Virtual Assembly and Disassembly Analysis: An Exploration into Virtual Object Interactions and Haptic Feedback

A Thesis
Presented to
the Academic Faculty

By

Adam S. Coutee

In Partial Fulfillment
of the Requirements for the Degree of
Doctor of Philosophy in Mechanical Engineering

Georgia Institute of Technology
May 2004

Evaluating Haptic Feedback in a Virtual Environment for Assembly and Disassembly

Approved by:

Bert Bras, Chair
Professor, Mechanical Engineering

Nelson Baker
Associate Professor, Civil and Environmental Engineering

Paul Griffin
Associate Professor, Industrial and Systems Engineering

Chris Paredis
Assistant Professor, Mechanical Engineering

David W. Rosen
Professor, Mechanical Engineering

May 27, 2004

ACKNOWLEDGMENTS

With the completion of this dissertation, I conclude five years of graduate study and five fantastic years of my life, during which time I have grown, in life and in size, including the marriage to my beautiful wife. Although only one name appears as the author of this document, there are many more who deserve thanks and credit, without which its completion would have never come to realization.

I would first like to thank my advisor Dr. Bert Bras for his guidance throughout my graduate studies. His support, advice, and encouragement through both good and troublesome times have been invaluable.

I would like to thank my committee members for their guidance, support, and suggestions. The feedback received has greatly improved the quality of my research and this dissertation.

Throughout my graduate study, I have had the pleasure of being a member in the Systems Realization Laboratory. The atmosphere and friends I have made through this lab will be remembered always.

I am very appreciative of the people who volunteered to participate in the experimental studies conducted during this work, without which this research would not have been possible.

I would like to thank my loving wife, who has supported this endeavor from the day we met. Your unending support has made this journey possible. I look forward to beginning a new chapter in our lives together, and all thereafter. I thank my family for

their enthusiasm and interest in my progress toward the completion of this research. It is your support throughout my life that has led me to this point.

I am thankful for the financial support from the National Science Foundation (NSF Grant DMI-9908335, NSF Grant DMI-9624787, and an NSF Graduate Research Fellowship) and the Georgia Research Alliance's Environmental Technology Consortium. It is this funding that makes this research possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

2D	Two-dimensional
3D	Three-dimensional
ANOVA	Analysis of Variance
API	Application Programming Interface
DOF	Degrees of freedom
GHOST	General Haptic Open Software Toolkit
GLUI	OpenGL User Interface
GLUT	OpenGL Utility Toolkit
GUI	Graphical User Interface
HIDRA	Haptic Integrated Dis/Re-assembly Analysis
OpenGL	Open Graphics Library
PHANToM	Personal HAptic iNTerface Mechanism
POSIX	Portable Operations System Interface
Pro/E	Pro Engineer
SDK	Software Development Toolkit
SGI	Silicon Graphics Incorporation

SUMMARY

In recent years, researchers have developed virtual environments, which allow more realistic human-computer interactions and have become increasingly popular for engineering applications such as computer-aided design and process evaluation. For instance, the demand for product service, remanufacture, and recycling has forced companies to consider ease of assembly and disassembly during the design phase of their products. Evaluating these processes in a virtual environment during the early stages of design not only increases the impact of design modifications on the final product, but also eliminates the time, cost, and material associated with the construction of physical prototypes. Although numerous virtual environments for assembly analysis exist or are under development, many provide only visual feedback. A real-time haptic simulation test bed for the analysis of assembly and disassembly operations has been developed, providing the designer with force and tactile feedback in addition to traditional visual feedback.

The development such a simulation requires the modeling of collisions between virtual objects, which is a computationally expensive process. Also, the demands of a real-time simulation incorporating haptic feedback introduce additional complications for reliable collision detection. Therefore, the first objective of this work was to discover ways in which current collision detection libraries can be improved or supplemented to create more robust interaction between virtual objects. Using the simulation as a test bed, studies were then conducted to determine the potential usefulness of haptic feedback for analysis of assembly and disassembly operations. The following significant contributions

were accomplished: (1) a simulation combining the strengths of an impulse-based simulation with a supplemental constraint maintenance scheme for modeling object interactions, (2) a toolkit of supplemental techniques to support object interactions in situations where collision detection algorithms commonly fail, (3) a haptic assembly and disassembly simulation useful for experimentation, and (4) results from a series of five experimental user studies with the focus of determining the effectiveness of haptic feedback in such a simulation. Additional contributions include knowledge of the usability and functionality of current collision detection libraries, the limitations of haptic feedback devices, and feedback from experimental subjects regarding their comfort and overall satisfaction with the simulation.

CHAPTER I

MOTIVATION FOR HAPTIC ASSEMBLY AND DISASSEMBLY SIMULATION

In recent years, researchers have developed virtual environments, which allow more realistic human-computer interactions and have become increasingly popular for engineering applications such as computer-aided design and process evaluation. For instance, the demand for product service, remanufacture, and recycling has forced companies to consider ease of assembly and disassembly during the design phase of their products. Evaluating these processes in a virtual environment during the early stages of design not only increases the impact of design modifications on the final product, but also eliminates the time, cost, and material associated with the construction of physical prototypes. Although numerous virtual environments for assembly analysis exist or are under development, many provide only visual feedback. We are developing a real-time haptic simulation for assembly and disassembly evaluation, providing the user with force and tactile feedback in addition to traditional visual feedback. In this chapter, the motivation for and an introduction to the research described in this dissertation is provided. In Section 1.1, the incentive for a virtual assembly and disassembly environment will be discussed, followed by an introduction to haptic technology, which can be used to provide feedback through the sense of touch (Section 1.2). A review of literature in automated techniques and virtual environments for assembly and

disassembly analysis, including those with and without haptic feedback, is presented in Section 1.3. Finally, the focus for the research in this dissertation is outlined in Section 1.4, including development of the research questions, strategies for answering these questions, and the contributions from the research. The organization of the remaining chapters in this dissertation is discussed in Section 1.5.

1.1 THE INCENTIVE FOR VIRTUAL ASSEMBLY AND DISASSEMBLY ANALYSIS

Modern design practice demands that the entire life cycle of a product be considered during the design stage. Design engineers must not only consider the functionality of a product but also address expected requirements throughout its useful life, from initial manufacturing and assembly of a product to service, maintenance, and final disposal or recycling. This requires that engineers are forward thinking and predict and design for the various stages of a product's life. The diagram shown in Figure 1.1 highlights the role of service, reuse, and recycling during the life of a typical product, from design to disposal.

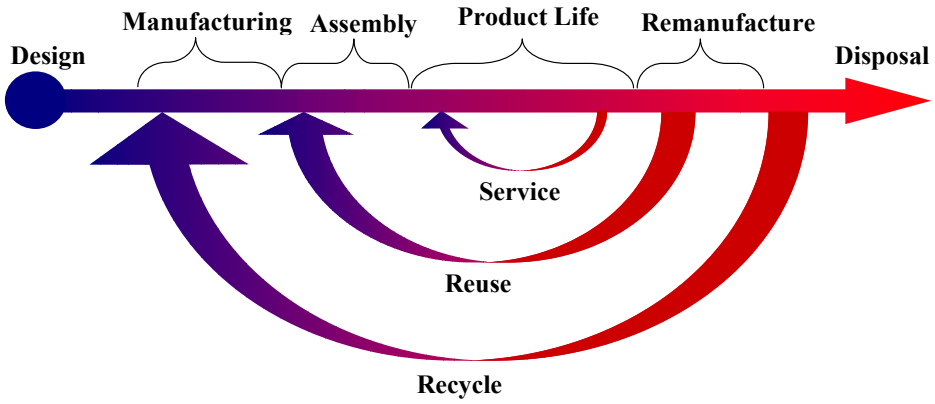


Figure 1.1: Service, Reuse, and Recycling during the Life of a Product

The major stages during the life of a product, as depicted in Figure 1.1, are design, manufacturing, assembly, product life, remanufacture, and disposal. First, a product must be designed, whether it is a new or modified design. Then, each component must be manufactured, and the manufacturability must be considered during the design phase. The final step in manufacturing is the assembly of all components into the final

product. At this stage, the product may be utilized for its intended purpose. Remanufacture allows the reuse of components that may still provide value to future products. Finally, disposal of unusable components signifies the end of the product's time line as shown above. The typical life of a product may last from months to years, but each of the stages described above must be considered during the design process. Additionally, service, reuse, and recycling have the commonality that assembly and disassembly are required to perform such tasks. Therefore, ease of assembly and disassembly are important aspects of a product that should be addressed during its design.

Ease of assembly has always been a key consideration during the design process because it has a direct effect on the manufacturability and cost of the product. The effort to create a product with simplified and less expensive assembly procedures is known as Design for Assembly (DFA). In addition to DFA, other design techniques such as designing for service and designing for recycling require the designer to address the ease of assembly and disassembly. When servicing a product, simplicity in removal and replacement of worn or old components is necessary to reduce time and cost in the process. Environmental concerns have also driven the desire for easy disassembly. At the end of a product's life, the design should allow simple removal of reusable components for remanufacture. Also, separation of recyclable materials and possibly hazardous materials facilitates the reuse of raw materials for other products. Each of these concerns can be directly affected by the ability to assemble or disassemble the components of a product.

In recent years, computers have provided engineers with increased computational capability and software tools that can assist in the design and testing of a product. Using

the computer as a tool, researchers have developed virtual environments, which allow more realistic human-computer interactions and have become increasingly popular for engineering applications such as computer-aided design and process evaluation. The virtual environments can also be utilized to evaluate ease of assembly and disassembly as companies are forced to address the demand for product service, remanufacture, and recycling. Evaluating these processes in a virtual environment during the early stages of design not only increases the impact of design modifications on the final product, but also eliminates the time, cost, and material associated with the construction of physical prototypes. Although numerous virtual environments for assembly analysis exist or are under development, many provide only visual feedback. We are developing a real-time haptic simulation for the analysis of assembly and disassembly operations, providing the designer with force feedback in addition to traditional visual feedback. The ultimate goal is to provide an assembly and disassembly simulation environment where designers and process engineers can get high fidelity visual and force feedback with respect to the assembly and disassembly of their products. The work conducted in this research and described in this document are initial steps toward this ultimate goal.

To begin the discussion of force enabled virtual environments, the genre of haptics, which allows a user to interact with a virtual environment through the sense of touch, will be introduced in the next section.

1.2 THE ESTABLISHMENT OF HAPTIC TECHNOLOGY

The term haptic comes from the Greek word *haptesthai* meaning 'to touch'. Essentially, haptic refers to anything relating to or based on the sense of touch. In the engineering domain, haptics is the science of integrating the sense of touch between a

human and the computer, or, more correctly, a virtual environment modeled on the computer. Through a haptic device, or haptic interface, a person can interact with the virtual environment by manipulating components or features, while receiving feedback through the sensation of touch. As such, a haptic interface acts as both an input and output device to the computer, transmitting the motion of the user and the force response of virtual objects touched. Given steady advances in the computational power of computers, haptics has become a major research issue in the past decade. In the following sections, a review of haptic devices will be provided, followed by areas of research application, and a motivating example for use of haptics in assembly and disassembly analysis.

1.2.1 A Review of Haptic Devices

Haptic interfaces can be organized into two main categories: tactile and kinesthetic feedback. Tactile haptic devices stimulate the receptors for touch in our skin, providing information such as surface texture, slippage, and temperature. Kinesthetic haptic devices, on the other hand, respond to the general motions of the arm, hand, or other body part. These devices provide information on the inertia, weight, or compliance of an object in a virtual environment. Burdea provides good overview of current haptic technology, its limitations for interaction with virtual environments, and the physical modeling necessary for haptic interaction in a virtual environment (Burdea 2000). In this paper, Burdea discusses two basic types of haptic interfaces: desktop, or grounded, devices and wearable devices. Three major limitations are also cited: large haptic interface weight (for wearable devices), tracker errors in large workspaces (again, for wearable devices), and high bandwidth requirements.

Haptic devices come in a variety of forms, with respect to design, capability, and price. The methods for supplying force or tactile feedback also vary widely between haptic devices, including wire-driven, electric motors, pneumatics, magnetism, and air jets. Arguably the most common haptic device is a force feedback joystick. These devices, produced by a number of companies are generally used in the gaming industry and are comparatively inexpensive. However, when considering assembly and disassembly analysis, joystick devices are somewhat limited in their capability. Haptic interfaces that can be used to model forces to the human hand, or grasping procedures using the hand, are more appropriate for assembly and disassembly analysis.

There are a limited number of commercially available haptic devices that supply force or tactile feedback capable of simulating grasping. Immersion Corporation is a leader in the haptic industry that provides a number of haptic devices for use in engineering applications. One of their most basic devices is the CyberTouch™, a tactile feedback glove. The CyberTouch™ is built upon the position measuring CyberGlove®, which provides 22 joint-angle measurements of the hand. There are 6 vibrotactile actuators mounted on the glove that operate at frequencies up to 125 Hz. A limitation of the CyberTouch™ for haptic assembly and disassembly is that it does not provide force feedback. The CyberGrasp™, also developed by Immersion, is a wearable haptic device that provides resistive force feedback to each finger. Finally, the CyberForce® incorporates the CyberTouch™ and a grounded arm to provide force feedback to the fingers and the hand as a whole.

Another leader in commercial haptic devices is SensAble Technologies, Inc, who provides a full line of PHANToM haptic interfaces. The original PHANToM design was

a result of graduate research conducted by Massie (Massie 1996). All PHANToM interfaces are desktop devices that provide force and tactile feedback through an arm to the user's fingertip or hand, depending on the configuration. The array of PHANToMs vary based on the size of the workspace and the degrees of freedom for force interaction, either 3D or 6D.

1.2.2 Research Application of Haptic Technology

Haptic technology is the subject of a large and diverse body of research, as adding the sense of touch to the traditionally visual and/or auditory virtual environments promises to improve the realistic nature of such simulation. Much of haptics research is in the medical field, relating to medical training (Baumann and Clavel 1998; Burdea *et al.* 1998; Dawson and Kaufman 1998), surgical simulation, and rehabilitation. Progress toward computer interaction for the blind (Colwell *et al.* 1998; Hardwick *et al.* 1998) has also begun.

In the engineering domain, explorations regarding the use of haptics for scientific data visualization (Neves *et al.* 1997; Dureck *et al.* 1998), computer-aided design (Springer and Gadh 1997; Stewart *et al.* 1997), and telerobotics (M'Sirdi 1993; Sheffield *et al.* 1993) are also being conducted. In a study to test the effectiveness of haptics, Volkov and Vance (Volkov and Vance 2001) show that the use of force feedback leads to a reduction in time for making design decisions. Research at Georgia Institute of Technology has been conducted to design a 3D haptics device known as Digital Clay, which enables tactile and force interaction with a virtual model through a deformable surface (Askins and Book 2003; Rosen *et al.* 2003). Thus far, this work has included defining the geometry, manufacturing, and control of such a device.

In recent years, additional development has begun on haptic environments for assembly and disassembly evaluation and analysis. As this is also the focus of this research and dissertation, a more in-depth literature review is provided in Section 1.3.

1.2.3 A Motivating Example

Imagine the task of replacing the oil filter in a car. The oil filter in a car is typically found on the engine block, but can be placed in many locations around the engine compartment. In this example, let's assume that the oil filter screws into position on the backside of the engine block when looking from the front of the car. Additionally, imagine that you, the reader, are the individual completing the task.

After raising the hood of the car, you first look around the engine compartment and see that the oil filter is located on the rear side of the engine block, furthest from the front bumper and placed about midway down the engine. As you lean over the engine, you feel warmth as the car was recently driven, and as you stand upright you notice a grease spot already on the front of your pants. As a precaution not to harm yourself and to protect your clothing, you reach for a mat and place it over the engine. Leaning back over the engine in an awkward position, you reach for and attempt to unscrew the oil filter, but it is fixed to the engine too tight. You reposition yourself in a more convenient location around the side of the car and reach to unscrew the oil filter a second time. This time, with a closer position and better grip, you are able to loosen the oil filter and proceed to remove it. As you turn the oil filter on last time to unscrew it from the engine block, the filter almost slips from your hand, but a tightened grip prevents it from falling to the ground. After achieving a good grip on the oil filter, you stand upright again, bumping your head on the hood in the process. Slightly annoyed about hitting your head, you turn

and place the old oil filter on the workbench to the side and grab the new filter. Leaning back over the car, you attempt to screw the new oil filter in position but are unsuccessful on the first few tries. Finally, after a couple of minutes you are able to get the threads on the oil filter to match those on the engine, and the filter begins to screw in. After several turns, the filter begins to tighten and one last twist assures that the filter is in the proper place. Standing up this time, you are careful not to bump your head on the hood. After only a short period of time and satisfied with your success in replacing the oil filter, you remove the protective mat, close the hood, and look for a drink of water.

Now imagine that you are in an empty room wearing stereoscopic goggles and a full body haptic feedback suit. The entire process of replacing the oil filter was performed in a virtual world modeled on a nearby computer. Although the car does not actually exist, you were able to feel the heat radiating from the engine and the car body pressing against your legs as you leaned over the edge of the car. In addition, you can still point to the spot where you bumped your head on the hood. Even though you completed the task in a virtual environment, everything felt as if it would in the real world. Luckily, however, you don't actually have a grease spot on your pants.

So what is the point of this example? Replacing the oil filter of an automobile in a virtual environment, as described above, is a futuristic vision of the power of haptic devices. Obviously, full body force feedback through a haptic suit is beyond the technical capabilities of haptic devices and virtual environments today. In fact, haptic technology is currently at the infant stage of its development, desiring many more years and even decades of research. The example given above is provided as an inspiration of the possible future of haptic feedback in simulation.

The person performing the oil filter replacement could be an engineer designing an automobile while considering ease of maintenance. Replacing the oil filter of a car is a common task performed through the life of a product, and the ease in which it can be accomplished can save both time and money. Design engineers can use virtual environments to accurately simulate such tasks without requiring the production of a costly and time-consuming physical prototype. The individual replacing the oil filter could also be a mechanic training on a new car design or even a car owner learning maintenance techniques and procedures for a personal vehicle using a manual or training site over the internet.

Although the example given is narrow in scope, many other tasks or procedures could be imagined in which haptic feedback may be implemented to improve the realistic nature of virtual environments. Through the oil filter replacement example, an attempt was made to portray the powerful potential of haptic feedback in simulation. Given the relatively young status of haptic technology, the research conducted and presented in this dissertation is an initial step toward this vision. In the next section, a review of literature in automated and virtual environments for assembly and disassembly evaluation is presented.

1.3 Computer Environments for Assembly and Disassembly Analysis – A Literature Review

Thus far in Chapter 1, the motivation for assembly and disassembly analysis was presented and the concept of haptic technology, including current devices, application areas, and a futuristic vision, was introduced. As mentioned, the research conducted in this dissertation aims to develop a virtual environment with force feedback capable of

simulating assembly and disassembly procedures and to test the effectiveness of such an environment. Although limited, there has been some research in using the computer as a tool for automated or virtual assembly and disassembly analysis. In the following sections, a literature survey describing this previous work is provided and the general consensus of the research is highlighted.

1.3.1 Automated Assembly and Disassembly Analysis

Many automated techniques have been developed to determine assembly and disassembly sequences for a product, some addressing the difficulty of each alternative proposed. Thomas and Torras (1988) introduces a system to infer assembly configurations based on three types of constraints: shape-matching constraints, constraints on the degrees-of-freedom, and non-intersection constraints. After determining the constraints between all of the components of a product, possible assembly configurations can be generated. This technique finds achievable assembly configurations for a set of parts, but does not address the required assembly sequence or processes. Srinivasan *et al*. (1997) have developed a software program that performs disassembly analysis on a product and gives recommendations as to the most desirable design based on the disassemblability and design rating of each alternative. More recently, Adams and Whitney (1999) utilizes screw theory to determine the mating constraints between two parts in an assembly based on the geometry of the features that join them. Again, this method does not address assembly or disassembly sequences, rather determines the state of constraint for each part in an assembly.

Since the techniques described above are automated, the designer has minimal interaction with the product being designed. Typically, design engineers desire greater

interaction with a product during design assessment, either through a physical or virtual prototype. This increased interaction could provide additional information to the designer about the processes of assembly and disassembly. For instance, ease of assembly or disassembly requires that the role of the human or machine be accounted for. The level of interaction provided by these automated techniques limits the designer's ability to address these issues.

1.3.2 Virtual Environments for Assembly and Disassembly Evaluation

Given the recent surge in virtual technologies, immersive environments have become popular for many applications. Some researchers have already explored the use of virtual environments for analysis of assembly and disassembly operations. A description of several virtual environments for assembly and disassembly evaluation follows, detailing observations and important findings of the authors.

Inventor Virtual Assembly (IVY) (Kuehne and Oliver 1995) allows a designer to import data from a CAD package for analysis of assembly hierarchies. Since the assembly hierarchies are predefined by the CAD package, the designer is simply visualizing and interacting with a product to determine the validity of a predefined assembly sequence. Interaction with the environment includes a head mounted display and a Cyberglove® that allows the user to grab and move objects. IVY, however, lacks collision detection between virtual objects, modeling of inertia and gravity, and force feedback, citing these as areas for future improvement.

Users are able to realize complete disassembly processes for a product via virtual prototyping in (Siddique and Rosen 1997), implementing automated and interactive techniques. First, automated methods determine the feasible removal direction space for

all parts in an assembly, computed from normal mating directions and mating fits between objects. These reasoning methods generate partial disassembly sequences, which then depend on the judgment and decisions of the designer to complete the process. Additionally, a virtual hand and tools are implemented to test the difficulty of the proposed disassembly processes, but user input is limited to the keyboard and traditional mouse.

Antonishek, et al. (Antonishek *et al.* 1998) describe a virtual environment that uses tracked gloves, shutter glasses, and two-handed gestures to interact with and verify product assemblies on a virtual workbench. Minimal collision detection was implemented using simplified bounding box intersection testing to determine if the user's hand was touching an object or if objects were colliding. Users were supplied with auditory and visual feedback, but the authors note that haptic feedback would add to the realism of an assembly simulation.

An approach for virtual assembly combining physically based modeling and constrained is discussed in (Wang *et al.* 2001). The method concentrates on the dynamic behavior of parts in three situations: free motion in space, constrained linear motion on a plane or along an axis, and constrained rotation about an axis. The simulation handles each of these situations separately. For the simulation of dynamic parts, the authors note that acceleration due to gravity needs to be scaled down to achieve a realistic feeling.

A constraint-based virtual environment for interactive assembly and maintenance tasks is presented (Fernando *et al.* 1999; Fernando *et al.* 1999). Constraint management techniques support the constraint-based interaction between parts to ensure realistic behavior of assembly components.

Johnson and Vance implement the Voxmap PointShell (VPS) algorithm for virtual assembly planning in Virtual Environment for General Assembly (VEGAS) (Johnson and Vance 2001). Within VEGAS, which allows only one dynamic object at a time, they demonstrate the assembly of a parking brake into a cab. A position-tracked wand, along with buttons to send the commands grab, release, and assemble, is used to interact with objects within the simulation. Feedback is limited to a color variation of colliding objects in the virtual environment, as a dynamic object colliding with static object turns red. The authors note, "VPS was not designed to provide high accuracy collision detection" and "if the direction of the research was to evaluate the fit of the parts together, a more exact [collision detection] method would be necessary, however this application is focused on the human interaction within a fully immersive environment" (Johnson and Vance 2001).

The advent of virtual reality technology has allowed great advancement in the simulation of assembly and disassembly through virtual environments, allowing the analysis of assembly processes and sequences, not just testing assembly configurations, which limited previous automated techniques. However, two prevailing themes originate from the above research: the need for haptics and the need for a physically based simulation for assembly and disassembly evaluation, each leading to a more realistic environment.

