was eliminated as well as the third orthographic camera as there is no need for them anymore (Figure 48).
After implementing this method for visuo-haptic colocation and achieving coordinate system alignment, the configuration component can be constructed to adapt to other similar VR applications. For the NPSS, the configuration component will be integrated to adjust the mirrored image in one of the two viewports and the transformation matrices will be used to adjust the coordinate system configuration between the graphical and haptic interfaces. Figure 49 shows a schematic of the result NPSS setup.

Figure 49. Schematic of NPSS structure with configuration component.
5.3 Comparison and Evaluation of Methods

The four methods developed for visuo-haptic collocation via optical reflection were implemented as explained in previous sections. The implementation showed that some of them fill gaps that others do not. To quantify these differences, a comparison was designed where each method implemented is evaluated against relevant criteria.

This section reflects the results of that comparison and assesses each method according to the explained design.

5.3.1 Method 1 Evaluation

In this method, the camera was placed on the left side of the chest and rotated 180 degrees which resulted in a (so-appearing) correct right side view of the patient lying down in the correct position. The side displayed as the right side here is in fact composed of the polygons of the left side but only appearing as the right side as a result of the camera’s position. In order for this method to be viable, the 3D model must have symmetrical sides, and the chest’s right side must not differ in shape from its left side. This makes this method dependent on the symmetry of the used object in the scene.

This method’s implementation, however, does not affect the polygons composing the object itself in any way as only the camera’s position and orientation is manipulated. This method does not need any in-between steps when displaying the scene as the camera’s view is directly rendered to the display. With this method implemented, the simulation was running at a speed of 78 FPS.
5.3.2 Method 2 Evaluation

In this method, the polygons of the objects are flipped in the direction of reflection so that what appears as the right side in the mirror is actually composed of the right side polygons. For this method to be applicable, however, the objects used must have identical sides so that flipping the polygons will not be a problem. This makes this method dependent on symmetry as well. Furthermore, this method disturbs the composition of the 3D models and reverses their structure.

This method does not need any in-between steps either since the view of the camera is directly sent to the display. With this method implemented, the simulation was running at a speed of 77 FPS.

5.3.3 Method 3 Evaluation

In this method, a 2D texture is extracted from the camera viewing the right side of the model and applied to a plane. The plane is then inverted in the direction of reflection and another camera is viewing this plane and projecting that view on the mirror. This way, the used camera does not need to be placed on the left side or inverted in any way which makes it independent of the symmetry of the models. Furthermore, the model is preserved and its polygons are not inverted in any way.

The image appearing in the mirror, however, is not of the same quality as when the model is displayed directly. Using an orthographic projection to display a perspective rendering of another camera reduces the resolution. With this method implemented, the simulation was running at a speed of 69 FPS.
5.3.4 Method 4 Evaluation

In this method, the graphical scene is rendered normally without any alterations. Then, before displaying it, the Y-component of the Projection Matrix is inverted in a code assigned to that camera only resulting with the correct view desired.

The camera is not moved or inverted in any way and nor is the model itself altered. The quality and resolution of the image on the mirror is very clear and is displayed directly from the scene. With this method implemented, the simulation was running at a speed of 78 FPS.

5.3.5 Evaluation Summary

After discussing each method's outcome, a brief inclusive summary of the evaluation for each method is presented according to the criteria introduced earlier. Table 3 shows a summary of the comparison of the four used methods against the criteria defined.

<table>
<thead>
<tr>
<th></th>
<th>Method 1</th>
<th>Method 2</th>
<th>Method 3</th>
<th>Method 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symmetry Condition</td>
<td>symmetry-dependent</td>
<td>symmetry-dependent</td>
<td>symmetry-independent</td>
<td>symmetry-independent</td>
</tr>
<tr>
<td>Model Preservation</td>
<td>model-preserving</td>
<td>model-disturbing</td>
<td>model-preserving</td>
<td>model-preserving</td>
</tr>
<tr>
<td>Resolution and Clarity</td>
<td>clear</td>
<td>clear</td>
<td>indirect</td>
<td>clear</td>
</tr>
<tr>
<td>Speed and Performance</td>
<td>78</td>
<td>77</td>
<td>69</td>
<td>78</td>
</tr>
</tbody>
</table>
After performing the comparison, the 4th method which implements an adjusted projection matrix proved to be the best solution. It fulfills all requirements with no exception as it preserves the location of the camera as well as the objects in the scene and it provides direct rendering to the display without in-between steps. Furthermore, the solution does not have any negative influence on the run speed of the simulation.

5.4 Synchronous Real-Time Multiple Displays (Task c)

The third and last task requires that the simulator is able to run the surgery providing an external view of the patient's right side on the LCD projecting the image on the mirror (Mirror Display), as well as an internal thoracic view displayed on a separate monitor (Thoracoscopic Display) both at the same time in real-time. MiddleVR was utilized in Unity to provide two separate viewports, each controlled by a separate camera in the scene and each linked to a separate display.

This setup was implemented adapting the 4th and most suitable method developed for visuo-haptic collocation which is through utilizing customized projection of the graphical scene. After doing that, the user is able to perceive both views running synchronously on the two separate displays smoothly. Figure 50 shows the two displays of the surgical simulator, and Figure 51 shows both displays running at the same instant providing two separate views of the same model, in two separate viewports.
Figure 50. View of Thoracoscopic Display (top left) and Mirror Display (top right).

Figure 51. Synchronos run of both displays in real-time after configuring two parallel viewports.
The overall objective of a simulator is a consistent imitation of the real system with appropriate realism. Derived from this idea, the user's interaction with our surgical simulation scene was constructed through vision and haptic feedback. This visuo-haptic interface can be achieved through the use of a half-silvered mirror on which the scene is displayed and underneath which the user perceives, in the same viewing frame, his/her hand moving the surgical tool in space using the haptic device.

The setup of the Nuss Procedure surgical simulator, therefore, implements optical reflection principles to allow the user to view the external view of the right side of the virtual patient's torso lying down. At the same time, the user can observe the patient's interior through another monitor that displays the thoracoscopic view. Therefore, the main objective was to have one graphical scene and display it in two different ways at the same time. The difference between the two views is the reversed image on the mirror.

This work fulfills this idea through creating, for each view, a different configuration that modifies the configuration of the displayed image as well as the different coordinate systems achieving a smooth and correct function of the simulator.

A multi-configuration mirror display capability was developed for virtual environments that incorporate the use of haptic feedback and mirror reflection with the user interface in real-time. The main objective of this work is to construct a virtual environment component that is able to (a) achieve coordinate system alignment between haptic device, graphical scene and mirror-projected graphical scene, (b) correct the
reversed image in the mirror in an optimized manner and (c) be able to synchronously run multiple displays in real-time with or without a mirror component.

The construction of this component was a part of the Nuss Procedure surgical simulator development and that was the platform on which it was implemented. However, it can be further implemented to any comparable work or setup consisting of similar primary components and achieve the same objectives. The aim is, therefore, to achieve visuo-haptic collocation in any generic virtual environment while using optical reflection.

After reviewing the literature, it was found that most comparable works achieve one or two of the previously discussed objectives. Various solutions were introduced by researchers to achieve these tasks and they were used in this work to reach an optimal collocation method.

For the first task, transformation matrices were developed to allow for consistent behavior between the haptic device and the virtual scene. An investigation was performed to define the different world spaces for the haptic device, the graphical scene and the graphical scene's reflection in the mirror. Then, the suitable transformation matrices for position and rotational components were constructed to achieve correct alignment between the haptic device and the regular display on one hand and between the haptic device and the mirror display on another.

The results showed consistent behavior of the haptic device's motion in the regular display setup, in the mirror display setup and in the setup where both displays are running at the same time.
For the second task, four different methods were attempted to correct the reversed image in the mirror. In the first one, the camera was positioned and oriented in a way to display the opposite side of the scene so that when viewed in the camera, it will seem correct. However, this solution is not reliable for applications that involve interactions and complexity. In the second method and to correct the error from the previous one, the polygons composing all the objects in the scene were inverted in the direction of reflection causing the left and right sides of the polygons to be swapped. Despite the fact that this action eliminates the error from the previous method, it involves a change in the object's construction which is in many cases undesirable. Therefore, a third method was developed where the camera was placed in its correct position and orientation, and its view was rendered to become a texture of an external thin plane placed far away in the same scene. The polygons composing this plane were then inverted in the direction of reflection and another camera was then created and situated to view this plane's texture. This resulted with the correct view of the scene without disturbing the structure of its objects. However, not only does this approach contain multiple tedious steps, but the resulting image in the mirror is actually an image from a camera's projection that is a projection of another camera which provides a low resolution image quality not to mention unclear details especially if the scene contained small and detailed objects. Therefore, in the fourth method all these steps were eliminated by adjusting the most relevant step in the viewing pipeline which is the projection transformation where the 3D scene is transformed into a 2D image to be projected on the screen. To adjust it, a new customized projection matrix was used to invert the scale of the matrix in the direction of
reflection. This resulted in a seamless correction of the mirrored image and a faster frame rate with no need to affect the properties of used cameras nor the model itself.