1.3.3 Integration of Haptics in Virtual Assembly and Disassembly

Using haptic technology, researches have recently developed virtual environments for assembly and disassembly simulation that provide feedback to the user through the

sense of touch. The environments developed for this purpose and with haptic capability are discussed in this section.

Gutiérrez, et al. (Gutierrez *et al.* 1998) describes an assembly simulation environment composed of a PHANToM haptic interface combined with DATum, a geometric modeling system. DATum uses a hybrid representation for each object containing constructive solid geometry and boundary representation information, attempting to utilize the advantages of each type. The environment provides object manipulation capabilities such as touch, grasp, and move using one PHANToM to provide force feedback. Interaction between objects, however, is limited as the collision detection technique utilized provides only contact status, making realistic collision response impossible.

Gupta et al. (Gupta and Zeltzer 1995; Gupta *et al.* 1997) developed the Virtual Environment for Design for Assembly and tested its effectiveness for a 2-dimensional peg-in-hole insertion experiment using a dual-PHANToM setup. This was the first time that haptics were used to allow realistic interaction with a dynamic object in a virtual assembly environment, providing the ability to grasp an object using two fingers. The results demonstrate that force feedback through the haptic interfaces reduced the time needed for assembly completion and reduced the number of handling errors occurring during the task. On the other hand, the authors found that 3D visualization and sound feedback had little effect on the time required and the number of errors committed during the assembly. This research was limited in that only 2D planar motion of objects was permitted. To allow analysis of real-world assembly and disassembly tasks, objects must be allowed 3D motion.

Another virtual environment (Pere *et al.* 1996) developed to test mechanical assemblies uses the Rutgers Master II glove, which provides force feedback to four fingertips on one hand. This simulation is not limited to local assembly procedures, as it also involves navigation through a large virtual environment using hand gestures. Grasping of virtual objects is automated using a point and touch technique. The major limitation of this environment is that force feedback is not provided to the hand as a whole. For instance, if a grasped object passes through another object, its position is simply reset to the previous position and the user's hand is free to keep moving.

A constraint-based environment has been implemented in (Gomes de Sa and Zachmann 1999) to investigate the steps needed to accurately simulate assembly and maintenance. The most important results of this research come from a user survey of the virtual environment. Voice input was preferred over 3D menus and the keyboard as the method for giving commands to the simulation. For the purposes of assembly, vibro-tactile feedback alone was considered an unnatural feeling. It is noted that force feedback would've been much more desirable, without which some assembly tasks are almost impossible.

Jayaram et al. (Jayaram *et al.* 1999; Jayaram *et al.* 1999) developed the Virtual Assembly Design Environment (VADE) to explore the potential use of a virtual environment for assembly planning and evaluation. VADE simulates most part interaction using constrained motions, implementing data obtained from the CAD system that was used to design the part. By using constrained motions, the developers bypass the need for computationally expensive collision detection algorithms. In VADE, however, real-time collision detection is an option for interaction between parts that are not

constrained. VADE supports one-handed and two-handed operation (although only one hand can be dexterous) and provides only vibro-tactile feedback. The main limitation of VADE is that it lacks kinesthetic force feedback, which may add to the effectiveness of an assembly and maintenance simulation.

1.3.4 Literature Review Summary

The literature review conducted and described above highlight several important aspects when developing a virtual environment for assembly and disassembly.

- First, automated techniques have been used to infer assembly configurations, mating constraints between parts, and assembly or disassembly sequences. However, such algorithms lack the capability to model the human or machine performing the assembly and limit the degree of interaction of the designer.
- Modeling of interactions between objects in a virtual environment is cited as a desired trait for a virtual assembly and disassembly evaluation environment, adding to simulation realism.
- Most researchers found haptic feedback useful, though some suggest that kinesthetic force feedback is more realistic and beneficial than vibro-tactile feedback.
- Grasping capability and grounded force feedback to the motion of the hand and arm are important for virtual assembly and disassembly analysis. Otherwise, a user's virtual representation may travel through walls or other immovable objects.

Aside from these findings, there has been a limited amount of experimental research to actually test the effectiveness of haptic feedback in these haptic assembly and disassembly environments. In the next section, the focus of the research in this

dissertation will be presented, along with development of the research questions and hypotheses.

1.4 RESEARCH FOCUS IN THE DISSERTATION

Based on this literature review, previous development of our simulation (to be discussed further in Chapter 2), and the overall goals of this research, two main areas of interest were investigated. First, the modeling object interactions between virtual objects need to be improved, such simulation failure due to improper inter-object penetrations are reduced to the point that allows meaningful experiments to be run using the simulation. Upon completion of the work involved with improved object modeling, volunteers will complete a set of experiments designed to determine the improvement provided by haptic feedback in the virtual environment over a purely visual simulation.

The research questions relating to these goals are presented in Section 1.4.1 along with the corresponding research hypotheses. The tasks required to address the research questions and hypotheses are discussed in Section 1.4.2. The resulting research contributions are outlined in Section 1.4.3.

1.4.1 The Principal Goal, Research Questions and Hypotheses

As previously mentioned, the overall goal of the research conducted for this dissertation aims to determine the effectiveness of haptic feedback for simulation of assembly and disassembly procedures. In an effort to address this principal goal, two primary research questions were developed and are described below.

The development of a real-time haptic simulation for virtual assembly and disassembly requires not only haptic interaction with objects, but also modeling the interaction between objects, each of which is a computationally expensive process.

Previous development of our virtual environment included minimization of the computational load required by the haptic loop (McDermott 1999; McDermott and Bras 1999; Coutee *et al.* 2001). The techniques developed allow more efficient haptic interaction with complex object geometries and the simulation of a large number of dynamic objects simultaneously. Similarly, we wish to explore and integrate techniques into the simulation that improve the interaction between virtual objects, creating a more robust and usable simulation. We feel this is a logical and necessary step prior to addressing the usefulness of the virtual environment as an assembly and disassembly evaluation tool. This desire leads us to answer the following research question.

> ***Question 1:*** *How can collision detection between arbitrary non-convex objects be improved to support the dynamic real-time simulation of such objects for haptic assembly and disassembly evaluation?*

Specifically, I propose to test the following hypothesis:

> ***Hypothesis 1:*** *Significant improvements can be made over current collision detection libraries for the interaction between arbitrary non-convex objects for dynamic real-time simulation, particularly for haptic assembly and disassembly evaluation.*

The research question and hypothesis above encompass both improvements in current public domain software packages for collision detection and supplemental techniques to support these packages for object interaction. To assist in addressing this research question, three sub-questions will be utilized. The supporting questions and corresponding hypotheses are listed below.

Question 1.1: *Do current collision detection libraries provide the usability and functionality necessary to efficiently handle dynamic object interactions, and, if not, how can they be improved?*

Hypothesis 1.1: *Public domain software libraries for collision detection provide the usability and functionality to simulate dynamic object interactions, while there are areas for improvement.*

Question 1.2: *How can current collision detection libraries be supplemented to improve object interactions within HIDRA, while limiting detrimental effects and preserving the realistic quality of the simulation?*

Hypothesis 1.2: *Techniques such as constraint integration, velocity slowdown, and others can be implemented to improve object interactions in HIDRA.*

Question 1.3: *How can a collision detection library be integrated with the HIDRA simulation to maximize the collision performance?*

Hypothesis 1.3: *Programming HIDRA using threads offers the most flexibility and maximizes performance for integration of a collision detection library.*

The work completed to address these research questions and hypotheses is presented in Chapters 3 and 4.

Upon successful completion and acceptance of the first hypothesis, we will be ready to test the benefit of haptic interaction for assembly and disassembly evaluation using the simulation developed. Therefore, the second research question bears a great deal of resemblance to the overall goal of this research. The second research question and hypothesis follow.

> ***Question 2:*** *Does a haptically enabled simulation environment provide a significant improvement over a purely visual simulation for assembly and disassembly evaluation and related engineering issues?*
>
> ***Hypothesis 2:*** *A haptically enabled simulation environment does provide a significant improvement over a purely visual simulation for assembly and disassembly evaluation and related engineering issues.*

As with the first research question, Research Question 2 is composed of three supporting questions. These sub-questions and corresponding hypotheses are listed below.

> ***Question 2.1:*** *How does the perception of weight in a virtual environment with haptic feedback compare to that in the real environment?*
>
> ***Hypothesis 2.1:*** *The perception of weight and ability to distinguish between weights in a virtual environment closely matches that in the real environment.*

> ***Question 2.2:*** *How does haptic feedback affect the ability of a user to detect motion tolerances in a virtual environment?*

Hypothesis 2.2: *In a virtual environment, haptic feedback improves a user's ability to detect differences in the range of motion of an object when compared to a purely visual simulation.*

Question 2.3: *Does haptic feedback provide a significant improvement over a purely visual simulation for assembly and disassembly operations?*

Hypothesis 2.3: *Haptic feedback can provide significant improvements over a purely visual simulation for assembly and disassembly operations by reducing task completion time.*

Experiments were completed to test each hypothesis above using human subject volunteers. The details, results, and analysis of these experiments are discussed in Chapters 5 and 6. In the next section, the strategies implemented to test the research hypotheses are outline, including discussion of the tasks performed.

The relation between each research question and the sections in this dissertation that directly address the question is shown in Table 1.1. As shown, components of the response to the first research question are covered over several sections in Chapters 2, 3, and 4. The experiments conducted to answer the second research question are encompassed entirely in Chapters 5 and 6.

Table 1.1: Relations Between Research Questions and Dissertation Sections

Question		Dissertation Section(s)
RQ 1.1	Do current collision detection libraries provide the usability and functionality necessary to efficiently handle dynamic object interactions, and, if not, how can they be improved?	Sections 3.2.3, 3.2.4, 3.3.2, 4.4
RQ 1.2	How can current collision detection libraries be supplemented to improve object interactions within HIDRA, while limiting detrimental effects and preserving the realistic quality of the simulation?	Sections 2.4.1, 4.1, 4.2, 4.4
RQ 1.3	How can a collision detection library be integrated with the HIDRA simulation to maximize the collision performance?	Section 4.3
RQ 2.1	How does the perception of weight in a virtual environment with haptic feedback compare to that in the real environment?	Section 5.2
RQ 2.2	How does haptic feedback affect the ability of a user to detect motion tolerances in a virtual environment?	Section 5.3
RQ 2.3	Does haptic feedback provide a significant improvement over a purely visual simulation for assembly and disassembly operations?	Sections 6.2-6.5

1.4.2 Strategies for Addressing the Research Hypotheses

As will be discussed, the main tool for modeling object interactions is through specialized collision detection software libraries. The focus for improving collision detection libraries (Research Question 1.1) will be in their usability rather than the techniques used to find the minimum distance or intersection status between two objects, as other research groups concentrate entirely on this subject. Current collision detection libraries, while a valuable and indispensable tool for simulation of dynamic bodies, have shortcomings that limit the effectiveness of real-time simulation incorporating haptic feedback. Such shortcomings include the inability to effectively handle non-convex

objects, penetrating object pairs, and objects in close proximity to one another. Though much of these limitations can be solved with faster computers and greater computational capability, this work will identify areas where these libraries and the information provided can be more beneficial, particularly in a real-time haptic simulation of arbitrary non-convex objects. In addition, this dissertation discusses the design, implementation, and testing of a number of supplemental techniques that work in combination with a collision detection library to enhance object interactions (Research Question 1.2). These techniques will be designed to alleviate fatal interaction sequences such as excessive object overlap that sometimes occurs with the use of collision detection libraries alone, leading to a more robust simulation. To answer Research Question 1.1, the following tasks were performed.

1. Conduct a preliminary evaluation of current collision detection libraries to determine which are most suitable for use in real-time haptic simulation of dynamic objects (Section 3.2).
2. Implement candidate collision detection libraries (Section 3.3) and evaluate performance (Sections 4.3 and 4.4).
3. Identify areas where candidate libraries fail or desire improvement, either through usability or functionality. How easy is the library to use? What additional information and features are desired? What are the advantages and disadvantages of libraries that have different implementations (Sections 3.2.4 and 4.4)?

Although many collision detection libraries can detect intersections and compute distances between two objects, there are often failures that occur due to the demands of real-time haptic simulation. For this reason, Research Question 2 is geared toward

identifying techniques that can be implemented in simulation to supplement a collision detection library and improve object interactions. To accomplish this goal, the following tasks were conducted.

1. Identify techniques that could improve the speed, robustness, and/or reality of object interaction. The generation of ideas for these techniques comes through literature surveys and prior experience with our simulation, particularly the methods of failure with respect to object interaction.
2. Implement each technique, preferably with the option to use or not use during a particular session of the simulation. This will allow for simplified testing as described next.
3. Determine the usefulness of each technique. Is there an improvement? Does each technique provide an advantage over not using it? Are there disadvantages associated with each technique? If so, are they more significant than the advantages?

How do I propose to test each technique, determining its usefulness? The techniques implemented are tested by performing a simple assembly (see Section 5.4) and comparing the results to previous studies in completing the same task. As will be shown, the integration of the supplemental techniques developed provide a drastic improvement in the reliability of virtual object interactions.

In addition to these tasks, the method of integrating a collision detection library into a virtual simulation environment, particularly with respect to the high-level simulation architecture, will be scrutinized (Research Question 1.3). To accomplish this, several versions of our simulation were created, each with a slightly different simulation

layout for incorporating collision detection. The advantage of certain integration and organization techniques for implementing these libraries will be portrayed through a simple assembly example.

The main focus of Research Question 2 is to assess the viability of haptic feedback for assembly and disassembly analysis in a virtual environment. Upon completion of the work performed to answer the first research question, experiments were conducted using our simulation and human subject volunteers as users. In total, five experiment sets were designed to address the second research question and its sub-questions. These include a weight sensation experiment, a motion tolerance study, a peg-in-hole assembly, a toy train track assembly, and replacement of an automotive break pad. Using these experiments as case studies, the usefulness of haptic feedback, particularly kinesthetic force feedback, in our virtual environment for assembly and disassembly was tested. The details, results, and analysis of these experiments are discussed in Chapters 5 and 6.

As required, approval from the Georgia Tech Institutional Review Board[1] was obtained prior to running any experiments using human subject volunteers.

1.4.3 Contributions from the Research

The following contributions are a direct result of the research presented in this dissertation.

- *A Haptic Assembly and Disassembly Simulation (useful for experimentation)* – Our simulation is the only known real-time assembly and disassembly simulation for arbitrary 3-dimensional objects that combines two PHANToM haptic

[1] Website: http://www.osp.gatech.edu/compliance/humans/humans.htm

interfaces and high precision collision detection and object response. The two-PHANToM setup allows the grasping of virtual objects rather than merely touch feedback. This provides a more realistic user interface than other simulations, in which the motion of a single haptic device guides object motions directly. In addition, the simulation described provides real-time collision detection and realistic physical modeling of object responses, a feature not entirely common among the virtual environments for assembly and disassembly analysis surveyed in Section 1.3.

- *Improvements in Object Modeling* – Object interactions will combine the strengths of impulse-based modeling (see Section 3.1) and a constraint maintenance technique to be discussed in Section 4.1. The need for such a hybrid simulation combining the strengths of constraint- and impulse-base modeling techniques was discussed in (Mirtich 1995). The constraint maintenance scheme working in tandem with the collision detection library will handle disjoint objects and those in assembled configurations with fewer intersections that lead to simulation failure.
- *Techniques to Support Collision Detection* – In this dissertation, a number of supplemental techniques designed to provide more reliable object interactions in situations where a collision detection library alone fails were implemented. As will be shown, the modes of failure for a collision detection library in real-time simulation result from strict computational requirements, the demands of haptic feedback, and the non-convex nature of objects in an assembly.

Using our simulation as a research tool, this work contributes to the research knowledge base for assembly and disassembly simulation in the following ways.

- Knowledge about collision detection libraries from the user's standpoint of usability and functionality will provide researchers in that community an outside view of how their product may be improved. In addition, we expose the detrimental effects of haptic interaction on ability to effectively simulate object interactions, particularly the task of detection collisions.
- The experimental studies described, provide direct evidence for the effectiveness of haptic feedback in simulation over a purely visual simulation, particularly for the analysis of assembly and disassembly operations, and in what ways it provides an advantage. Insight is gained as to whether a virtual environment with haptic feedback can possibly be a viable alternative to building physical prototypes.
- A user survey provides direct feedback from users involved in the experimental studies as to their reactions to haptic feedback and the virtual environment, including their likes and dislikes. This may help predict the reaction of practicing engineers to such a tool as all experimental subjects will be engineering students. User comfort with the PHANToM interface will also be addressed through the user survey.
- Through this work, knowledge will be gained about current haptic devices, particularly the PHANToM, including the limitations and how such devices can be improved. The advantages and disadvantages of two PHANToMs in a dual configuration will be explored.

Clearly, all of the findings will be generated from studies using our virtual environment, which is designed for assembly and disassembly simulation. However, the conclusions from this work will be applicable across a much wider area of research, not just haptic assembly and disassembly analysis.

1.5 ORGANIZATION OF DISSERTATION

In this chapter, the motivation for use of haptic technology in virtual environments for assembly and disassembly was described. In addition, a review of current research in the field was presented. This led to the development of our principal research goal and the research questions for this dissertation. The remaining chapters in this dissertation discuss the work and experiments conducted to address and answer these research questions. A general overview of the content of each chapter is provided below.

As mentioned, we have developed a virtual environment for assembly and disassembly simulation with the intention of evaluating the effectiveness and usefulness of haptic feedback. In Chapter 2, I will provide an overview of this simulation environment. Topics covered include the previous work in the development of the simulation, including construction of virtual objects via CAD transfer, integration of collision detection and response algorithms, reduction of computational load on the haptic process, and demonstration of the simulation capabilities. Also to be discussed are general simulation enhancements that improve user interaction with the simulation. Finally, the major limiting components of the simulation will be highlighted, providing partial motivation for Research Question 1.

In Chapter 3, the realm of virtual object interactions will be presented, beginning with physically based modeling in simulation. Based on this review of modeling

techniques, collision detection becomes an important component of our simulation. The major aspects of collision detection, minimum distance computation and bounding volumes, are highlighted, followed by a review of publicly available software packages for collision detection (partially answering Research Question 1.1). Lastly, important details of collision detection and response integration with our simulation are discussed. In this chapter, the work required to answer first research question is initialized.

Based on the findings from Chapter 3, Chapter 4 is dedicated to the development of supplemental techniques that can be integrated with these collision detection libraries for improved object interactions. Within Chapter 4, methods for high-level integration of collision detection libraries in our simulation will be reviewed and tested (Research Question 1.3). Additionally, the supplemental techniques will be tested using a simple assembly example (Research Question 1.2). In the process, additional knowledge will be gained concerning the usability of the collision detection libraries discussed in the previous chapter (Research Question 1.1).

Beginning with Chapter 5, the focus of the dissertation will be redirected toward Research Question 2. Two experiments completed by a number of human subject volunteers, as users of the simulation, will be described in detail. The first experiment is a study on weight sensation in the virtual environment, when compared to that in real life (Research Question 2.1). The second experiment will determine the value of haptic feedback for determining motion tolerance in the virtual environment (Research Question 2.2). Each of these experiments will shed light on the capabilities provided by haptic feedback in virtual environments.

In Chapter 6, the human subject experiments continue as the details and results of three more experiments, focused on assembly and disassembly, are presented. All experiments in this chapter are designed to assess the usefulness of haptic feedback for assembly and disassembly as compared to a purely visual simulation. The experiments include a peg-in-hole insertion, a model train track assembly, and a simulated automotive brake pad replacement. An analysis of the users' performance in these experiments will allow us to answer Research Question 2.3.

Finally, in Chapter 7, a discussion and summary of the dissertation will be provided, including the answers to the research questions. The achievements and contributions are highlighted, a critical analysis including limitations of the work is presented, and ideas for future work are provided.

In Figure 1.2, a roadmap for the thesis is provided, linking the research questions to the sections of the dissertation in which they are discussed. In this image, there are two main subsections within the main body of the dissertation, corresponding to the two main research questions. The response to Research Question 1 will be formulated and discussed in the next three chapters. Upon completion of the simulation work to improve object modeling, the newly enhanced simulation will provide the platform to perform a series of experiments in conjunction with Research Question 2. Finally, as previously discussed, the dissertation will close with a discussion of all achievements and recommendations for future work. The pictorial map, proceeds from bottom to top beginning with the research motivation provided in this chapter.

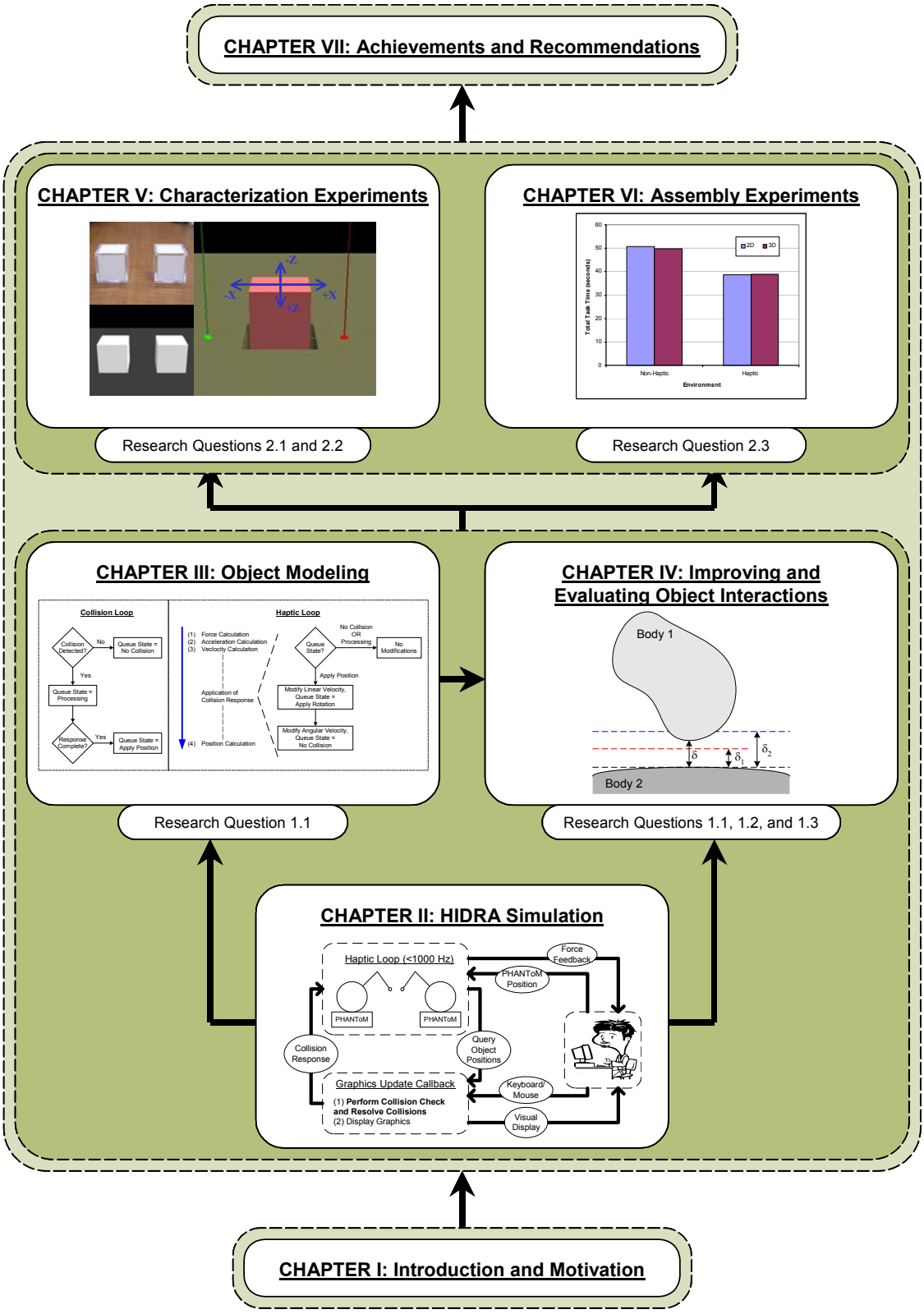
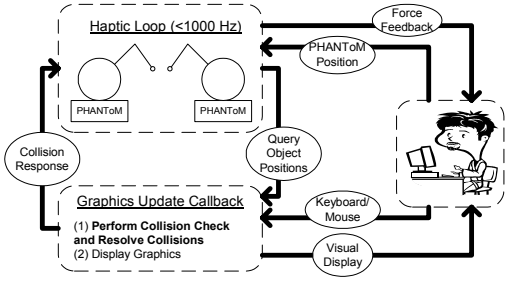


Figure 1.2: Dissertation Roadmap

CHAPTER II

HIDRA – A HAPTIC ENABLED ASSEMBLY AND DISASSEMBLY SIMULATION



The overall goal of this research is to determine the usefulness of force feedback for the simulation of assembly and disassembly procedures. The literature review in the previous chapter mentioned several environments designed for a similar purpose, and most researchers found haptic feedback useful for tasks in a virtual environment. However, most of these environments lack physically based modeling, which may be used to realistically simulate the motion and interaction between objects. During this and previous research, the HIDRA simulation test-bed, a haptically enabled virtual environment for assembly and disassembly analysis, has been developed. The advantages of HIDRA, when compared to the other environments, are that it incorporates both physically based modeling and a haptic setup that allows the user to grasp objects, similar to grasping a physical object. The chapter starts with an overview of HIDRA (Section 2.1), followed by a brief discussion of the previous work conducted during the initial

design and testing of the simulation (Section 2.2). In Section 2.3, general enhancements for improved user interaction and visualization are presented. Some limiting components of HIDRA are discussed in Section 2.4, which provide partial motivation for some of the research questions in this dissertation. A chapter review is provided in Section 2.5.

2.1 AN OVERVIEW OF HIDRA

HIDRA is a test-bed application developed to test the feasibility and usefulness of force feedback for the simulation of assembly and disassembly operations. HIDRA was initially created through the graduate work and thesis of McDermott (McDermott 1999). Prior to discussing this previous work, a general overview of the HIDRA simulation is provided.

HIDRA is designed around two Model 1.5A PHANToM® haptic interfaces and the GHOST® SDK Version 3.0 from SensAble Technologies, Inc.[1], which provides functions to handle all interactions with the PHANToM devices. Each PHANToM device is a 3-DOF linkage capable of providing a force as output to the user. The implementation of both devices simultaneously in a dual-PHANToM setup allows grasping through a virtual index finger and thumb. The setup of the HIDRA simulation with the PHANToM devices is shown in Figure 2.1.

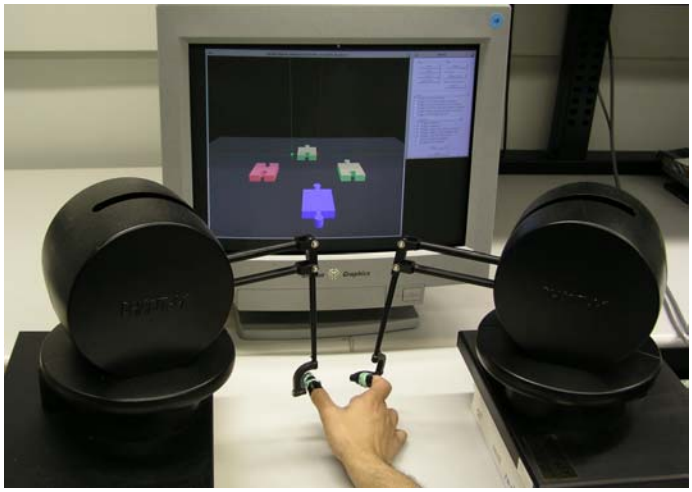


Figure 2.1: Dual PHANToM Configuration with HIDRA

[1] See website for more information: http://www.sensable.com

The graphics are created using OpenGL®1.1[2] and the graphical user interface is written in GLUI, a library built upon the GLUT OpenGL library that provides menu capability. An image of the main menu of HIDRA is shown in Figure 2.2, and many of the options available to the user are visible. The GLUI library provides advanced menu capabilities through a platform independent API. In addition, all menus are separate from the main graphical window, which allows interaction with both at the same time. This is beneficial by allowing modification of the available options during the uninterrupted real-time simulation. For instance, a user can set motion constraints on an object while manipulating it through the haptic interface.

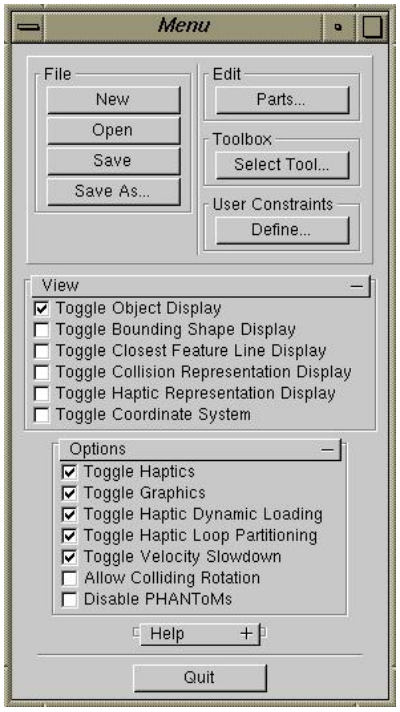


Figure 2.2: HIDRA Main Menu using GLUI

[2] OpenGL is a registered trademark of Silicon Graphics Incorporated.

Various specialty libraries, particularly for collision detection (see Chapter 3), were also used in the simulation. All additional programming needed to create HIDRA was accomplished using the C++ programming language. All components of the simulation run on an SGI Octane MXE with two 250 MHz processors.

The main functions of HIDRA can be broken down into three components: haptic interaction with the virtual environment, modeling the interaction of virtual objects, and graphic display. A graphical depiction showing the interaction between these three components and the user is shown in Figure 2.3.

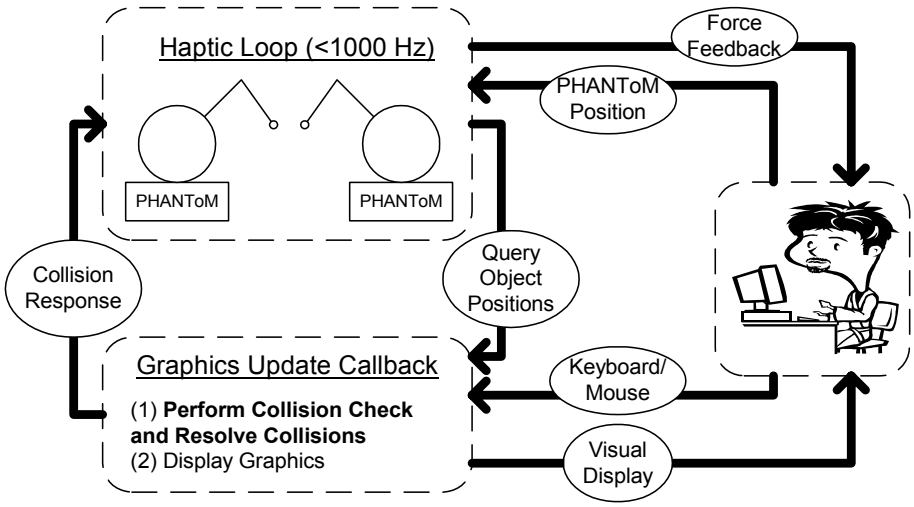


Figure 2.3: High-Level Simulation Architecture

As shown in Figure 2.3, the haptic loop monitors the interaction of the user through the dual-PHANToM interface with all haptic objects in the simulation. The haptic loop, upon initialization of the haptic scene and PHANToM interfaces, runs on a continual basis and requires the highest priority for the computer's resources. In order to create the realistic sensation of touch and as required by the PHANToMs, the haptic loop

of HIDRA must update with a frequency of 1000 Hz (Massie 1996). Therefore, the dynamic state of all objects in the simulation, to include the associated positions, velocities, and accelerations, must be maintained by the haptic loop. As a result, the graphics and virtual object interaction routines must run as secondary processes.

Next, we turn our attention to the graphical display of and interaction between virtual objects in the simulation. A requirement of the GLUT window management system of OpenGL is that upon entering GLUT's main graphical loop, the routine will never return control to the calling program. As a result, all routines that are executed from that point forth must be carried out through the various callback routines available through GLUT. HIDRA accomplishes this task through a graphics update request using a GLUT callback function. During this update, two processes are performed. The first step during a graphics callback is the detection and response to inter-object collisions, otherwise known as the collision loop. Upon completion of all collision processes, the collision loop stores object responses in a queue for later processing by the haptic loop (to be discussed further in Section 3.3.3). Finally, the graphic display of each virtual object is created using current object positions. During the simulation, various procedures also require input from the mouse and keyboard.

It should be noted that the simulation architecture depicted above originated during previous work on HIDRA. As will be discussed in Sections 2.4.3 and 4.3, this organizational structure has limitations and was modified during the course of this research. Although significant changes were made, the function of the haptic loop, collision loop, and graphic display remain the same.

2.2 PRIOR ACCOMPLISHMENTS WITH HIDRA

In this section, the work accomplished by McDermott will be discussed, including creation of virtual objects from CAD representations, integrating collision detection and response algorithms, reducing computational demand of the haptic devices, and demonstration of simulation capabilities. The purpose of this section is to describe the focus of previous work in the development of HIDRA prior to the research described in the remainder of this document. Refer to McDermott's thesis for more information (McDermott 1999).

2.2.1 Construction of Virtual Objects via CAD Transfer

An initial and continuing requirement of HIDRA is that it can be seamlessly incorporated into existing design processes. As such, the creation of virtual objects for HIDRA should incorporate any models developed in the process. In today's design processes, engineers typically utilize commercial CAD packages to model product designs. As such, it is beneficial to obtain design information, particularly object shape and properties, directly from a CAD representation.

Most software libraries, including GHOST and OpenGL, cannot directly read information from a file native to a particular CAD package. For this reason, a neutral file format was desired, and VRML format was chosen, which can be exported by all major packages. These VRML files typically contain a tessellated representation of the object. The original intent was that the simulation could interpret all VRML object files, however it was found that the file format differed somewhat between CAD packages. Therefore, a parser was written to decipher VRML 1.0 as exported from Pro/Engineer modeling software.

In HIDRA, all data obtained from the VRML file(s) for an object are stored in a database, which contains vertex, edge, and face information and their relations. Using this information, all representations of simulation objects can be constructed: haptic, collision, and graphic.

2.2.2 Integration of Collision Detection and Response Algorithms

HIDRA's collision loop satisfies two major functions: detection of collisions between two virtual objects and generation of each object's response to these collisions. Much of McDermott's work focused on understanding and integrating the response algorithm to handle collisions between virtual objects (McDermott 1999). The theory and implementation of the response algorithm used was described in the dissertation of Mirtich (Mirtich 1996). In addition, McDermott conducted a survey of collision detection techniques and a qualitative comparison of collision libraries.

In this research, a more thorough qualitative comparison between these libraries is conducted, including more recent collision detection libraries. Based on this comparison, a quantitative study comparing the performance of the two most desirable libraries, as implemented in HIDRA, was carried out. The results, presented in Chapter 4, highlight additional limitations and benefits of each library for use in a real-time simulation. In addition, the effect of high-level simulation architecture on the performance of these collision detection libraries was studied, leading to the restructuring of HIDRA. Details about the collision detection process, the layout of the most recent collision loop, and the qualitative and quantitative analysis are described in Chapters 3 and 4.

2.2.3 Reduction of Computational Load on Haptic Loop

As mentioned, the haptic loop demands a very fast update frequency of 1000 Hz. In fact, update rates lower than this can introduce instabilities, resulting in buzzing and jolting of the PHANToM devices. As a safety precaution, for the haptic device and the user, the haptic process will cease operations if computational time exceeds the 1-millisecond limit.

The computational load placed on the haptic loop is magnified when dynamic objects are present in the simulation. In preliminary tests performed, the simulation of relatively simple tasks involving dynamic objects such as placing a ring on a shaft frequently required more than 1 millisecond. As a result, several optimization techniques were developed, each designed to reduce the computational load on the haptic loop.

The first, known as haptic dynamic loading, reduced computation time by removing objects from the haptic scene that are not close enough for the PHANToM virtual fingertip to touch. This was accomplished by placing a bounding box around each fingertip and checking for intersections with the virtual objects in the scene. If neither bounding box intersected an object, its haptic representation was unloaded and the haptic process ignored the object. This essentially reduced the number of objects the haptic loop had to process and decreased computation time. The intersection detection was accomplished using a collision detection algorithm, which will be discussed in more detail in Chapter 4.

Another method to reduce computation load on the haptic loop was to anchor objects. As expected, a static object requires fewer calculations than a dynamic object during the haptic process.

As discussed in Section 2.2.1, the haptic representation of an object was generated using tessellated geometry. In certain cases, objects can be partially or completely composed of primitives, which provide a simpler representation and typically more efficient calculations. For example, a sphere is much more effectively represented as the sphere primitive rather than a tessellated sphere. The GHOST library contains haptic primitive representations for just this purpose. Therefore, the capability to swap tessellated geometry with primitives was incorporated into HIDRA. Again, this resulted in decreased computation time.

For dynamic objects, the simulation must define the equations of motion, as GHOST does not do this. Initially, forces, accelerations, velocities, and positions were calculated every iteration of the haptic loop. However, it was determined that these calculations were very time consuming and unnecessary. Although the demand of the haptic process is an update frequency of 1000 Hz, the human ability to detect motion in our limbs has a maximum bandwidth of around 10 Hz (Srinivasan and Basdogan 1997). Therefore, the calculations to compute object position could be partitioned over several haptic loop iterations. In HIDRA, the computation of the equations of motion is partitioned over ten iterations of the haptic loop, resulting in decreased computation time per loop. This technique is known as haptic loop partitioning.

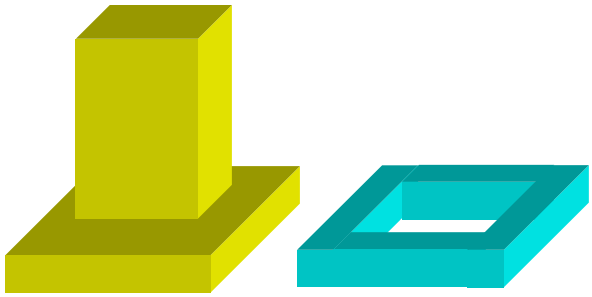


Figure 2.4: Simple Ring and Shaft Example. Taken from (McDermott 1999).

To demonstrate the effectiveness of these optimization techniques, a simple assembly task was run using the ring and shaft shown in Figure 2.4. The haptic loop processing times were recorded during the assembly scenario for various combinations of the techniques described above. Without any optimization techniques in place, the 1-millisecond threshold of the haptic loop was surpassed, whereas using the techniques resulted in a maximum update time of around 0.5 milliseconds, a significant improvement. This same example, assembling the ring and shaft, will be utilized in Chapter 4.

2.2.4 Demonstration of Simulation Capabilities

Two examples were presented to demonstrate HIDRA's capabilities. The first was the simulation of a relatively large number of dynamic objects. In this example, seventeen parts, consisting of spheres, cubes, bolts, and nuts, were scatter throughout the virtual workspace and allowed to fall to the floor (see Figure 2.5). As the user moved the virtual fingertips through the environment, interacting with multiple objects at a time, a noticeable lag was present and at times the graphics became jumpy. However, throughout the scenario the simulation continued to function without catastrophic failure. It was

noted that this type of component interaction, where many objects are in motion at once, is seldom seen in an assembly or disassembly operation.

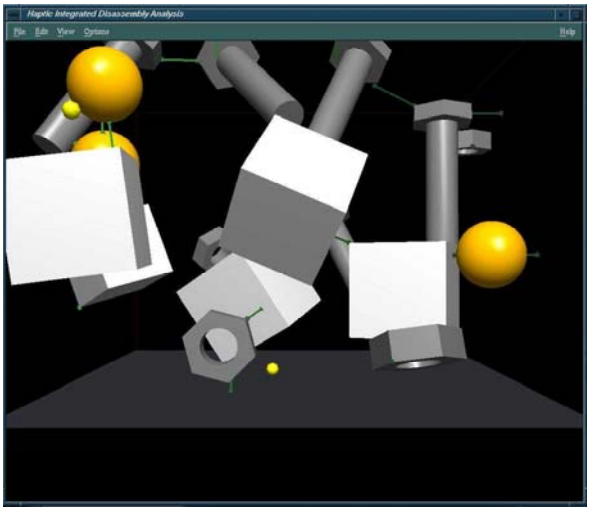


Figure 2.5: Large Number of Objects. Taken from (McDermott 1999).

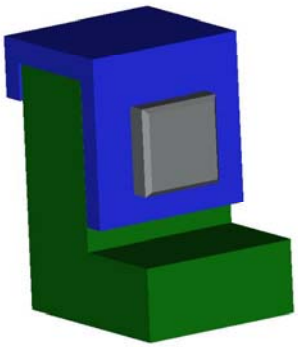


Figure 2.6: Small-scale Assembly. Taken from (McDermott 1999).

In the second example, a small-scale assembly operation was performed. In this task the top component was placed on top of the base, and the bolt inserted through a hole in both (see Figure 2.6). Both assembly and disassembly of the parts were carried out.

Disassembly was successful as long as care was taken not to exert too much force on the parts. Otherwise, penetration would occur and the objects would react erratically. The reassembly was a more difficult task. In fact, successful assembly required that the top piece be anchored so that the bolt could be inserted. In both assembly and disassembly, the base component was anchored. Bolt insertion was the most difficult step in assembly, and trials most often resulted in failure, with the bolt being ejected from the assembly. In later chapters, improved object interactions in HIDRA and more complex examples will be demonstrated.

2.3 GENERAL ENHANCEMENTS TO HIDRA

The majority of the simulation modifications and improvements were in the realm of object interactions. However, several significant enhancements were made to HIDRA that improve the user interface of the simulation. These improvements are discussed in the next three sections.

2.3.1 Simulation Fly-through Capability

In the previous version of HIDRA, the user was limited to visualization of the scene from a single viewpoint. In essence, the user sat in a virtual chair in front of the workspace and all manipulations of virtual objects were carried out from this virtual position. There are a number of obvious limitations in maintaining a static viewpoint. First, certain objects or component features may not be visible from a given view. Allowing the user to rotate around the virtual workspace overcomes this dilemma, permitting an unobstructed view from all sides of an assembly.

With respect to assembly and disassembly analysis, certain perspectives or viewing angles may assist the user in fitting two pieces together. For example, consider

the ring and shaft in Figure 2.4. The task of placing the ring on the shaft can be very difficult if the user is permitted only a side view of the two objects. The task becomes much easier when the objects can be viewed from the top, allowing more straightforward alignment of the components. In addition, a static viewpoint limits the interaction a user can have with each piece through the dual-PHANToM setup. Object manipulations are easiest by grabbing the virtual objects on the sides, whereas the ability to grab objects on the front and back or at an angle is more difficult and sometimes unnatural. Ideally, the user could move around the virtual workspace, making the desired object manipulations more natural.

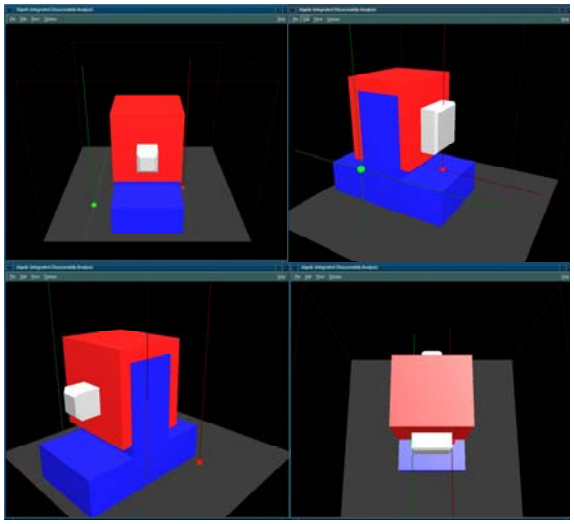


Figure 2.7: Multiple Views of an Assembly using Fly-through Capability

For the reasons described above, the ability to control the user's position and viewing angle has been integrated into HIDRA. Essentially, the user can fly through the virtual environment and view the workspace from any position and angle, facilitating assembly and disassembly sequences. In Figure 2.7, an example of the need for this fly-

through capability is shown. Given the initial view (upper-left), the user cannot create a definitive mental picture of this basic assembly. In addition, the ability to reposition the user's viewpoint clearly facilitates disassembly of these components.

There are many ways in which user position can be controlled. These may include movement commands through speech recognition, mouse input, or keyboard control. In HIDRA, the viewing angle and position are controlled using the arrows keys on the keyboard. This method allows the user to manipulate objects through the dual-PHANToM interface with one hand and change viewing angle with the other.

2.3.2 3D Positioning Capability in the Virtual Environment

Virtual environments enable more realistic viewing and interaction with virtual representations of objects or scenes in real life. The initial version of HIDRA, as mentioned, created this visual scene using OpenGL, but was limited to viewing the 3D environment on a 2D plane, the computer monitor. As a consequence, depth perception was difficult and limited to visual cues such as the size of an object or whether one object passed in front of the other. This impacted the ability to grasp an object, as the user would sometimes reach behind or in front of the object

In an effort to assist the user of HIDRA with the perception of depth, positioning guides were integrated. A snapshot of HIDRA and these position guides is shown in Figure 2.8.

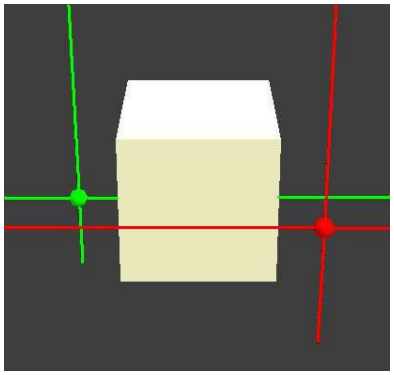


Figure 2.8: Position Guides for Depth Perception

Displayed in this image are a virtual cube, the two virtual fingertips, and the positioning guides. In HIDRA, the user's fingertips are color-coded, red for right finger and green for left, and the positioning guides are lines of the same color extending from left to right and top to bottom. A similar line from front to back is unnecessary as this would not help in depth perception. As can be seen, the positioning guide for the left finger intersects the cube, signaling that this finger is at the appropriate depth. For the right finger, however, both positioning guides are fully visible and appear in front of the cube, indicating the finger is closer than the front face of the cube. Although these positioning guides greatly improve depth perception, there is one immediately noticeable limitation. It was previously stated that the left finger positioning guide intersects the cube. However, as shown, it is possible that the left finger is actually behind the cube. In this case, the actual depth of the virtual fingertip, as compared to the cube, must still be judged by the user. For all experiments described in Chapters 5 and 6, the positioning guides are active for assisted depth perception when using 2D monitor viewing.