The results showed that the fourth method achieves the required task without generating any of the errors that the previous three generate. It was also shown that the method of using a customized projection matrix results in a clear direct image of the scene without affecting the speed of the simulation.

For the third task, a third party plug-in "MiddleVR" was utilized to create two separate viewports for the graphical scene and assigned a camera and a display (hardware) to each viewport. When both viewports were configured correctly implementing also the results of the two previous tasks, the simulation operated on two separate displays providing two different views to the scene in the simulation.

The component developed in this work can be standardized and generally used for implementation on any application consisting of any monitor- or projector-based display setup and a haptic device. This component of the virtual environment can be utilized to adjust the model's configuration to make the application run correctly taking into account the error that a mirror image creates.

Future work includes the final packaging of the designed configuration component to result in a small plug-in that can be integrated with the structure of any virtual reality application. In addition, the design of the configuration component will be expanded to be able to handle two haptic devices and adjust their configurations correctly as many applications utilize the use of two devices, one for each hand.
REFERENCES


[68] W. Ji, "Viewing Options for the Virtual Haptic Back (VHB)," Master of Science (MS), Mechanical Engineering, Ohio University, Athens, Ohio, 2005.


APPENDIX A: PERMISSION TO USE FIGURES

<table>
<thead>
<tr>
<th><strong>ELSEVIER LICENSE</strong></th>
<th><strong>TERMS AND CONDITIONS</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Jul 12, 2013</strong></td>
<td></td>
</tr>
</tbody>
</table>

This is a License Agreement between Mohammad F Obeid ("You") and Elsevier ("Elsevier") provided by Copyright Clearance Center ("CCC"). The license consists of your order details, the terms and conditions provided by Elsevier, and the payment terms and conditions.

**All payments must be made in full to CCC. For payment instructions, please see information listed at the bottom of this form.**

| Supplier               | Elsevier Limited          |
|                       | The Boulevard,Langford Lane |
|                       | Kidlington,Oxford,OX5 1GB,UK |
| Registered Company Number | 1982084                  |
| Customer name         | Mohammad F Obeid          |
| Customer address      | 4500 Elkhorn Ave          |
|                       | NORFOLK, VA 23529         |
| License number        | 3186570388326             |
| License date          | Jul 12, 2013              |
| Licensed content publisher | Elsevier                  |
| Licensed content publication | Computers in Biology and Medicine |
| Licensed content title | A virtual training simulator for learning cataract surgery with phacoemulsification |
| Licensed content author | Kup-Sze Choi,Sophia Soo,Fu-Lai Chung |
| Licensed content date | November 2009             |
| Type of Use           | reuse in a thesis/dissertation |
| Portion               | figures/tables/illustrations |
| Number of figures/tables/illustrations | 1 |
| Are you the author of this Elsevier article? | No |
| Will you be translating? | No |
| Order reference number | 1 |
| **Title of your thesis/dissertation** | A Multi-Configuration Display Methodology Incorporating Reflection for Real-Time Haptic-Interactive Virtual Environments |

| Elsevier VAT number | GB 494 6272 12       |
| Permissions price  | 0.00 USD             |
| VAT/Local Sales Tax | 0.00 USD / 0.00 GBP  |
| Total               | 0.00 USD             |

Terms and Conditions
This is a License Agreement between Mohammad F Obeid ("You") and Springer ("Springer") provided by Copyright Clearance Center ("CCC"). The license consists of your order details, the terms and conditions provided by Springer, and the payment terms and conditions.

All payments must be made in full to CCC. For payment instructions, please see information listed at the bottom of this form.

License Number: 3191480832710
License date: Jul 17, 2013
Licensed content publisher: Springer
Licensed content publication: Springer eBook
Licensed content title: Realistic haptic volume interaction for petrous bone surgery simulation
Licensed content author: A. Petersik
Licensed content date: Jan 1, 2002
Type of Use: Thesis/Dissertation
Portion: Figures
Author of this Springer article: No
Order reference number: 3
Title of your thesis / dissertation: A Multi-Configuration Display Methodology Incorporating Reflection for Real-Time Haptic-Interactive Virtual Environments
Expected completion date: Aug 2013
Estimated size(pages): 130
Total: 0.00 USD

Terms and Conditions
VITA

Mohammad F. Obeid

Modeling, Simulation and Visualization Engineering Department

Old Dominion University

Norfolk, VA 23529

Educational Background

M.Sc. : August 2013, Old Dominion University, Norfolk, VA, USA
Major : Modeling and Simulation

B.Sc. : June 2011, German-Jordanian University, Amman, Jordan
Major : Industrial Engineering
Thesis : Utilizing Lean Principles for Changeover Time Reduction at Hikma Pharmaceuticals.