2.3.3 Stereoscopic Vision using Shutter Glasses

Another technique that is commonly implemented to assist in depth perception for viewing a virtual environment is stereoscopic vision. This capability was added to HIDRA by implementing the CrystalEyes stereoscopic shutter glasses from StereoGraphics Corporation[3]. Shutter glasses are a lightweight pair of goggles worn by the user and work in synchronization with the graphics display of the simulation to create stereoscopic, or 3D, vision. Essentially, the graphics loop of HIDRA creates two images of the same scene, each at a slightly different perspective corresponding to each eye. These images are alternately displayed to the monitor at a frequency of 96 Hz, where the image for each eye is displayed at 48 Hz. At the same time, each lens of the shutter glasses alternately impedes vision of this scene while the image for the other eye is visible. This synchronization, enabled by an infrared emitter, creates the visual sensation of a 3D scene, even though the graphics are displayed on the 2D monitor. Shutter glasses are a more high-tech version of the red and blue cardboard glasses used in 3D movie theatres. The image shown in Figure 2.9 illustrates the communication between the computer and the shutter glasses through the infrared emitter.

[3] Website: http://www.stereographics.com/

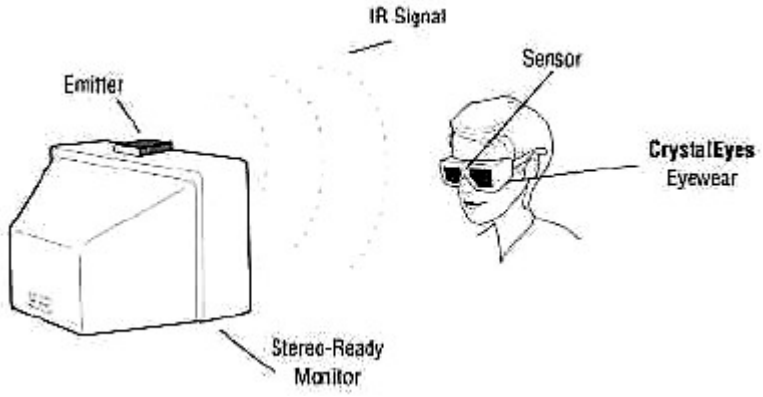


Figure 2.9: Stereoscopic Vision using CrystalEyes Shutter Glasses[4]

Shutter glasses have the limitation that there is only one focal point when viewing the scene. In real life, the human eye changes focus depending on the object being viewed. The graphics scene cannot modify this focal distance since there is no way to determine the object being viewed. In HIDRA, this focal point is set to be the middle of the workspace. As a result, any object extending to far from this point appears in double vision. Typically, the objects are close enough to the center of the workspace that this is not an issue. The positioning guides described in the previous section almost always appear as two lines since the thickness of the guides are small. Therefore, the positioning guides are disabled when stereoscopic viewing is active.

Each technique, positioning guides and stereoscopic vision, attempts to improve depth perception. The relative performance of stereoscopic vision using the shutter glasses versus 2D monitor viewing with the positioning guides will be analyzed in the experiments described in Chapters 5 and 6.

[4] Taken from StereoGraphics CrystalEyes User's Manual.

2.4 LIMITING COMPONENTS OF THE HIDRA SIMULATION

Thus far in this chapter, general information about the HIDRA simulation has been provided, including an overview of the structure and software and hardware components used, a description of previous work on the simulation, and some general enhancements to improve simulation usefulness. In this section, the major limitations of HIDRA will be highlighted, some leading to the research conducted for this dissertation.

2.4.1 Collision Detection Bottleneck

A necessary and key component for realistic simulation of assembly and disassembly procedures is the modeling of the interaction between virtual objects. To model these interactions, collision detection is required in order to detect and respond to collisions. However, the collision detection process is considered to be a "major computational bottleneck" in many applications of dynamic bodies, including assembly and disassembly analysis (Lin and Gottschalk 1998).

During initial development and testing of HIDRA, the potential collision detection bottleneck became apparent during the insertion of a simple bolt into a hole (see Figure 2.6). In fact, during this scenario the collision scheme failed with "unacceptable regularity" by allowing large object intersections (McDermott 1999). Suggestions were made for improvement of object interactions through candidate techniques that could support collision detection algorithms for enhanced performance. In this research, such techniques are developed and tested in an effort to achieve more robust object interaction performance and answer Research Question 1.2.

RQ 1.2 – How can current collision detection libraries be supplemented to improve object interactions within HIDRA, while limiting detrimental effects and preserving the realistic quality of the simulation?

Further discussion on this topic and research question is provided in Chapters 3 and 4.

2.4.2 Limitations of Haptic Interaction in HIDRA

Although haptic interaction provides additional feedback from the simulation, it can cause complications with respect to collision detection. As previously mentioned, the necessary update frequency for haptics in a virtual environment, 1000 Hz for the PHANToM, requires that all other routines, which are generally slower, run as secondary processes. Given that the haptic loop runs at a faster rate than the collision loop, object positions are sometimes updated faster than collisions can be detected. As a result, unpredictable accelerations in HIDRA initiated by the user through interaction with the PHANToM interfaces can cause a large change in the position of an object between iterations of the collision routine. This can lead to excessive overlap between objects and failure of a realistic simulation. This problem is less likely to occur in non-haptic simulations where external forces are more predictable, such as gravity. Techniques developed to mitigate this effect are discussed and tested in Chapter 4.

The advantage of a dual-PHANToM setup rests in its grasping capability, rather than just a point and select method where an object follows the motion of a single PHANToM. The ability to grasp an object offers a closer representation to physical reality. Unfortunately, with the increased realism of this setup comes a limitation. The GHOST software provides point-force interaction, wherein the virtual representation of the device is a point at which a single force can be applied. Using the dual-PHANToM

setup, handling the rotation of an object can sometimes be difficult. A 2D example is shown in Figure 2.10.

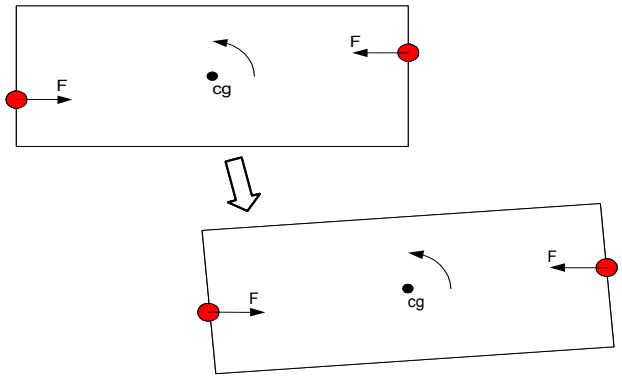


Figure 2.10: Undesired Rotation with the Dual-PHANToM Setup

In this example, two virtual fingertips, represented as spheres, grasp the object. If the grasp positions are not aligned with the center of gravity of the object, rotation will occur. To compensate, the user would need to adjust the fingertip positions to create a moment in the other direction. At times, this limited control of the rotation of an object can cause slight wobbling, an undesired effect. When attempting to assemble objects, the wobbling can make the task of collision detection much more difficult. In the real world, a two-finger grip on an object involves surface interactions, or the contact between the finger pad and the object. Under these grip conditions, the rotation of an object can be handled with ease.

2.4.3 High-Level Simulation Architecture

In Section 2.1, the high-level simulation architecture of HIDRA was presented. As discussed, simulation processes are divided into two main components, the haptic loop and a graphics update callback. Within the graphics update callback are the methods

required to model object interactions, or the collision loop, and the graphic display of the virtual environment. These operations are completed in sequential order. This means that the collision loop can only be executed as fast as the time required to generate the graphic display.

As mentioned in Section 2.4.1, the collision detection process can be very computationally expensive in itself. Additionally, the presence of unknown forces and accelerations inherent with haptic interaction produces a greater demand on the collision loop to detect collisions before unrecoverable penetration occurs. Ideally, the collision loop should execute each and every time the haptic loop completes a cycle, or at a frequency of 1000 Hz, in effect limiting large object movements between collision checks. However, this update frequency is not possible for the collision loop under the current simulation architecture since the cycle time is dependent on the graphics update, which generally requires several milliseconds. As such, the collision loop should be executed as a stand-alone process, rather than a piggyback of the graphics display. In this manner, collision checks can be performed at a higher frequency, improving the robustness of object interactions.

This need for more efficient simulation architecture, particularly with respect to executing the collision loop, initiated the desire for Research Question 1.3.

RQ 1.3 – How can a collision detection library be integrated with the HIDRA simulation to maximize the collision performance?

This question and its corresponding hypothesis will be addressed in Chapter 4.

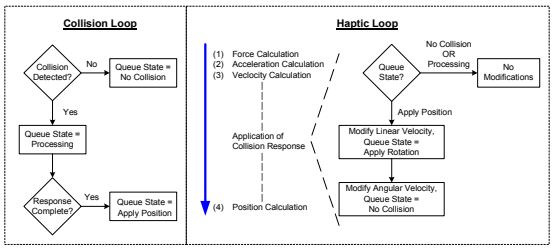
2.5 CHAPTER SUMMARY

In this chapter, the HIDRA simulation, which was designed for assembly and disassembly analysis in a virtual environment with force feedback, was presented. An introduction and overview of HIDRA was provide in Section 2.1, including the hardware and software resources implemented, the graphical user interface, and the high-level simulation architecture. In addition, the major components of the simulation, the haptic loop, collision loop, and graphic display, were introduced. A basic understanding of these components will facilitate the discussion in the chapters to come. In Section 2.2, the prior accomplishments during creation and initial testing of HIDRA were discussed. The major contributions were construction of objects via CAD transfer, integration of collision detection and response algorithms, and reduction of the computational load on the haptic loop. It became apparent that the collision detection and response algorithms were insufficient to realistically model object interactions during assembly and disassembly of a relatively small number of components. General enhancements to improve user interactions with HIDRA were presented in Section 2.3. Lastly, some of the limiting factors of the HIDRA simulation were discussed (Section 2.4). These factors, particularly the collision detection bottleneck and high-level simulation architecture, contribute directly to the formulation of Research Questions 1.2 and 1.3.

In the next chapter, the details of modeling object interactions and collision detection will be discussed. The topics covered will begin to address Research Question 1 and the corresponding hypothesis.

CHAPTER III

MODELING OF OBJECT INTERACTIONS IN A VIRTUAL ENVIRONMENT



The modeling of interactions between virtual objects is an integral part of HIDRA, without which realistic simulation is not possible. In this chapter, the details of object interactions in HIDRA are presented. In Section 3.1, the concept of physically based modeling is discussed, including the advantages and disadvantages of each approach. Then, a review of collision detection techniques for virtual objects is provided in Section 3.2. Also, a description and qualitative comparison of publicly available software libraries for collision detection is presented. Finally, the details of implementing object interactions in HIDRA are discussed in Section 3.3. The topics covered in this chapter are directed towards answering the first research question.

3.1 PHYSICALLY BASED MODELING IN SIMULATION

Physically based modeling refers to the modeling of an object's motion based on mathematical models representing the dynamic characteristics of the object and its surroundings. Modeling the motion of a point is the most basic form of physically based modeling. Here, particle dynamics governs the motion of the point. For instance, the acceleration imparted on a particle by an external force equals that force divided by the particle's mass. For two and three-dimensional objects, rigid body dynamics can be used to model the behavior of an object. However, the task of modeling the dynamic behavior of three-dimensional bodies becomes more complex when they interact with one another.

Two distinct approaches have been studied for modeling non-penetrating bodies in a virtual environment. The first, known as constraint-based simulation, attempts to model the interaction between each pair of contacting bodies as a set of constraint equations. On the contrary, impulse-based simulation models contact between bodies as a series of collisions. Impulse-based simulations employ a collision detection strategy that is well suited to deal with objects that are free moving and randomly collide with one another. A constraint-based simulation is more suited to objects that are constrained in their motions due to the presence of another object. Gillespie and Colgate (1997) provide an overview of techniques used to simulate multi-body dynamics. Ideally, the collision detection scheme for a virtual assembly should contain impulses and constraints, combined to create a hybrid simulation capable of handling all types of interaction equally well. A more in-depth description of these object-modeling techniques is provided in the following sections.

3.1.1 Constraint-based Simulation

As the name suggests, constraint-based simulation models interaction between a pair of virtual objects as constraints. Upon solution to a set of equations, forces are applied to each body that maintain constraints and prevent inter-object penetration. A constraint-based simulation would more efficiently simulate a door on a hinge because the door is constrained to rotation about the axis of the hinge (see Figure 3.1). It would be very difficult for an impulse-based simulation, on the other hand, to maintain all of the small collisions occurring between a door and its hinge. Another example in which a constraint-based simulation would be more appropriate is a stack of blocks. In this case, each block is constrained from movement in the vertical direction by the blocks above and below.

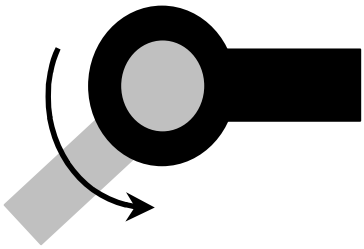


Figure 3.1: Rotating Hinge – Ideal for Impulse-based Simulation. Taken from (Mirtich 1995).

As a leading researcher in the field of multi-body dynamics, Baraff (1989) introduced a method for analytically calculating the forces between rigid polyhedra in resting contact. The method solves a first order system of coupled ordinary differential equations for contact forces that will prevent inter-penetration of objects. Although

designed to handle constraint equations, his formulation can be modified to deal with colliding objects.

Baraff (Baraff 1991) also extended his work in rigid-body simulation to include dynamic friction between bodies. He presented two preliminary approaches for dealing with static friction. In (Baraff 1994), Baraff provides a summary of his work on the dynamic simulation of non-penetrating rigid bodies, including the algorithms and detail for implementing his contact model and the modeling of static and dynamic friction.

3.1.2 Impulse-based Simulation

An impulse-based simulation models object interactions by detecting collisions and applying an impulse response to each object. As such, this type of simulation is more adaptable to a dynamic environment where object positions are continually changing with respect to one another. The interaction of balls in a game of billiards is an example of a situation where impulse-based modeling has a distinct advantage over constraints. Given the fact that billiards balls spend little time in contact with each other, a constraint-based approach would be virtually helpless.

Much more research has been performed in impulse-based simulation. Hahn (Hahn 1988) modeled the reaction between bodies colliding at a single point as an impulse. For bodies in resting contact, a series of impulses is used to prevent interpenetration. As such, Hahn modeled rolling and sliding contacts using impact equations. Similarly, Moore and Wilhelms (Moore and Wilhelms 1988) also calculate an impulse generated between two bodies colliding at a single point. However, they utilize a spring force penalty method to prevent resting bodies from inter-penetrating.

In (Mirtich and Canny 1995; Mirtich and Canny 1995; Mirtich 1996), Mirtich and Canny present the theory, mathematical derivation, proofs, and implementation algorithms of an impulse-based technique for the dynamic simulation of rigid body objects. In this technique, non-colliding contacts between bodies are simulated as a series of small collisions.

Chang and Colgate (Chang and Colgate 1997) have developed a real-time impulse-based simulation for haptic display of rigid bodies. In this simulation, due to the variable external force from a haptic interface, complications arise in the collision detection and response algorithms. Several different techniques were discussed to accurately model an objects reaction to a collision, but all had their downfalls. The authors note that the state of objects at the exact moment collision in simulation is rarely achievable, and, therefore approximations have to be made. Ultimately, the authors suggest that the best choice for collision response between virtual objects would involve both impulses and force constraints.

The two modeling techniques discussed thus far, using constraints or impulses, each have their own shortcomings and are more appropriate in different situations. In the next section, research leading toward physically based simulations incorporating both impulses and force constraints will be highlighted.

3.1.3 Hybrid Simulation: Combining Impulses and Constraints

Clearly, each type of simulation, constraint-based and impulse-based, has its advantages over the other. For instance, constraint-based simulations simulate rolling and sliding contacts very well. Also, mechanical systems such as hinges can be maintained very easily by constraint equations. On the other hand, impulse-based simulation can

model the interaction between many colliding objects more efficiently, such as the balls in a game of billiards.

Mirtich (Mirtich 1995) proposes a hybrid simulation of impulses and constraints, combining the strengths of each. Processing many collisions between two objects, when one is resting on or against the other, is overkill. A simulation should utilize constraints for situations such as this. Therefore, Mirtich (Mirtich 1998) has introduced a more sophisticated method for determining contact forces, rather than using a series of impulses, to prevent inter-penetration of non-colliding bodies. In this work, if the collision velocity falls below a certain threshold, the closest points are considered contact points. A small impulse is delivered to the body to bring collision velocity to zero. Then, the contact forces required to enforce separation and the appropriate frictional forces are calculated.

In Figure 3.2, we show a scenario in which a hybrid simulation would be beneficial. The figure depicts a ball rolling up a ramp and over the edge of a drop-off. During the first half of the balls path, a hybrid simulation would use a rolling constraint to model the interaction between the ball and the surface. As the ball leaves the surface of the ramp and begins to bounce, all constraints would be removed and impulses would govern the interaction. Finally, as the ball stops bouncing and begins to roll again, a constraint would be added between the ball and surface, and impulse generation would cease. Although the concept of a hybrid simulation is straightforward, the implementation of such a model has eluded researchers to this point.

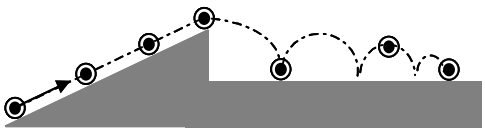


Figure 3.2: A Hybrid Simulation could Model Rolling with Constraints and Bouncing with Impulses. Adapted from (Mirtich 1995).

The techniques of constraint-based and impulse-based responses for rigid-body simulation have been discussed. We have also introduced the idea of a hybrid simulation, combining the strengths of each. These methods address the response of dynamic objects to collisions. Before calculating the response, however, collisions must be detected between these bodies.

3.2 COLLISION DETECTION FOR VIRTUAL OBJECTS

Collision detection is considered by many to be a major computational bottleneck in the simulation of dynamic objects. The arbitrary shape and complexity of some objects, along with a possibly large number of objects, compounds this problem. Nonetheless, many different techniques have been developed to detect collisions between objects in 3D space, utilizing the many representations of an object. Lin and Gottschalk (Lin and Gottschalk 1998) provide a good overview of collision detection techniques based on the model representation of an object, types of queries required, and simulation environment characteristics. Figure 3.3 illustrates a hierarchy of 3D model representations.

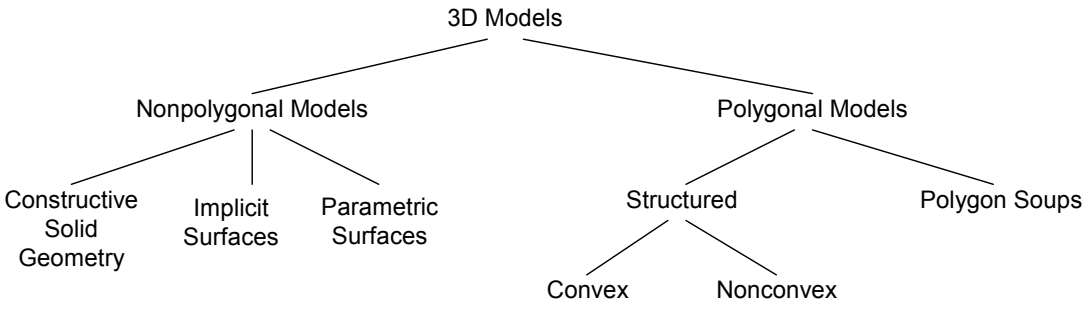


Figure 3.3: A Taxonomy of 3D Model Representations. Taken from (Lin and Gottschalk 1998).

For our purposes, we are only interested in collision detection techniques for polygonal models. Polygon soups, the simplest form of polygon models, contain a listing of polygons that are grouped to form the surface of an object. For a convex polygonal model, all interior angles between adjoining faces are no more than 180 degrees. Non-convex polygonal models do not have this limitation.

Some techniques for collision detection do not depend on the type of model representation used. For example, a general contact analysis method for planar mechanical systems and related tasks of dynamic simulation and tolerance analysis was implemented using configuration spaces (Joskowicz and Sacks 1998). Another commonly used technique is space partitioning (Ganter and Isarankura 1993), which reduces the time required for collision detection, at the expense of memory, detail, and accuracy. However, in order to calculate the dynamic response to an inter-object collision, certain information is required. We must be able to find the point of closest approach, or minimum distance, between a pair of objects.

3.2.1 Minimum Distance Computation

Not much has changed in the calculation of minimum distance between two polyhedra over the past decade. Two techniques have dominated dynamic body simulation. The first technique, developed by Gilbert, Johnson, and Keerthi (Gilbert *et al.* 1998), uses the Minkowski difference and optimization techniques to compute the distance between convex polyhedra by finding the closest points between them. Similarly, Lin and Canny (Lin and Canny 1991) calculate the minimum distance between convex polyhedra based on local features and exploiting geometric coherence. Almost all techniques used to calculate the minimum distance between to polyhedra are based on these two methods. In (Lin *et al.* 1993), the Lin-Canny method is extended to find the minimum distance between non-convex polyhedra, assuming they are subdivided into convex pieces. Other supplemental procedures, aimed at reducing the time to compute the minimum distance or collision status between two objects, have been developed and will be discussed in the next section.

3.2.2 Bounding Volume Hierarchies

One general technique utilized to improve the efficiency of collision detection is the use of a bounding volume hierarchy (BVH). A BVH is a hierarchy of simple volumes such as spheres or boxes, which are used to estimate the actual shape of an object at varying levels of detail. Not only can BVHs be implemented to expedite the process of collision detection, but they can also be exploited to compute an estimation of the minimum distance between two objects. Below is a brief discussion of many of the BVHs that have been developed.

The two simplest types of bounding volumes include spheres and axis-aligned bounding boxes (AABB). Determining the distance or intersection status between two spheres requires subtraction of the sphere radii from the distance between the sphere centers. An AABB places a box around an object with its sides aligned with predefined axes in the simulation, normally world or base coordinates. The distance or overlap between two AABBs can be checked by computing the separation along each axis. These bounding volumes provide very simplified distance and intersection calculations, and, therefore provide the inspiration for more complex BVHs.

Gottschalk et al. (Gottschalk *et al.* 1996) introduce a hierarchical representation of models using tight fitting oriented bounding box trees, or OBBTrees. An OBB is a rectangular bounding box at an arbitrary orientation. OBBTrees were shown to converge asymptotically faster for close proximity situations than hierarchies of spheres or axis-aligned bounding boxes.

In (Hubbard 1996), polyhedral objects are approximated with a hierarchy of spheres using medial-axis surfaces, or a skeletal representation of the object. The method optimizes the tightness each hierarchy level of spheres approximates an object and is shown to converge more rapidly to the actual object than an octree-based algorithm. The problem with this technique is that spheres cannot accurately depict flat surfaces without many hierarchies.

A hierarchical bounding volume structure composed of convex polyhedra is presented in (Kim *et al.* 1997). One weakness of the hierarchical convex tree, or HCTree, is the fact that the automated generation of the polyhedral pieces may not be optimal. Also, the traversing order of the tree for collision testing may not be optimal.

Krishnan et al. (Krishnan *et al.* 1998) have developed a BVH composed of spherical shells, defined as a portion of the volume between two concentric spheres, enclosing the underlying geometry. They have shown that an overlap test for spherical shells is only two to three times slower than that for an OBB. Spherical shells, which exhibit local cubic convergence to underlying geometry, demonstrate improved proximity query performance for objects in close proximity configurations or highly curved objects.

Klosowski et al. (Klosowski *et al.* 1998) introduce the use of discrete oriented polytopes, or k-dops, as a bounding volume structure. A k-dop is a convex polytope whose facets have outward normals in one of k orientations. The advantage with k-dops, as opposed to bounding volumes such as OBBs, is that the orientation of each face is specified, and expensive transformations do not have to be performed.

A technique that combines and extends the advantages of OBBs and k-dops is presented in (He 1999). Quantized Orientation Slabs with Primary Orientations, or QuOSPO trees, is a hierarchy consisting of bounding boxes whose faces are oriented in one of a specified number of directions. QuOSPO trees combine the simple shape of an OBB for easier collision detection with the discrete orientations of k-dops. This BVH achieves best performance when there are multiple simultaneous collisions and/or complex objects.

Johnson et al. (Johnson and Cohen 1998; Johnson and Cohen) have developed an algorithm that works on any bounding volume to speed up bound minimum distance computations. The algorithm uses upper and lower bound estimations to prune BVHs more quickly, additionally exploiting temporal coherence. The lower-upper bound framework, or LUB-tree, can be used for polygonal or parametric models.

3.2.3 Public Domain Software Libraries

Much research and progress has been accomplished in the field of collision detection for rigid body simulation. In the preceding two sections, I have discussed the major contributions in minimum distance computation and BVHs for polyhedral objects. However, many optimization techniques that have been developed, such as geometric and temporal coherence, were not discussed in full. As we are not in the field of developing optimization algorithms and collision detection routines, the details of individual optimization techniques are inconsequential. Many software libraries for collision detection are publicly available for use. Our goal in reviewing the literature concerning collision detection was not to develop our own algorithm, or optimize another. On the contrary, we wish to understand the methods implemented by each of the public domain software libraries, so we can make an informed decision on the appropriate library for our use. Below is a brief description of all of the publicly available software libraries for collision detection.

I-COLLIDE

I-COLLIDE, described in (Cohen *et al.* 1996), supports N-body processing and uses AABBs to sweep and prune pairs of objects. It also exploits temporal coherence to speed collision detection when objects move only a small amount. I-COLLIDE can return the minimum distance between object pairs using the Lin-Canny method mentioned in Section 3.2.1 and simply returns zero for penetrating objects. This library requires objects to be convex polyhedra, and the data must be read from an external data file.

RAPID

RAPID, introduced in (Gottschalk *et al.* 1996), is a pair-processing collision detection package that provides contact status only. This was the first library to use the OBBTree discussed above. One advantage of RAPID, compared to other libraries, is that it accepts polygon soups. This provides the user freedom from addressing the issue of object convexity as with structured polygonal representations (see Figure 3.3).

V-COLLIDE

V-COLLIDE (Hudson *et al.* 1997) is an N-body processor that is built on top of RAPID. It unifies the OBBTree BVH implemented in RAPID and the sweep and prune technique of I-COLLIDE. It first performs a high-level sweep-and-prune routine to determine potentially colliding objects. V-COLLIDE then uses RAPID to determine if any candidate pair of objects is colliding. Objects, which may be dynamically added or deleted from the simulation, must be polygon soups. V-COLLIDE only provides contact status on a pair of objects.

Enhanced GJK

The Enhanced GJK (Gilbert, Johnson and Keerthi) algorithm is a pair-processing collision detection technique (Cameron 1997). The algorithm first computes the simplex, or the difference polyhedron between two convex polyhedra, and then searches the simplex for distance minima, thus returning the distance of separation/penetration. The software distribution utilizes Qhull, a library capable of creating convex polyhedra from a collection of vertices.

V-Clip

The Voronoi-Clip, or V-Clip, algorithm (Mirtich 1997) implements an improved version of the Lin-Canny minimum distance computation method. V-Clip is a pair-processing algorithm that requires the use of convex polyhedra or hierarchies of them. It can compute minimum distance, depth of penetration, and closest features between a pair of objects. Similar to the Enhanced GJK algorithm, V-Clip uses Qhull to automatically create convex polyhedra for each object.

SWIFT

SWIFT (Ehmann and Lin 2001), which stands for Speedy Walking via Improved Feature Testing, is an N-body algorithm that requires convex polyhedra or composites. It uses an improved Lin-Canny algorithm to track closest features and a sweep and prune technique with ability to choose the bounding box type. The library provides proximity queries such as intersection detection, exact and approximate distance computation, and contact determination. According to the authors, it provides the same functionality of I-COLLIDE and more.

SWIFT++

SWIFT++ (Ehmann and Lin 2001) is also an N-body collision detection algorithm that uses the SWIFT core for the overlap test between the convex pieces within two objects. The main difference between SWIFT and SWIFT++ is that the latter allows general polygon models. It computes a surface decomposition reducing each polyhedron into small convex pieces, and then groups the pieces hierarchically using convex hulls. SWIFT++ provides all the same functionalities as SWIFT and includes tolerance verification.

H-COLLIDE

H-COLLIDE (Gregory *et al.* 1999) performs collision detection between the PHANToM haptic interface and objects in a virtual scene. The algorithm first computes a hybrid hierarchical representation of all the objects in a scene as a part of pre-computation. It then partitions the workspace into coarse-grain uniform cells and creates OBBTrees for each cell containing primitives. At run-time it determines the cells touched by the PHANToM probe path, and traverses the OBBTree(s) in each of those cells to determine collisions and the surface contact point, if any. Due to the large amount of preprocessing involved, it is believed that this library has only been implemented in scenes with static objects.

Voxmap PointShell

The Voxmap PointShell (VPS) method, introduced in (McNeely *et al.*), is an algorithm developed at Boeing that provides both collision detection and haptic feedback to a 6 DOF PHANToM. Dynamic objects are represented by a set of surface points and inward normals (point shell), while the static environment is represented by a voxmap (volume occupancy map). To create the voxmap, space partitioning is used to divide the virtual environment small cubic cells, and each cell's value correlates to the presence or absence of an object. Collisions are declared when the surface points of the dynamic object intersect an occupied cell of the voxmap. Thus far, applications have been limited to the manipulation of one modestly complex dynamic object at a time. The user directly controls the movement of the dynamic object through a complex static environment via the PHANToM interface. Forces and torques are computed for the dynamic object for all collisions with the environment and sent back to the haptic device.

3.2.4 Qualitative Comparison of Collision Detection Packages

Thus far, I have provided a background for object modeling and collision detection strategies in simulation. In this section, I will begin to address the functionality of the public domain software libraries for collision detection, providing the basis for discussion of Research Question 1.1:

> *RQ 1.1 – Do current collision detection libraries provide the usability and functionality necessary to efficiently handle dynamic object interactions, and, if not, how can they be improved?*

and the corresponding hypothesis:

> *H 1.1 – Public domain software libraries for collision detection provide the usability and functionality to simulate dynamic object interactions, while there are areas for improvement.*

To accomplish this goal, I will present a comparison of the capabilities provided by each of the libraries and discuss the direct benefits and drawbacks to the simulation of dynamic objects.

All of the algorithms described in Section 3.2.3 can be used to determine the collision state between two objects and are assumed to do so with an adequate speed. Libraries such as these are a necessary component in the simulation of dynamic bodies in a virtual environment. For use with HIDRA, however, other features are desired in order to simplify the process of creating a realistic collision response algorithm. Previous work identified a list of features desired for a collision detection library. These are described below.

- *Collision Points (CP):* The collision points are a pair of points, one on each object, that define the closest approach, or minimum distance, between the objects. Without the collision detection library providing this information, the contact points are virtually impossible to determine with a significant degree of certainty, making collision response difficult to simulate accurately.
- *Collision Features (CF):* Collision features are the vertices, edges, faces, or any combination of the three, on which the closest points are located. Although not completely necessary, collision features can enable more accurate collision response.
- *Depth of Penetration (DP):* While the impulse generated from the collision is generally strong enough to eliminate any initial inter-penetration if left untouched, if subject to a constant force parts will inter-penetrate. Knowledge of the penetration distance allows the application a means to remove inter-penetration directly. It should be noted that HIDRA actually declares collisions before two objects intersect. The reasoning behind this approach will be discussed more in Section 4.1.
- *Programmatic Geometry Construction (PGC):* Programmatic geometry construction describes the ability to build the collision detection representation directly from the geometry. This eliminates the need to generate external files, capitalizing on work already done and reducing the risk of error.
- *N-body Detection (NB):* An N-body collision detection algorithm performs collision detection for all objects in a simulation at the same time. More specifically, after the positions of all objects are provided to the library, only one

query is required. N-body libraries may implement bounding volumes or BVHs to reduce the time required to compute the exact distance between all object pairs by eliminating those that are a large distance apart. Since performing an exact collision detection check between each pair of objects in the scene would be very taxing for large scenes, algorithms that integrate this technique are a plus.

These features were used to provide an initial qualitative comparison between candidate libraries for implementation in HIDRA. A table summarizing the features available in each of the four most promising algorithms appears as Table 3.1. There are several libraries discussed in Section 3.2.3 that are not included for various reasons. Firstly, V-Collide, RAPID, and Enhanced GJK do not provide collision point information, which is an absolute necessity for calculating a collision response. The authors of I-Collide state that SWIFT and SWIFT++ are faster, more robust, and provide greater functionality and that these libraries should be used instead. Lastly, H-Collide only provides collision detection between the haptic interface and its environment, rather than between two arbitrary virtual objects. In HIDRA, GHOST performs collision detection between the haptic interface and each object.

Table 3.1: Collision Detection Feature Summary

Algorithm	Features Provided				
	CP	CF	DP	PGC	NB
V-CLIP	✓	✓	✓	✓	
SWIFT	✓	✓		✓	✓
SWIFT++	✓	✓		✓	✓
Voxmap PointShell	✓				

The first collision detection algorithm eliminated from consideration in HIDRA was Voxmap PointShell. This method uses space partitioning which can be memory intensive. Also, its accuracy is limited to the size of each voxel, whereas all of the other algorithms in Table 3.1 can provide exact distance information. Previous work using VPS has suggested that for a simulation looking to "evaluate the fit of parts together, a more exact [collision detection] method would be necessary" (Johnson and Vance 2001). Additionally, due to the design of the algorithm, a simulation can have only one dynamic object in a static environment. This prevents any realistic modeling of the interaction between two or more dynamic objects.

The distinguishing characteristics of the remaining three collision detection libraries are less drastic. First, V-Clip does not provide N-body processing, forcing the simulation to cycle through all object pairs, whereas SWIFT and SWIFT++ process all objects in one query and offer sweep and prune sorting using bounding volumes. As mentioned previously, bounding volumes are effective in reducing the time required to determine if objects are intersecting or are within some tolerance of one another. This

sweep and prune sorting is referred to as broad phase collision detection, and must be defined by the simulation for any implementation using V-Clip. The details of broad phase collision detection will be discussed in more detail in Section 3.3.2.

One major advantage V-Clip has over SWIFT or SWIFT++ is that is provides penetration distance and the corresponding collision points. However, previous work has shown that when faced with relatively large penetration distances, the information reported by V-Clip can sometimes be incorrect (McDermott 1999). When using SWIFT or SWIFT++, neither the collision points nor depth are ever reported for intersecting objects. In either case, excessive penetration for V-Clip or any intersection with SWIFT and SWIFT++, the result is simulation failure. Supplemental techniques have been developed in an attempt to alleviate this problem. In particular, the inability to report the penetration distance between two objects can be overcome by declaring collisions when the separation of two objects is less than a predefined minimum distance. This will be explained in detail in Section 4.1.

The SWIFT++ collision detection library is an extension of the SWIFT library, essentially using SWIFT as a core subroutine to perform collision queries. SWIFT++ adds the functionality that any general polygonal model, convex or non-convex, can be used, whereas SWIFT requires all objects to be convex. Given its advanced capabilities, SWIFT++ is preferred.

Another attractive feature that SWIFT++ provides is the ability to return multiple collision points for a single object pair. V-Clip and SWIFT do not offer this functionality. This feature only becomes important when dealing with non-convex objects, as a pair of convex objects will never have more than one contact point. Figure 3.4 depicts a 2D

collision between two objects, one convex and the other non-convex. SWIFT++ detects and returns both collision points at the same time, whereas V-Clip and SWIFT will only find one collision.

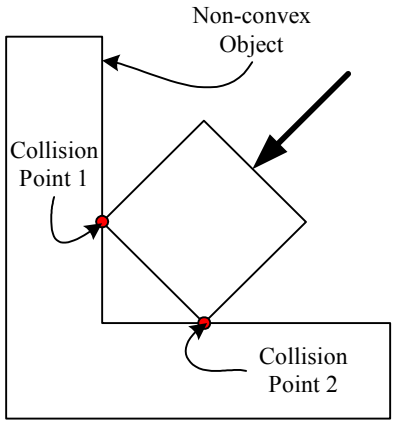


Figure 3.4: Multiple Collision Points for a Non-convex Object

Given this qualitative comparison of public domain software libraries, it becomes apparent that there are two libraries that seem to be most suited for use in HIDRA, V-Clip and SWIFT++. Although similar in many ways, each provides a unique feature, penetration depth for V-Clip and N-body processing for SWIFT++. In order to further compare their capabilities and select a final collision detection library, it was decided to analyze the performance of each library using quantitative data from a simulation scenario. This study will be presented in Chapter 4. A review of this comparison and its relation to RQ 1.1 will be provided in Section 3.4. In the next section, the details of the collision detection process in HIDRA and implementation of both libraries will be discussed.

3.3 MODELING OF OBJECT INTERACTIONS IN HIDRA

Subsequent to the qualitative comparison of collision detection libraries, reviewed in the previous section, two versions of the HIDRA simulation were constructed. The first version, using V-Clip, was a result and direct descendant of previous work (McDermott 1999). By implementing the SWIFT++ library into the simulation, a second version was created. This section will provide an overview of our approach to object modeling in HIDRA, and discuss the differences in implementation between the two simulations.

3.3.1 An Overview of the Collision Loop in HIDRA

The modeling of object interactions in HIDRA occurs within the collision and involves six distinct steps, which are depicted in Figure 3.5.

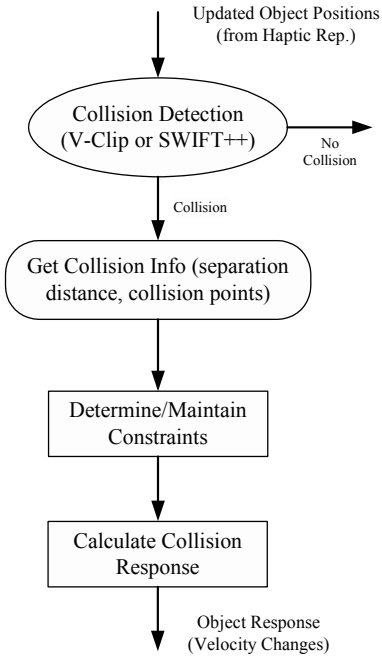


Figure 3.5: Flow Chart Depicting Collision Loop Sequence

A description of each step in the collision loop is described below.

- *Retrieval of Object States from Haptic Loop* – As mentioned in Section 2.1, the dynamic state of each object is maintained by the haptic representation of the object. During the collision detection loop, the first step is the retrieval of object positions.
- *Collision Detection* – This step involves both broad phase and narrow phase collision detection. The broad phase utilizes bounding volumes to quickly prune object pairs that are a relatively large distance apart, whereas the collision detection library, V-Clip or SWIFT++, is queried during the narrow phase to find collisions or near collisions.
- *Get Collision Info* – For each collision found in the previous step, we get the collision points, penetration distance or separation distance, and the collision

features. This information is used in the steps that follow to determine object constraints and response.

- *Determine/Maintain Constraints* – When a collision occurs we must determine the impending constraints on the movement of each object. Also, constraints that are no longer valid must be removed. Detailed discussion of the constraint maintenance technique will be provided in Chapter 4.
- *Calculate Collision Response* – The response of each object to its collisions must be calculated using the collision information and object properties.
- *Report Object Response to Haptic Loop* – An object's change in velocity (linear and rotational) resulting from all collisions is returned to haptic loop for processing with the haptic representation of the object.

The first and last steps described above are essentially handshakes between the haptic and collision loops, in which information regarding the dynamic state of objects is passed. The heart of the object modeling process occurs in the interior functions. The process of gathering collision information from the collision detection libraries is self-explanatory and will not be discussed further. Details of the collision detection process and collision response will be given in the following sections. The process of determining and maintaining constraints, although at this point undefined, will be presented in Chapter 4.

3.3.2 Collision Detection: Broad Phase and Narrow Phase

The collision detection process can be broken down into two main components: broad phase and narrow phase. The difference between the two can be thought of in terms of level of detail. As mentioned, the broad phase refers to intersection testing of the

bounding volumes of each object in the scene with all others. The purpose of broad phase collision detection is to cull the number of object pairs in a scene for which detailed collision information is required. In fact, exact distances and closest points of approach are unnecessary and are not calculated for objects that are a relatively large distance apart. The result is reduced computation time during the collision detection loop.

Due to the nature of the collision detections libraries implemented in HIDRA, V-Clip and SWIFT++, this task varies between the two. In Section 3.2.4, we learned that SWIFT++ is an N-body algorithm, meaning it processes all objects in a scene during one query, rather than a separate query for each object pair. Inherent in this process is broad phase collision using bounding volumes for the virtual objects. SWIFT++ gives the user the option to use either a bounding cube or a dynamic bounding box, which better estimates the shape of objects with large aspect ratios at the expense of computation time.

As a pair-processing algorithm, V-Clip does not perform broad phase collision, and the task is left for the simulation. Therefore, in this version of the simulation, HIDRA implements bounding spheres, which are constructed as part of the collision representation for each object. Bounding spheres offer simplicity, in that the intersection status between two spheres can easily be computed using the difference between the center of the spheres and the sum of their respective radii. If this value is negative, the bounding spheres intersect and narrow phase collision detection is required. Otherwise, the two objects are considered disjoint and further processing is not necessary. In Figure 3.6, the bounding spheres of a ring and shaft are shown to intersect.

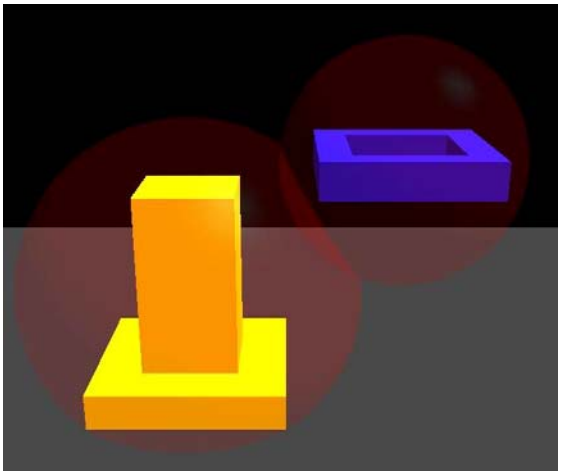


Figure 3.6: Bounding Spheres depicting a Broad Phase Collision

In a simulation using a pair-processing collision library, the need to create bounding volumes for broad phase collision detection initially appears as a disadvantage. However, such an algorithm provides more flexibility in its implementation. This will become apparent during our quantitative analysis of V-Clip and SWIFT++ in Chapter 4.

The narrow phase collision process is handled explicitly by the particular collision detection library, and methods that these algorithms use were highlighted in previous sections of this chapter. For the sake of this discussion, both V-Clip and SWIFT++ are capable of returning separation distance, collision points, and collision features, which are used to calculate object response.

3.3.3 An Overview of the Collision Response Algorithm

After the detection of a collision between two virtual objects, the next step in the collision loop is the response calculation. In HIDRA, this process remains the same, regardless of collision library employed. Details of the technique implemented in HIDRA for impulse-based collision response, including theory, mathematical derivation, and algorithm, are presented in Chapter 3 of the dissertation by Mirtich (Mirtich 1996). In this

section, an overview of the collision response algorithm will be provided, including a discussion of the major assumptions and limitations.

The collision response algorithm implemented in HIDRA models object interactions using impulses. There are three assumptions made in this collision response model. These are described below.

- *Rigid bodies* – All objects in the simulation are perfectly rigid. This implies that collisions and the response to each collision occur over an infinitesimally small interval, and the velocity of each body is changed instantaneously. Also, since "non-impulsive forces have no effect over infinitesimal intervals" (Mirtich 1996), they are ignored during the analysis of collisions. The force of gravity and spring-damper forces are examples of non-impulsive forces.
- *Stronge's hypothesis* – The collision response includes compression, the time when the two objects are approaching each other, and restitution, the time when the objects are receding from one another. Stronge's hypothesis states that the work done by the normal component of the collision impulse during the restitution phase equals $-e^2$ times the negative work done during the compression phase, where e is known as the coefficient of restitution ($0.0 < e < 1.0$). Unlike simpler models, Stronge's hypothesis guarantees that the effects of normal and tangential frictional forces are always dissipative and cannot add energy to the system. The equation representing Stronge's hypothesis is shown in Equation 3.1, where W_z represents the work done by the normal component of the collision impulse, t_{mc} refers to the time at maximum compression, and t_f is the final time of the collision, when the restitution phase completes.

$$W_z(t_f) - W_z(t_{mc}) = -e^2 W_z(t_{mc}) \tag{3.1}$$

- *Coulomb friction law* – The tangential force on a body is related to the normal component of the contact force by a factor μ, the friction coefficient.

As mentioned, object responses to collisions are modeled in two separate phases: compression and restitution. In Mirtich's dissertation, the above three assumptions were utilized to derive a set of differential equations that model each the compression and restitution phases of the collision response. As Mirtich's work does not suggest a particular method for solving these equations, HIDRA implements the 4th order Runge-Kutta method for this purpose. A brief overview of each phase of the collision response algorithm is provided below.

The compression phase is defined as the time during which the colliding objects are approaching each other. During this phase, tangential friction forces are non-constant, and thus dependent variables of the integration. Also, in order to implement Stronge's hypothesis during the restitution phase, the total work done in the normal direction during the compression phase must be known. As such, this variable is also a dependent variable. The independent variable during the compression phase is the normal velocity between the collision points. The value of the normal velocity is known at the outset of the integration, and completion of the compression phase occurs at the point that this velocity reaches zero. At this point, the objects are no longer approaching one another, and the restitution phase begins. At the conclusion of the compression phase, the tangential velocity and the total work done in the normal direction are also known.

The restitution phase is defined as the time during which the bodies are receding from one another. During the restitution phase, the tangential velocity is again non-

constant and is a dependent variable. Although the normal component of relative velocity between the two bodies is known at the beginning of this phase, zero, its value at the end of integration is unknown. Therefore, normal velocity is also modeled as a dependent variable. The independent variable for the restitution integration is the work done in the normal direction. For integration, the normal work starts at zero and proceeds until it reaches the value predicted by Stronge's hypothesis using the compression phase work. The differential equations used to model the restitution phase are not stable when the normal component of the collision velocity is very near zero. As such, the compression phase actually continues computation until this velocity is safely positive.

At the completion of restitution phase, the change in relative velocity of the collision points for the two bodies is known in terms of the collision coordinate frame. The impulse, $\vec{p}$, delivered to each body as a result of the collision can be obtained using this change in relative velocity. As a final step, the impulse is used to compute the change in linear and angular velocities for each body due to the collision. The equations for these velocities are shown below, where m is the mass of the body, I is the body's moment of inertia, and $\vec{r}$ represents the vector from the body's center of mass to the collision point.

$$\Delta\vec{v}_{collision} = \frac{1}{m}\vec{p} \tag{3.2}$$

$$\Delta\vec{\omega}_{collision} = I^{-1}\vec{r} \times \vec{p} \tag{3.3}$$

The major limitation in this collision response algorithm occurs when one object is resting on another. In such a case, the algorithm maintains separation between the two objects through a series of small collisions. This technique works sufficiently when the

collision response is large enough to maintain separation between the two objects but fails when the forces causing repeated collisions are too great for the collision response to prevent penetration. In these instances when objects are resting on each other or constrained in motion in such a way, the interaction between the two objects is "more naturally and efficiently handled with a constraint-based approach" (Mirtich 1996).

In the next section, the integration of the collision response generated by this algorithm with other forces acting on an object, such as gravity and haptic interaction, will be discussed.

3.3.4 Integrating the Collision Response into Equations of Motion

In the previous section, an overview of the algorithm implemented to compute collision response was discussed, including the assumptions, general procedure, and its limitations. The output of the collision response algorithm is the change in linear and angular velocity for each object involved in the collision as a result of an impulsive force. The next step is to incorporate these velocity changes into the calculation of the dynamic state of each object.

There are three factors that affect an object's linear and angular velocity: haptic interaction through the PHANToM interfaces, response to collisions, and gravity. The summation of these factors during the update of an object's dynamic state in the haptic loop results in the overall change in linear and angular velocity for the object. The equations shown below are a generic representation of this summation.

$$\Delta\vec{v} = \Delta\vec{v}_{haptic} + \Delta\vec{v}_{collision} + \Delta\vec{v}_{gravity} \tag{3.4}$$

$$\Delta\vec{\omega} = \Delta\vec{\omega}_{haptic} + \Delta\vec{\omega}_{collision} \tag{3.5}$$

In HIDRA, all virtual objects are modeled with constant density. As such, the force of gravity has no effect on an object's angular velocity. The change in linear velocity of all objects in the simulation due to gravitational force is defined in Equation 3.6, where g is the gravity constant.

$$\Delta\vec{v}_{gravity} = \vec{g}\Delta t \tag{3.6}$$

The method for calculating $\Delta\vec{v}_{collision}$ and $\Delta\vec{\omega}_{collision}$ was discussed in Section 3.3.3 and the equations were given in Equations 3.2 and 3.3. The last source of velocity changes on an object is related to haptic interaction. The equations used to derive the change in linear and angular velocity due to haptic interaction are provided in Equations 3.7-3.10.

$$\vec{a}_{haptic} = \frac{\vec{F}_{haptic}}{m} \tag{3.7}$$

$$\Delta\vec{v}_{haptic} = \vec{a}_{haptic}\Delta t \tag{3.8}$$

$$\vec{\alpha}_{haptic} = I^{-1}\vec{\tau}_{haptic} \tag{3.9}$$

$$\Delta\vec{\omega}_{haptic} = \vec{\alpha}_{haptic}\Delta t \tag{3.10}$$

In these equations, $\vec{a}_{haptic}$ and $\vec{\alpha}_{haptic}$ are the linear and angular accelerations of the body, respectively. Also, $\vec{F}_{haptic}$ and $\vec{\tau}_{haptic}$ refer to the force and torque delivered to each body through haptic interaction. Lastly, m and I are the mass and moment of inertia for the body.

The only additional assumption involved in calculating the change in a body's linear and angular velocity (Equations 3.4 and 3.5) is that the summation of the individual velocity changes is physically valid. As previously discussed, non-impulsive forces have no effect on the collision response algorithm implemented in HIDRA since the duration of a collision is considered to take place over an infinitesimal time period. Based on this assumption, the velocity change on an object due to all non-impulsive forces can be computed separately and summed with the collision response velocity change. Gravity is one such example of a non-impulsive force in the simulation. However, to maintain complete validity in the physical model for object motions and their interactions with other objects, the collision response must be computed and the dynamic states of objects involved must be modified at the exact moment of collision. This requires that the exact collision time be computed and that all object states be updated to the exact time of the collision. Afterwards, the collision response can be used to modify and reinitialize the dynamic state of each object involved in the collision at that time. Then, the integration and computation of object states can continue as normal. Only then will the model for object motion be truly physically valid. Due to the nature of real-time simulation of dynamic bodies, however, this approach may be too computationally expensive, and would occasionally result in unreliable simulation behavior. Also, software limitations, particularly with GHOST, inhibit such an implementation. The method described above closely approximates the laws of physics, such that object motions and interactions remain realistic to a human operator.

The other source for velocity changes on an object result from haptic interaction. To understand the physical meaning behind these velocity changes, we must first explain

how forces are computed for interaction between the PHANToM interface and an object. The model for computing the force on an object through haptic interaction is graphically depicted in Figure 3.7.

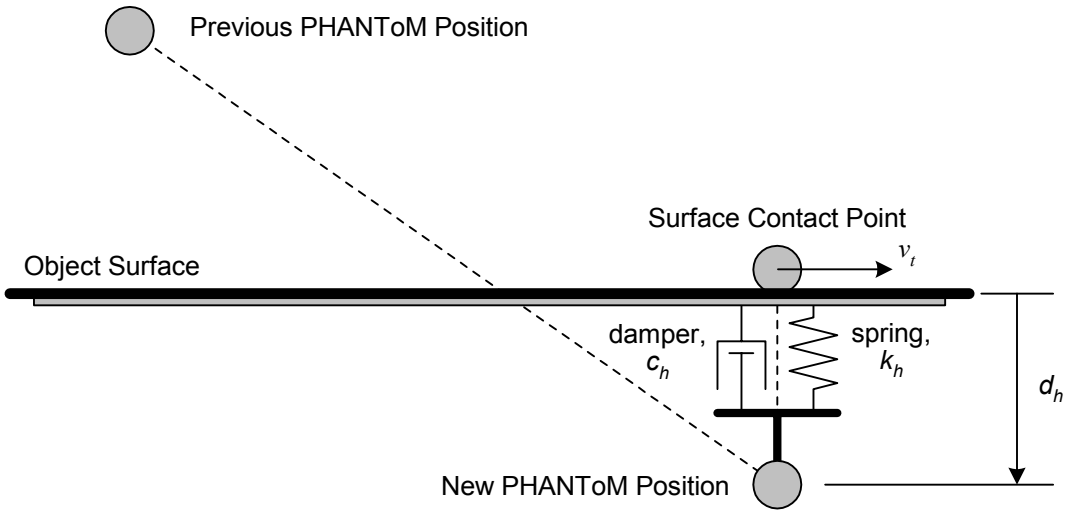


Figure 3.7: Representation of Haptic Interaction with a Virtual Object. Adapted from (SensAble Technologies 2000).

In this figure, the PHANToM interface moves from its previous position to a new position, intersecting the surface of an object along the way. GHOST handles such collisions by defining a surface contact point such that a line drawn from this point to the new PHANToM position is perpendicular to the surface of the object. The depth of penetration of the PHANToM into the object is determined to be the perpendicular distance, d_h (shown in figure), and the corresponding penetration vector in world coordinates is $\vec{d}_h$. A force of equal magnitude and opposite direction is applied to both the PHANToM interface and the object. A spring-damper model is used to compute the magnitude of this normal force, as shown in Equation 3.11.

$$\vec{F}_{haptic} = (k_h \vec{d}_h + c_h \dot{\vec{d}}_h) \cdot (1 + \mu_h \cdot \hat{v}_t) \tag{3.11}$$

In this equation, $\dot{\vec{d}}_h$ refers to the relative velocity between the PHANToM and the object along the line perpendicular to the surface of the object. The added term in Equation 3.11, $\mu_h \cdot \hat{v}_t$, corresponds to the frictional force between the PHANToM and the object using Coulomb's friction law, where μ_h is the friction coefficient and $\hat{v}_t$ is the unit vector of the relative tangential velocity of the PHANToM with respect to the object's surface in world coordinates. Within GHOST, the torque on the object, $\vec{\tau}_{haptic}$, is also computed using this force vector. The total force and torque imposed on an object through interaction with both PHANToM interfaces is then used to compute the change in linear and angular velocity due to haptic interaction using Equations 3.7-3.10. Since the haptic interaction model is a spring-damper system, the force and torque delivered to each object using this model are non-impulsive. Again, non-impulsive forces have no effect on the validity of the collision response algorithm. Therefore, based on the same arguments above when discussing gravity, the equations for the total change in linear and angular velocity of a body provide a good approximation to the laws of physics.

3.3.5 Collision Queues to Maintain Collision Response Data Integrity

In the previous section, the integration of collision response values into the equations of motion for an object was discussed. Prior to incorporating these collision response values, however, they must be reported back to the haptic representation of the object, which maintains the dynamic state for each object. Although this may seem a straightforward procedure, care must be taken to make sure that data integrity is maintained during the transfer between the collision and haptic representations. In other

words, the haptic representation should not access the collision response data during calculations of the collision loop.

In order to handle the data transfer of collision response information, HIDRA uses collision queues. During the collision loop, a collision queue is generated for each object involved in one or more collisions. The queue contains the change in linear and angular velocities due to collisions and the state of collision processing. The queue state can be one of four values: no collision, processing, apply position, and apply rotation. Figure 3.8 is shown to demonstrate the application of the collision queue and describe the function of the queue state.

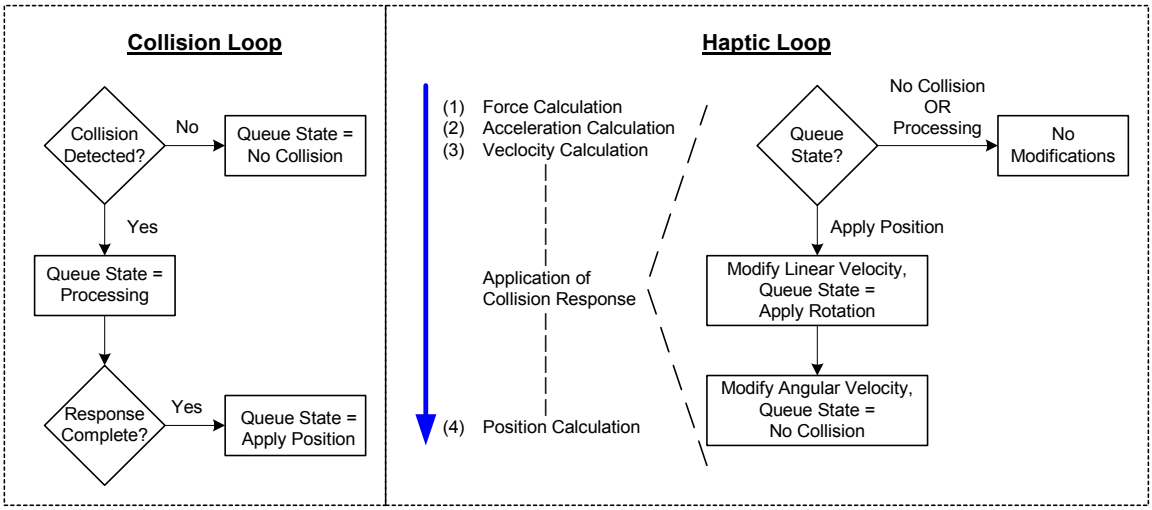


Figure 3.8: Generation and Application of Collision Queues in the Collision and Haptic Loops

In the collision loop, the value of the queue state for an object is set to 'processing' when a collision is detected. This prevents the haptic loop from attempting to access the data contained in the queue prior to completion of the response calculation.

Upon completion of the response calculations, the queue state is then set to 'apply position'.

The haptic loop calculates the state of each object in four distinct stages: force, acceleration, velocity, and position calculations. During each cycle, the haptic loop examines the object's collision queue state to check for new collision response data. This occurs between the calculations for velocity and position. If the queue state value is 'no collision' or 'processing', no modifications to the object velocity are required. However, when the queue state value is 'apply position', the object velocity is modified by adding the change in linear velocity due to collisions. The queue state is then set to 'apply rotation', and angular velocities are immediately modified. Finally, the queue state is set to 'no collision' to signal the end of processing.

By utilizing the collision queue and the queue state, the integrity of collision response data is maintained, and the information is only processed by the haptic loop when allowed.

3.4 CHAPTER SUMMARY

The modeling of object interactions is a necessary and very important component of a simulation for assembly and disassembly. In this chapter, a background on the concept of physically based modeling and techniques for simulating dynamically interacting objects is provided (Section 3.1). The major topics in collision detection research, including minimum distance separation, bounding volumes, and publicly available software libraries, were discussed in Section 3.2. In the same section, a qualitative comparison between the collision detection libraries was conducted,

highlighting desired features. In Section 3.3, details of how collision detection, collision response, and object interactions were incorporated into HIDRA are presented.

In this chapter, particularly Section 3.2.4, we took the first steps in addressing Research Hypothesis 1.1. Referring back to RQ1.1, do current collision detection libraries provide the usability and functionality to handle dynamic object interactions? The question of usability will be addressed during the quantitative comparison of V-Clip and SWIFT++ (Chapter 5). However, we can answer the question of functionality. In order to model collision response, two crucial ingredients are absolutely necessary: the exact distance between two objects and the collision points, and these bare essentials are available with V-Clip and SWIFT++. In that sense, these libraries do provide the functionality, or features, necessary to model object interactions.

Although recent collision detection libraries do provide the necessary functionality, there are areas in which most libraries desire improvement. These desired capabilities are listed below.

- Support for penetration depth, without which intersecting bodies may lead to simulation failure.
- Support for non-convex objects: Most objects in the real world that we wish to simulate are non-convex. In fact, assembly can't occur without at least one non-convex object. Of the collision detection libraries listed in Table 3.1, SWIFT++ and VPS are the only packages specifically designed to handle arbitrary non-convex objects, but each still requires preprocessing. Although SWIFT and V-Clip can support non-convex objects as a set of convex pieces, this requires that

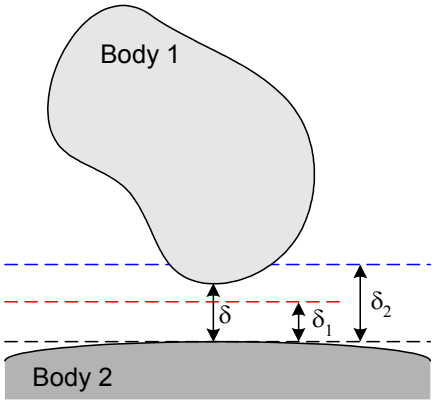
the simulation know the convex components of an object at run time or be able to modify the collision representation of an object during the simulation.

- Multiple collision points per object pair: Most collision detection algorithms only report one set of closest points per object pair. For non-convex objects, the ability to detect collisions occurring in completely different regions of the object, as in Figure 3.4, can be helpful for object modeling.

In short, there exist public domain software libraries for collision detection that provide the functionality necessary to simulate dynamic object interactions, but there are areas for improvement, as listed above.

CHAPTER IV

SUPPLEMENTING AND EVALUATING COLLISION DETECTION IN HIDRA



In the previous chapter, techniques for modeling object interactions in a virtual environment were discussed. Based on the decision to develop an impulse-based simulation, a review of collision detection techniques and software libraries necessary for such a simulation was conducted. In addition, description was then given on how two of these libraries for collision detection were successfully integrated into the HIDRA simulation. In this chapter, the research continues in a direction focused on improving the interaction between virtual objects. In Sections 4.1 and 4.2, techniques for the support of collision detection algorithms, including maintaining constraints between virtual objects, are presented. The method for integrating collision detection and response algorithms within the high-level simulation architecture of HIDRA is studied in Section 4.3. Finally, a quantitative comparison of the collision detection libraries integrated into HIDRA, V-

Clip and SWIFT++, is performed using a simple assembly scenario (Section 4.4). All of the research conducted in this chapter will assist in answering Research Question 1.

4.1 INTEGRATING CONSTRAINTS WITH IMPULSE-BASED OBJECT MODELING

As mentioned in Section 3.1, the simulation of dynamic non-penetrating bodies can be divided into two categories: constraint-based and impulse-based. We chose to create HIDRA as an impulse-based simulation due to the constantly changing interactions between components of a product as it is being assembled or disassembled. When components are in assembled form, however, it is easier to model their behavior using constraints. In addition, collision detection libraries and the corresponding collision response are less efficient in maintaining constraints between objects in close quarters, especially for large or complex scenes, due to the relatively large computation time when compared to the haptic loop frequency. As discussed in Section 2.4.1, previous experiments using HIDRA highlighted this deficiency, leading to the development of the second research question.

> *RQ 1.2 – How can current collision detection libraries be supplemented to improve object interactions within HIDRA, while limiting detrimental effects and preserving the realistic quality of the simulation?*

The first major technique implemented to address this research question is constraint maintenance. Essentially, HIDRA maintains a set of motion constraints between interacting objects, thus reducing the possibility of inter-object penetration. The details of this constraint maintenance scheme are described in the following three sections.

4.1.1 Definition of Collision

Defining a collision for the interaction between two real objects is straightforward, occurring when the two objects touch each other or the distance between them is zero. In fact, two rigid bodies will never have a separation or penetration distance

less than zero. Ideally, virtual collisions between two objects could be defined the same way, namely when the distance between the objects is equal to zero.

In HIDRA, however, the motions of objects are not perfectly smooth since a discrete time-step is used during the integration to calculate object positions. As such, the separation distance between two objects is rarely equal to zero. Objects essentially leap from one position in space to another, although this is perceived as smooth motion given small enough discretization. The effect this has on object interaction modeling is that objects may be separated during one time-step, or update of the object position, and penetrating the next. Although objects may penetrate in this way, the perceived collision and response to this penetration remains believable so long as the intersection distance is small.

As discussed during the qualitative analysis of collision detection packages (Section 3.2.4), most collision detection libraries do not yet provide penetration depth. In fact, V-Clip is the only library described that returns penetration distance for intersecting objects but has been shown to be unreliable when large penetrations occur (McDermott 1999). With SWIFT++, penetrating objects are simply identified as intersecting, and no collision information is provided whatsoever. This incorrect or lack of collision information can cause difficulties when modeling the interaction between two objects.

To alleviate the problems associated with intersecting objects, the definition of a collision is approximated (see Figure 4.1). Assume that two virtual objects are approaching each other and are separated by a distance, δ. As the distance between the objects falls below a predefined threshold, δ_1, a collision is declared, and a collision response is calculated for both objects. In HIDRA, the collision threshold, δ_1, is set at

0.25 millimeters. The dimensions of the simulation workspace are 200 mm wide, 200 mm deep, and 150 mm high, so this collision estimation is unnoticeable and not substantial, except for objects only a few millimeters in size.

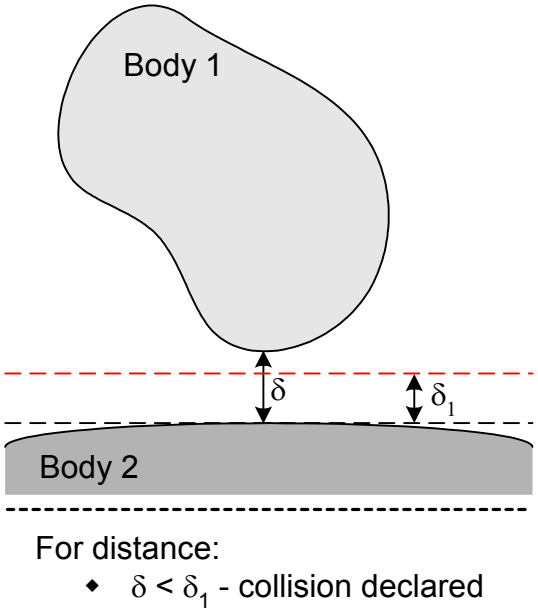


Figure 4.1: Boundary Layer for Collision Declaration

4.1.2 Maintenance of Constraints

To illustrate our approach to constraint maintenance, we refer to Figure 4.2. As Body 1 and Body 2 approach each other, the distance δ is calculated using collision detection software. If that distance becomes less than δ_1, a collision is declared, and a constraint is applied in the direction of the collision normal (denoted as vector *c* in Figure 4.2). This constraint is defined for both objects involved in the collision, and is applied to their respective absolute velocities. A list of vectors is maintained for each object containing the direction of each constraint imposed on the object by collisions with all

other objects in the simulation. Until the next collision loop, the movement of each object is limited by these constraints. In Figure 4.2, we demonstrate graphically the constraint placed on the velocity of Body 1 upon collision with Body 2 in two dimensions. After application of the constraint (see Section 4.1.3), the velocity of Body 1, initially v_1, becomes $v_{1,new}$. The same constraint, in the opposite direction, is placed on the velocity of Body 2. An illustrative example and a more detailed explanation of the application of constraints are provided in the Section 4.1.3.

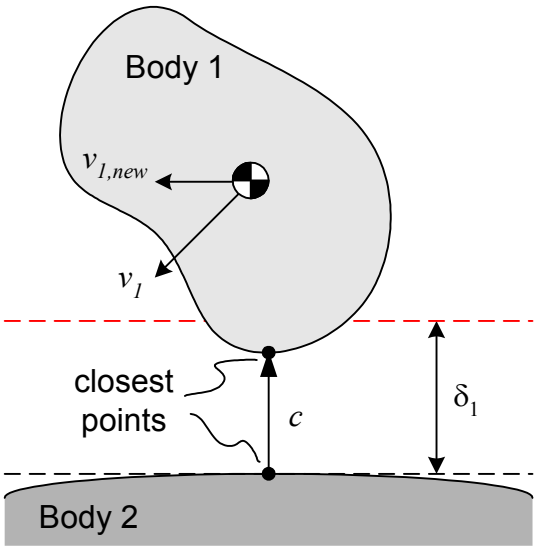


Figure 4.2: Application of a Collision Constraint Applied to Absolute Velocity of Each Body

After constraints are determined and applied, and objects begin to move away from each other, the simulation must determine when the constraint between a pair of objects is removed. Initially, constraints were removed when the movement of the objects separated them by a distance greater than the collision threshold, δ_1. This method, however, created an ill effect on the interaction between the two objects. When the two

objects would collide, not only would a constraint be applied, but also a response to the collision would be calculated and imposed. The response to the collision caused the objects to move apart by a distance greater than the collision threshold, generally by a small amount, releasing the constraint between the two objects. Occasionally, relatively large forces placed on the object(s) by the user through the haptic interfaces would then cause the objects to approach each other and collide again, creating the same constraint application and collision response. This resulted in a bouncing motion between the two objects, sometimes leading to instability in the interaction between the objects.

To compensate for this instability, an additional threshold was included in the constraint maintenance scheme. The constraint removal threshold, δ_2, can be seen in Figure 4.3 along with the original collision threshold, δ_1. Essentially, after a constraint is generated, the objects must be separated by the distance δ_2 before that constraint can be removed. This additional threshold achieved two positive outcomes. First, the undesired bouncing motion between two objects in close proximity was eliminated, resulting in a more stable simulation. Second, since the relative velocity between two objects in the simulation is generally small, corresponding to a small collision response, the constraint removal threshold assured that the collision response between the two objects would not immediately cause the constraint to be lifted. As a consequence, the constraint between the objects is only removed when the user intentionally separates the objects through interaction using the haptic interfaces.

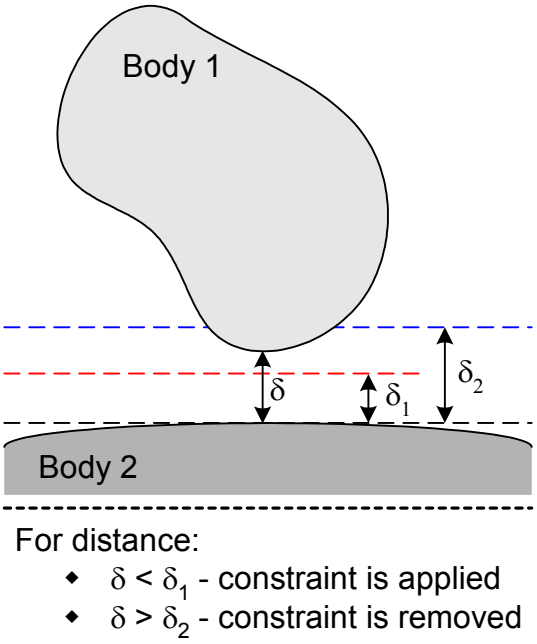
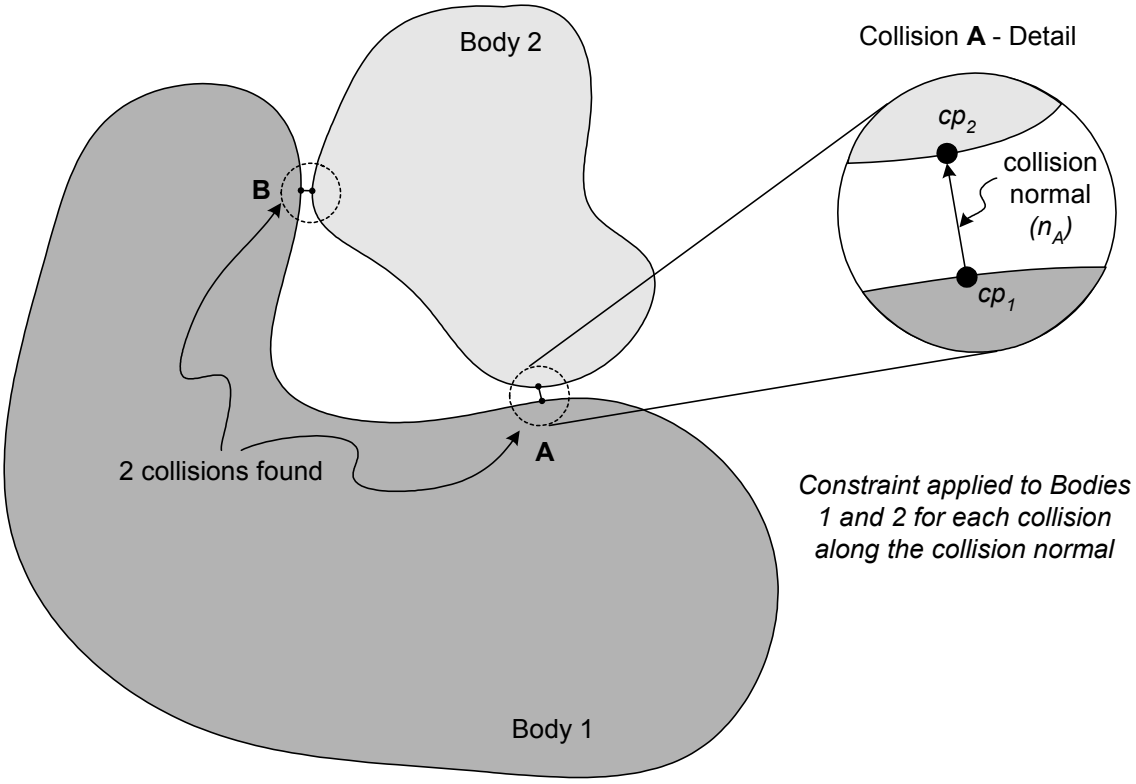


Figure 4.3: Thresholds for Constraint Generation and Removal

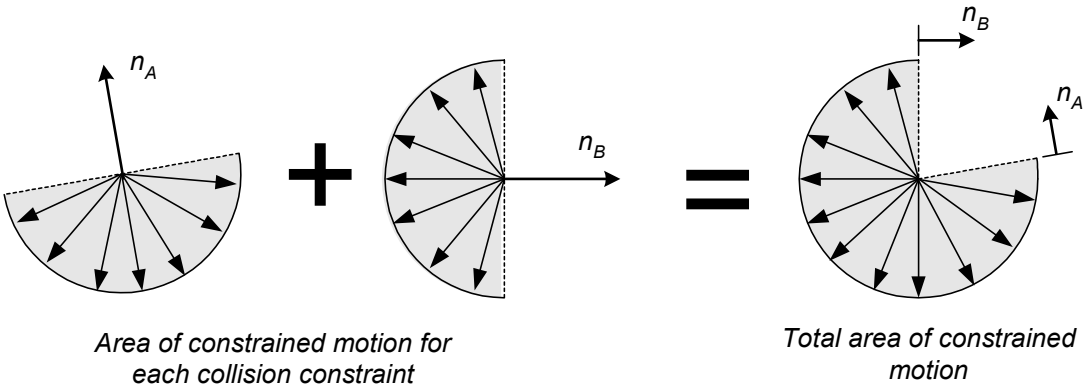
4.1.3 Application of Constraints

Prior to discussing the algorithm implemented in HIDRA to apply constraints resulting from object interactions, an illustrative example will be presented. This example serves to provide a conceptual understanding of the constraint application process.

In Figure 4.4, a 2D representation of the interaction between two arbitrary objects is shown, including the details for a collision and a graphical description of the motion constraints placed on one object due to that collision. In Figure 4.4(a), two collisions are detected between Body 1 and Body 2. For each collision, the collision normal is defined as the vector from the collision point on one body to the collision point on the second body. When these collisions are detected, a constraint is applied to each body in the direction along the collision normal of each collision. As a result of these constraints, the motion of each body is inhibited in the direction along the collision normal.



(a)



(b)

Figure 4.4: 2D Graphical Representation of (a) Multiple Collisions between Two Arbitrary Bodies and the Definition of the Collision Normal and (b) the Area of Constrained Motion on Body 2.

The area of constrained motion, based on the two collisions for Body 2, is graphically depicted in Figure 4.4(b). For Body 2, the area of constrained motion is in the opposite direction of the normal vector based on its previous definition. The area of constraint motion based on Collision A is shown as the first shaded semicircular region in the figure. This region represents the directions in which the modified velocity vector of Body 2, after application of the constraint from Body 1, cannot exist. If the original velocity vector lies within this shaded region, the modified velocity vector will lie on the line tangent to the normal vector, unless the velocity vector is in the exact opposite direction of the normal vector, in which case the modified velocity is zero. If the velocity vector does not lie in the shaded region, the velocity of Body 2 will remain unchanged by the constraint. Next, the area of constrained motion for Collision B is shown. The interpretation of the shaded region for this collision is the same as for Collision A. Lastly, the total area of constrained motion for Body 2 is shown. This total area of constraint is simply the union of the areas of constraint for all collisions between the two objects. Again, the interpretation of the shaded region remains the same. If the velocity vector of Body 2 lies within the shaded region, the new velocity as modified by both constraints will either lie along one of the normal vectors or be equal to zero. If the velocity vector is outside the shaded region, it remains unchanged. For more than two constraints, the total area of constrained motion will always be the union of the area of constrained motion associated with each constraint.

This method of understanding the area of constrained motion due to a collision can be extended to 3D by replacing the semicircular region with a hemispherical region. As with the 2D depiction, all the same rules apply. The total area of constrained motion

on an object corresponds to the union of the area of constrained motion for all collisions. Also, if the original velocity vector lies within the area of constrained motion, the modified velocity will either be perpendicular to one or more of the collision normals or be equal to zero.

For the interaction between two objects, the area of constrained motion on one object will be the reflection of the area of constrained motion for the other about two orthogonal axes. In three dimensions, this is accomplished by a reflection of the area of constrained motion about any three mutually orthogonal axes. In practice, however, an object may have constraints resulting from interaction with multiple objects, which would further increase its area of constrained motion. This discussion of the area of constrained motion has been provided for conceptual understanding, but the actual method for applying constraints handles each object separately.

In HIDRA, constraints are applied to an object between the velocity and position updates for the object. Essentially, the constraint application algorithm shown in Figure 4.5 cycles through all constraints on an object one by one, modifying the velocity of the object as necessary. After all constraints have been applied, the modified velocity is used to update the object's position.

```
    Given:
        vel:   The velocity vector of an object in world coordinates.
        con_i: The i^th constraint on an object in world coordinates.
        eps:   Threshold value (equal to 0.001) to test for degenerate cases.

    /* Cycle for all constraints on object */
1.  for (each constraint, con_i)

    /* Step 1: Determine if velocity vector and direction of constraint are in the same hemisphere */
    /* Compute the dot product of velocity and the constraint vector */
2.      v_dot_c = dot_product(vel, con_i)

    /* Continue if velocity and motion constraint are in same direction */
3.      if(v_dot_c > 0.0) {

    /* Step 2: Define a set of constraint axes with the x-axis parallel to the constraint */
    /* Constraint axes are vectors defined in terms of the world coordinate system */
    /* Check for degenerate case - when con_i is colinear with the world y-axis */
4.          if (x and z component of con_i are both less than eps) {
5.              if (y component of con_i is greater than zero) {
6.                  con_coord_y = world z-axis (0, 0, 1)
7.                  con_coord_z = world x-axis (1, 0, 0)
                }
8.              else (y component of con_i is less than zero) {
9.                  con_coord_y = world x-axis (1, 0, 0)
10.                 con_coord_z = world z-axis (0, 0, 1)
                }
            }
    /* Solve for constraint axes in normal, non-degenerate case */
11.         else {
12.             con_coord_y = cross_product of con_i and the world y-axis
13.             con_coord_z = cross_product of con_coord_z and con_coord_x
            }

    /* Step 3: Normalize the newly defined constraint coordinate system */
14.         con_coord_y = con_coord_y / 'magnitude of con_coord_y'
15.         con_coord_z = con_coord_z / 'magnitude of con_coord_z'

    /* Step 4: Calculate velocity components along y and z axes of constraint coordinate system */
    /* As defined, velocity along x-axis of constraint coordinate system is zero */
16.         vel_mod,y = dot_product(vel, con_coord_y) * con_coord_y
17.         vel_mod,z = dot_product(vel, con_coord_z) * con_coord_z

    /* Step 5: Calculate new velocity as modified by constraint con_i */
18.         vel_mod = velmod_y + velmod_z

        } /* End same hemisphere check for velocity and constraint */

    /* Modified velocity becomes input velocity for next constraint */
19.     vel = vel_mod

    } /* End cycle through all constraints */
```

Figure 4.5: Algorithm for Application of Constraints on an Object

The constraint application algorithm in HIDRA can be divided into five steps. The first step determines if the velocity vector is in the area of constrained motion for the given constraint (see Figure 4.4) by computing the dot product of the velocity vector and the constraint vector, both defined in terms of world coordinates. If the value of this dot product is positive, then the velocity vector is within the hemisphere of constrained motion and the constraint application continues. Remember, opposite to the illustrative example given above, HIDRA defines the constraint vector in the same direction as the area of motion constraint. If the value of the dot product is less than zero, then the object is moving away from the constraint and computations for the constraint are bypassed.

In Step 2, a constraint coordinate system is defined. For ease of implementation, the x-axis of the constraint coordinate system is defined to be equal to the normalized constraint itself as shown in Figure 4.6.

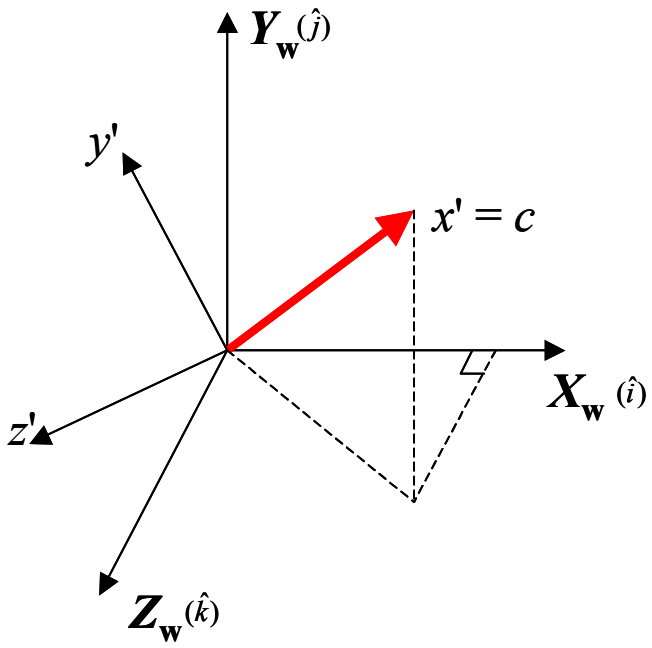


Figure 4.6: Defining the Constraint Coordinate System

The equation for the x-axis of the constraint coordinate system is then given as:

$$x' = c \equiv c_x\hat{i} + c_y\hat{j} + c_z\hat{k}\,. \tag{4.1}$$

Given the definition of the x-axis of the constraint coordinate system, there are an infinite number of possible directions for the y-axis and z-axis, so long as they remain mutually orthogonal. The solution chosen defines the z-axis in the direction of the cross product between x' and the world y-axis (see Figure 4.6). The y'-axis is then defined as the cross product between the z'-axis and the x'-axis of the constraint coordinate system. The resulting equations for the y'-axis and z'-axis are shown below.

$$y' = -c_x c_y\hat{i} + (c_x^2 + c_z^2)\hat{j} - c_y c_z\hat{k} \tag{4.2}$$

$$z' = -c_z\hat{i} + c_x\hat{k} \tag{4.3}$$

There is one degenerate case when Equations 4.2 and 4.3 fail to define a suitable set of axes for the constraint coordinate system. When the constraint lies along the world y-axis ($c_x = c_z = 0$), the definitions for y' and z' become defunct. When this happens, y' and z' are manually assigned values that define a suitable set of mutually orthogonal axes for the constraint coordinate system. In Step 3, the newly defined axes are normalized.

The next step in the constraint application algorithm (Step 4) is to compute the components of the object's velocity ($\vec{v}$) along the axes of our newly defined constraint coordinate system. To find the y' component of the object's velocity, the dot product of the velocity and the y'-axis, both defined in world coordinates, is computed. This gives

the magnitude of the velocity along the y'-axis. This value is then multiplied by the vector for the y'-axis to give the object's velocity along the y'-axis, as defined by world coordinates and in vector form. The equations for the y' and z' component of the object's velocity are shown below.

$$\vec{v}_{y'} = (v \cdot y')y' \tag{4.4}$$

$$\vec{v}_{z'} = (v \cdot z')z' \tag{4.5}$$

Note that, as defined, the component of the constrained velocity along x' equals zero since this is the direction of the constraint, and the equations above represent the only components of the object's constrained velocity. The last step of constraint application is to add these two components (Step 5). The resulting sum is the velocity of the object in world coordinates as constrained by the current and all previous constraints. The modified velocity would then be:

$$\vec{v} = \vec{v}_{y'} + \vec{v}_{z'}\,. \tag{4.6}$$

These five steps are repeated for each constraint until all motion constraints have been applied for the given object. The result of this process is a new velocity for the object that abides by all constraints currently active on the object.

4.1.4 Validity and Limitations of Constraint Maintenance Scheme

The typical method used to implement a constraint on an object is to apply a force against the object, which, in effect, will nullify the velocity along the constraint. However, due to possible interactions with the haptic interface, the forces on an object

are not always known, and our scheme for applying constraints is a simplified method that achieves a similar result. In this section, the validity of this constraint scheme with respect to the physical model of object interactions will be addressed.

The constraint maintenance scheme described in the previous sections essentially applies a force to nullify all other forces in the direction of a constraint. In general, the method implemented is not a physically valid model. There is one instance when the constraint maintenance scheme closely represents reality. A major difficulty for impulse-based modeling is when an object rests on another static object, such as a book resting on a table, or another immovable object. For this scenario, the constraint maintenance scheme works superbly and closely models reality, except for the possible loss of friction, which is explained further below. In fact, the constraint method described in this chapter would work well for any number of objects stacked on top of or pushed against any immovable object.

However, in several other instances, this constraint maintenance scheme deviates from realistic object behavior. To explain these limitations, three examples will be given.

Imagine the scenario shown in Figure 4.7. Body 1, with a mass of *m1*, travels along the surface toward Body 2 with a velocity, v_{m1}. Body 2 rests motionless on the surface and has a mass *m2*, which is significantly less than *m1*. Eventually, Body 1 reaches Body 2 causing a collision. In the real world, and in HIDRA without constraints implemented, the impulse delivered to Body 2 from the collision will impart a velocity v_{m2} on the cube. In addition, since Body 1 is heavier than Body 2, it will continue forward at a reduced velocity. With constraint maintenance active, the response of Body 1 to this collision is quite different. Instead of continuing forward at a reduced velocity, it will

stop and remain motionless. This occurs because a constraint is placed on Body 1 in the direction of Body 2 at the time of collision. When applied, the constraint eliminates the velocity of Body 1 in that direction. Without any further interactions or forces acting on Body 1, it remains motionless indefinitely, since the constraint was imposed on the absolute velocity of Body 1, rather than the relative velocity between Body 1 and Body 2..

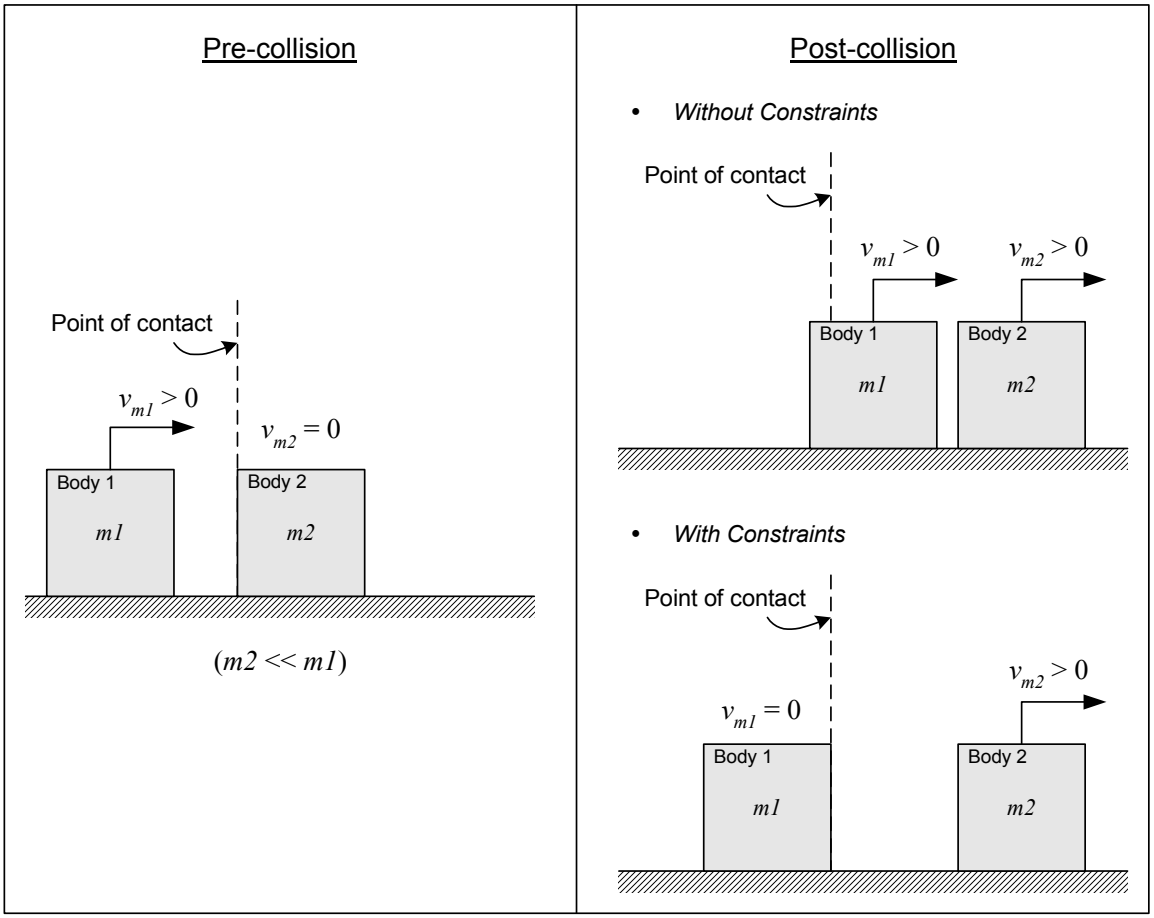


Figure 4.7: Limitation of Constraint Maintenance for Interaction between Two Freely Moving Bodies

In Figure 4.8, a similar but somewhat different scenario is depicted. In this scenario there are three bodies, two of which begin motionless (Body 2 and Body 3) and constrained in motion with respect to each other. Again Body 1 travels with a velocity v_{m1} toward the other two objects. At a certain time, Body 1 collides with Body 2. In the real world, and in the simulation without constraints, the collision causes a chain reaction response and imparts a velocity on Body 3 in the same direction as the original velocity of Body1. Also, Body 1 and Body 2 will generally have some post-collision velocity, which is dependent on the initial velocity, v_{m1}, the mass of each body, and the properties of each body. Again, with constraint maintenance implemented, the reaction is quite different. The impulsive force on Body 2 is not transmitted to Body 3 since a constraint exists between the two bodies, preventing motion of Body 2 in that direction. Due to the collision, Body 2 is also constrained by motion in the reverse direction from the collision with Body 1. Therefore, both Body 2 and Body 3 remain motionless after the collision. Depending on the characteristics of the objects and the collision, Body 1 may rebound slightly in the opposite direction from which it came.

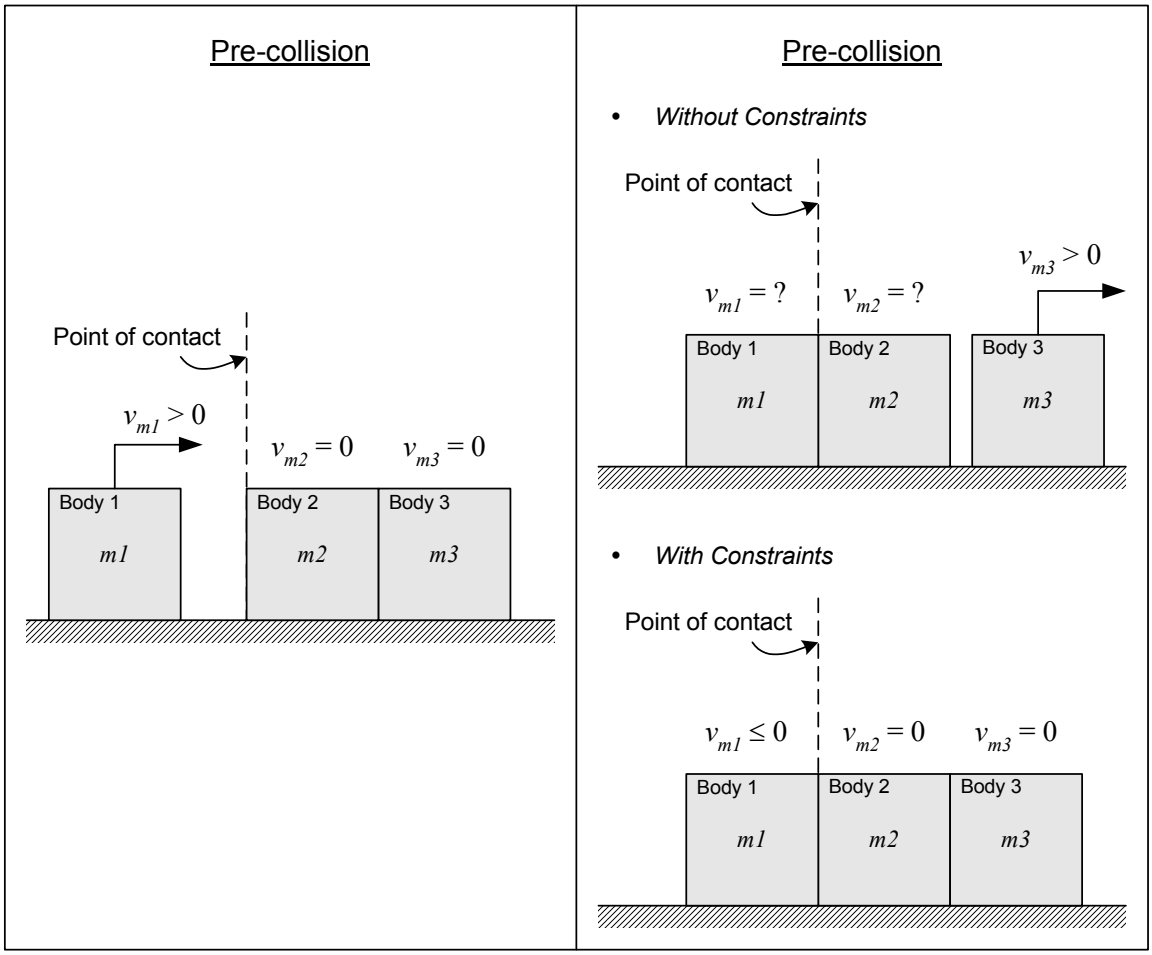


Figure 4.8: Limitation of Constraint Maintenance for Interaction between Multiple Touching Bodies

The last limitation in the constraint maintenance scheme results from the definition of the collision threshold and the threshold for removal of constraints, defined in Figure 4.3 as δ_1 and δ_2, respectively. As discussed in the previous chapter, the collision response algorithm incorporates friction into the model. When a collision takes place, the change in velocity of an object occurs not only parallel to the collision normal, but also perpendicular to this normal, the latter corresponding to the effects of friction. Friction is accounted for in the interaction between the colliding objects through the collision

response, so long as the distance between the two objects is less than δ_1. However, when the distance between the two objects becomes greater than the collision threshold, δ_1, but less than the constraint threshold, δ_2, a collision response is no longer computed because subsequent collisions between the objects are no longer declared. The constraint will still be active and will prevent the objects from colliding again, but the friction component of the collision response is lost. In this state, the objects are free to slide across each other without friction between them.

Clearly, the constraint maintenance scheme described in this chapter has some serious limitations that cause object interactions to deviate from a realistic model. The first two limitations discussed involve objects that randomly collide with one another. In the assembly and disassembly sequences discussed in this dissertation, however, objects do not interact in this way. Instead, only one object is typically in motion at a time during assembly and disassembly, and it is generally handled by the user. Additionally, most other objects in the assembly are anchored while the assembly of a given object occurs. More detrimental to assembly and disassembly scenarios is the potential loss of friction between two objects that are constrained in motion with respect to one another. If an object is placed into an assembly and only partially constrained in motion due to interaction with other objects, it can still inadvertently slide around on the frictionless contact with the other objects. This is obviously very detrimental if the assembly requires that the object remain in one position. These are limitations of the constraint maintenance scheme that one should be aware of when using the simulation.

In the next section, additional techniques to support collision detection and object interactions will be introduced.

4.2 ADDITIONAL TECHNIQUES IN SUPPORT OF COLLISION DETECTION

In addition to the definition and maintenance of constraints for colliding objects or those resting on one another, several other modifications were incorporated into HIDRA. The goal of these enhancements were to improve object interactions, either by reducing fatal object intersections, minimizing the computational load on the collision detection algorithm, or allowing easier object handling. These enhancements are described in the sections below.

4.2.1 User-Defined Constraints

During most assembly or disassembly procedures complex object motions are not required. Although HIDRA allows the full 6 DOF motion of objects, all DOFs are typically not needed to insert or remove a part to or from an assembly. For example, to insert a bolt through a hole requires that the bolt be translated in one direction, namely along the centerline axis of the bolt. Even removal of more complex fixtures such as unscrewing the nut from a bolt require only a small number of degrees of freedom, rotation and translation along the centerline axis.

During manipulations of an object in the real world, a person's hand, fingers, and finger pads provide the constraints necessary to limit the motion of an object for assembly and disassembly sequences. In HIDRA, this interaction with the part is limited to the user's virtual forefinger and thumb. As mentioned in Section 2.4.2, the PHANToM interface used in HIDRA only provides point-force interaction, resulting in a difficulty to handle the rotation of an object. To compensate, the user can dynamically specify translational and rotational constraints along or around the three primary axes of an object.

As an example, imagine the disassembly of the small-scale assembly first discussed in Section 2.2.4 (see Figure 4.9). The first component to be removed from this assembly is the bolt inserted through the top and base pieces. Assuming there are no other unseen parts or features preventing the removal of the bolt, it must simply be removed by translation along its centerline axis. In this case, the user can define constraints on rotation about all axes and on translation along both axes perpendicular to the centerline axis, limiting the bolts motion to 1 DOF. After removal of the bolt, the top piece in the assembly can be lifted again with only 1 DOF with similar constraints applied.

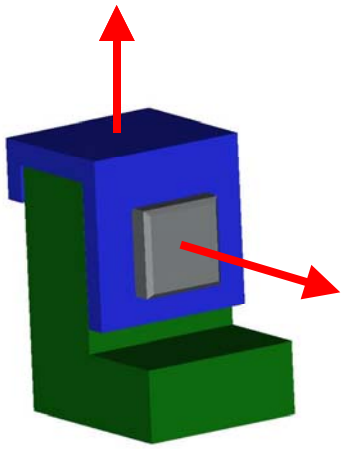


Figure 4.9: Implementing User-Defined Constraints for Small-Scale Disassembly

User-defined constraints, when used in the manner, provide two benefits. First, these constraints assist in the handling of objects through the haptic interfaces, which can sometimes be more difficult due to the point-force interaction. More importantly, they support object interactions by limiting excessive relative motion between objects, which can cause extreme performance degradation and increased computational requirements of the collision detection and response algorithms.

It should be noted that the capability to impose constraints on translation and rotation about arbitrary axes is not available. In most instances, however, this is not necessary. The design of an object in a CAD package typically begins with the definition of a base coordinate system, and the predominant features of the object are defined with respect to and many times aligned with these axes. Accordingly, most translation and rotation required to assemble or disassemble the components of a product are related to its primary axes. However, there are instances when this may not be the case, and this is a limitation of the implementation of user-defined constraints in HIDRA.

4.2.2 Eliminating Angular Velocities During Collisions

To explain the need and usefulness of eliminating angular velocities during a collision between two objects, an example is necessary. In Figure 4.10, a 3D block falls straight down to the surface of a plate. As the virtual block approaches the plate, there are an infinite number of closest points, resulting from the bottom surface of the block being parallel to the top surface of the plate. Regardless of collision detection algorithm used, V-Clip or SWIFT++, only one pair of closest points is tracked. By design V-Clip will never track more than one closest point pair. Although SWIFT++ can generally track multiple closest points, only one pair will be identified in this situation since the two surfaces are parallel and an infinite number of candidate pairs exist.

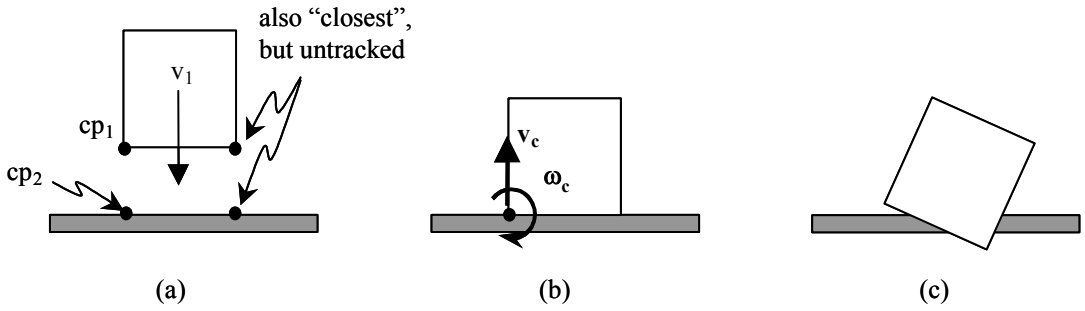


Figure 4.10: Illustration of post-collision rotational velocities problem. Taken from (McDermott 1999).

As the two objects collide, the collision response will generate a translational and rotational velocity vector on the block. However, since the other contact points on the surface of the block are not tracked, the torque applied to the block will cause it to rotate into the plate causing a fatal intersection. For this reason, HIDRA sets all post-collision angular velocities to zero. This solution, although not physically valid, should minimally impact any assembly or disassembly scenarios. More importantly, eliminating post-collision angular velocities prevents fatal object intersections that would otherwise occur regularly.

4.2.3 Velocity Slowdown for Objects in Close Proximity

Another option designed to support collision detection and reduce fatal object intersections in HIDRA is velocity slowdown compensation. In short, the linear speed of any object that gets close to another object is scaled to a slower velocity. Essentially, velocity slowdown imposes a linear reduction in the maximum velocity capable for an object based on the closest separation distance of that object to all others in the scene. The equation for the allowable maximum velocity of an object is shown below.

$$max\_allow\_vel = \frac{closest\_obj\_dist}{vs\_factor} \tag{4.6}$$

In this equation, *closest_obj_dist* refers to the closest distance between the given object and all other objects in the simulation. The value for *closest_obj_dist* is defined by the actual distance between two objects. For instance, when two objects are separated by a distance of 0.25 millimeters as described in Section 4.1.1, a collision is declared, but the value of *closest_obj_dist* is not zero. Instead, its value refers to the actual separation distance between the objects, or 0.25 millimeters. The variable *vs_factor* is the factor used to determine the maximum allowable velocity of the object. In HIDRA, the value of *vs_factor* is arbitrarily set to 30 milliseconds. An interpretation of *vs_factor* follows. For two objects approaching each other at a velocity equal to *max_allow_vel* and separated by a distance equal to *closest_obj_dist* , it will take the time equal to *vs_factor* for the distance between the objects to reach zero. In this formulation, as the value of *closest_obj_dist* approaches zero, so does *max_allow_vel* . Although this rarely occurs in simulation, it causes a very undesirable effect, as objects appear to stick to one another. Since the value of the velocity slowdown factor has no physical meaning, it was chosen through trial and error based on assembly of the ring and shaft pair (see Section 4.4).

Clearly, this is an ad-hoc solution to assist with the difficulties in preventing fatal object intersections in HIDRA. However, this technique is helpful when working with components of a product in close proximity to each other, especially non-convex parts. An example of the importance of its application is when a user attempts the seemingly simple task of assembling the ring and shaft shown in Figure 4.11 (revisited from Section 2.2.3).

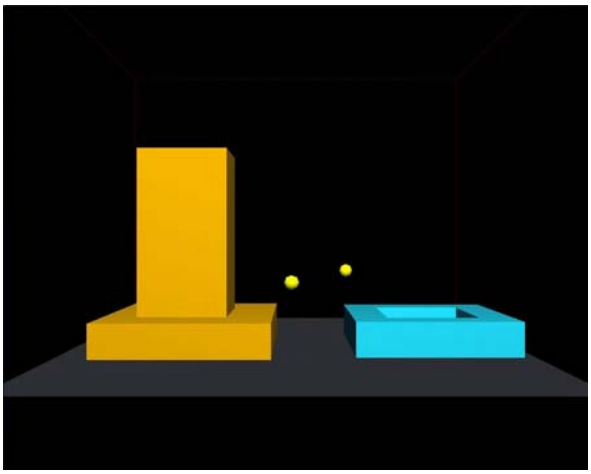


Figure 4.11: HIDRA Simple Ring and Shaft Scene

As the user begins to slide the ring down the shaft, the non-convex nature of the ring can create difficulties for a collision detection algorithm as follows. During the assembly sequence, the ring may shift horizontally along the shaft, causing the closest points between the two objects to jump from one side of the shaft to the other. This discontinuous movement of the closest point requires much more computational time for the collision detection algorithm to determine the contact point between the two objects. Without any of the supplemental techniques described in this chapter, the ring and shaft commonly intersect one another leading to simulation failure. Although, the constraint maintenance scheme improves the reliability of object interactions during this assembly sequence, fatal intersections still sometimes occur.

Given great enough computing power, in the form of a faster computer, these penetrations would cease to occur since the collision detection algorithm could be called frequently enough to resolve any collisions before intersection. Although appetizing,

hardware limitations prevent this from being possible. Since a faster computer is not an option, velocity slowdown essentially slows the motion of objects in the simulation, allowing the collision detection routine enough time to clearly identify the closest points between the two and prevent intersection.

In testing (see Section 4.4), even the largest forces applied to the ring through the PHANToMs were unable to cause object overlap when slowdown compensation was active. Although one would not normally attempt to apply such extreme forces to the ring, we want HIDRA to be as robust as possible. The one limitation of the velocity slowdown compensation is that it may detract from the realism of dynamically interacting objects since it is clearly not a physically based model. However, during experimental testing using human subject volunteers (Chapter 5 and 6), a survey was conducted to receive user opinions of HIDRA. None of the volunteers mentioned that they felt objects slowing down as they approached one another, leading to the notion that the velocity slowdown compensation was not overwhelmingly noticeable.

4.2.4 Bounding Spheres for Haptic Dynamic Loading

Previous work in the development of HIDRA largely focused on minimizing the computational load required by the simulation's haptic loop (discussed in Section 2.2.3). One technique implemented was haptic dynamic loading. The GHOST SDK provides interference detection between the virtual representation of the haptic interface and each haptic object in the scene. Even when the PHANToM position is relatively far away from an object, GHOST still checks for interference with the object, which wastefully consumes computation time. Haptic dynamic loading was developed to prevent this time-consuming interference check within the haptic loop by performing a preliminary

collision check between the haptic interface and other objects, similar to broad phase collision detection. First, the simulation defined a simple bounding box as the collision representation for the haptic interface. The collision detection library, a much faster algorithm, then performed an intersection test between the haptic interface's collision representation and other objects in the scene. If no intersection was found between the PHANToM's collision representation and an object, the haptic representation of that object was disabled. By disabling the haptic representation of the object, GHOST would not check for interference between that object and the PHANToM interface, saving computation time during the haptic loop.

The drawback of this technique is that it required the collision detection algorithm to perform additional intersection tests. Initially, the collision algorithm was only responsible for detecting collisions between the virtual objects, not including the PHANToMs. Using this haptic dynamic loading, the collision detection algorithm now had the increased burden of detecting collisions between each PHANToM's bounding box and all objects in the scene. As expected, this degraded the performance of the collision loop and object interactions as precious processor time was consumed for the additional collision checks.

To alleviate this increased burden on the haptic loop, bounding spheres were used to provide broad phase collision between the PHANToMs and other virtual objects in the scene. Recalling from Section 3.3.2, bounding spheres surround an object in its entirety and offer a simplified intersection test. Instead of the collision representation, a bounding sphere was defined surrounding the haptic interface. This bounding sphere could then be used to detect intersections with the bounding spheres of the virtual objects in the scene.

Any object whose bounding sphere did not intersect at least one of the bounding spheres of the PHANToM interfaces would be disabled during the interference check performed by the haptic loop. Not only is this an even faster technique, but it also reduces the computational load placed on the collision detection algorithm. Using bounding spheres for haptic dynamic loading improves performance of both the haptic and collision loops at the same time.

4.2.5 More Efficient Interaction with Simulation Workspace

The workspace in HIDRA consists of the floor, four walls, and a ceiling. The definition and size of this workspace is necessary since the PHANToM interfaces have a limited range of motion. Any object inadvertently pushed outside this workspace would be unreachable, so the walls and ceiling of the workspace must prevent this from occurring. Initially, this was accomplished using the collision detection and response algorithms. Each wall, the ceiling, and the floor had a collision representation. During the collision loop, the collision status between each object and all workspace-bounding surfaces was checked and the appropriate response calculated. Similar to the discussion of haptic dynamic loading in the previous section, these collision checks imposed an unneeded computational load on the collision detection algorithm.

As a replacement for collision detection with the walls and the ceiling, a more simplified method using a spring-back force was implemented. During each iteration of the collision loop, the geometric center of each object is compared to the location of each virtual wall. If the center falls outside the simulation workspace in any direction, a force increasing linearly with distance is applied to the object to push it back into the

workspace. The equation for the force placed on an object by all workspace boundary interactions is shown below.

$$\vec{F}_{wall} = \sum_{all\ walls} (k_{wall} \cdot \vec{\gamma}_{body,wall} \cdot \frac{1\,m}{1000\,mm}) \tag{4.7}$$

In this equation, k_{wall} is the spring constant for each wall and is equal to 10 kg/s^2. The vector $\vec{\gamma}_{body,wall}$ is defined for each wall and describes the perpendicular distance (in millimeters) and direction of the object's geometric center from the wall. If the object is within the workspace bounds defined by a particular wall, the magnitude of this vector is equal to zero. The final term in the summation is a conversion factor, included to insure the units of $\vec{F}_{wall}$ are in Newtons. The force vector defined by interactions with the workspace boundary will never have a vertical component.

No constraints are placed on the ceiling of the workspace, as the force of gravity is sufficient to cause the object to drop down. The floor is the only element of the workspace in which collision detection is continued, allowing accurate collision response of objects as they hit the floor. Again, this modification reduced the computational load placed on the collision detection and response algorithms.

4.2.6 Discussing the Effect of Supplemental Techniques on the Physical Model for Object Interactions

As mentioned, there were three goals addressed during the development of these supplemental techniques: reducing fatal intersections between objects, minimizing the computational load on the collision detection algorithm, and allowing easier object handling. In this section, the integration of these supplemental techniques into the update

of the dynamic state of objects will be discussed, along with their effects on the physical model for object interactions.

Firstly, bounding spheres for haptic dynamic loading are utilized solely for the purpose of reducing the computational load on the collision detection algorithm. As such, this procedure has no impact on either an object's dynamic state or the interaction between objects and will not be discussed further in this section.

Similarly, the goal of implementing a more efficient simulation workspace was to reduce the amount of processing required by the simulation's collision detection library. The only time that this technique affects an object, in any way, occurs when the object travels outside the boundary defined by the workspace. The force delivered to an object traveling outside this boundary is added to the haptic force on the object during force calculations in the haptic loop. The haptic force is modified by this spring-back force using the following equation.

$$\vec{F}_{haptic,modified} = \vec{F}_{haptic} + \vec{F}_{wall} \tag{4.8}$$

When wall interactions are encountered, this modified haptic force is used in the calculation of acceleration in Equation 3.7 discussed in Section 3.3.4. Although this type of wall interaction does not reflect the physical reality, the equation used to calculate the change in velocity of an object (Equation 3.4) remains mathematically valid since $\vec{F}_{wall}$ is a non-impulsive spring force.

All other supplemental techniques, including user defined constraints, elimination of angular velocities, and velocity slowdown, modify the linear and angular velocity

vectors of an object, $\vec{v}$ and $\vec{w}$. The method for applying these modifications for the linear velocity of an object is depicted in Figure 4.12.

```
Modifying Linear Velocity
    Variables:
        vel_wc:     The velocity vector of an object in world coordinates
        vel_lc:     The velocity vector of an object in local coordinates
        mag_vel:    Magnitude of vel_wc
        factor:     Velocity reduction factor for velocity slowdown

    /* Apply User-Defined Constraints on linear motion */
1.  vel_lc = transform_world_to_local(vel_wc)
2.  for (each local axis) {
3.      if (user-defined constraint exists) vel_lc[i] = 0
    }
4.  vel_wc = transform_local_to_world(vel_lc)

    /* Impose Velocity Slowdown */
5.  if (velocity slowdown is active) {
6.      mag_vel = 'magnitude of vel_wc'
7.      if (mag_vel > max_allow_vel) {
8.          factor = max_allow_vel / mag_vel
9.          vel_wc = vel_wc * factor
        }
    }

    /* Final vel_wc applied to object for position update */
```

Figure 4.12: Applying User-Defined Constraints and Velocity Slowdown to Modify the Linear Velocity of an Object

As seen in Figure 4.12, user-defined constraints and velocity slowdown directly modify the velocity vector of an object. For user-defined constraints the object's velocity vector is transformed into local coordinates, and any constraints cause this velocity along a particular local axis to be set to zero. After any modifications, the new velocity vector is transformed back into world coordinates. Assuming velocity slowdown is active, this new velocity is again modified to restrict its magnitude to a value equal to

max_allow_vel (defined in Equation 4.6) when necessary. The resulting velocity is then used to update the object's translational position in the environment.

The angular velocity of an object is modified by user-defined constraints and elimination of angular velocity during collisions, as shown in Figure 4.13. Here, transformations of the angular velocity are not necessary since angular rotation is defined with respect to the objects local axes. First, the angular velocity around each axis is set to zero for any rotational constraint that exists. Next, the simulation checks to see if any dynamic constraints, which result from interactions between objects, exist. If so, then all angular velocity of the object is eliminated, assuming the option is enabled.

```
Modifying Angular Velocity
    Variables:
        angvel:     The angular velocity vector of an object
        factor:     Velocity reduction factor for velocity slowdown

    /* Apply User-Defined Constraints on rotational motion */
1.  for (each local axis) {
2.      if (user-defined constraint exists) angvel_lc[i] = 0
    }

    /* If any constraints exist on object, eliminate angular velocity */
3.  if (any constraints/collisions exist on object) {
4.      if (angular velocity elimination enabled) {
5.          angvel_wc = 0
        }
    }

    /* Final angvel_wc applied to object for orientation update */
```

Figure 4.13: Applying User-Defined Constraints and Elimination of Angular Velocity to Modify the Angular Velocity of an Object

To this point, the method for modifying the linear and angular velocity of an object, with respect to user-defined constraints, velocity slowdown, and elimination of

angular velocity during collisions, has been discussed. The effect of these techniques on the physical model for object interactions will be discussed next. Clearly, none of these techniques is physically valid for modeling the real world, so the focus of the discussion will be on their limitations.

As mentioned in Section 4.2.1, user-defined constraints provide two benefits: easier object handling with the dual-PHANToM setup and reduced processing required by the collision detection library due to extreme jumps in the point of closest approach between two objects. The technique for elimination of rotational motion during collisions also helps to reduce the computational load on the collision detection algorithm for the same reason. The obvious limitation in activating user-defined constraints or eliminating angular velocity during collisions is that the interaction between two objects is less realistic, particularly for rotational response to a collision. For purely translational assemblies or disassemblies, eliminating angular velocity or defining rotational constraints has a limited effect on the user's ability to perform the operation. However, for assemblies or disassemblies requiring rotation of an object, their affects are more detrimental. In fact, if elimination of angular velocity is active and an object must be rotated into place while touching or resting on other objects, the task is not possible. This is a major limitation of eliminating the angular velocity of an object during a collision.

Velocity slowdown has a much different effect on object interactions. The obvious effect is that insertion or removal of an object to or from an assembly will be slowed, increasing assembly or disassembly time. Also, as a side effect, the magnitude of the impulse response resulting from a collision between two objects will be less significant since the closing velocity is reduced. The reduced velocity of objects in close

proximity is considered to be the most detrimental limitation of all supplemental techniques because it directly affects assembly and disassembly times.

Although drawbacks exist, the combination of all of the supplemental techniques discussed in this section and Section 4.1 greatly improve object interactions in HIDRA, as will be seen in tests later in this chapter and in Chapters 5 and 6.

4.3 CONSIDERING HIGH-LEVEL SIMULATION ARCHITECTURE

Research Question 1.2 focuses on the high-level simulation architecture of HIDRA, determining which layout of the haptic loop, collision loop, and graphics processing allows for the most efficient use of computer processing power. In this section, we describe and explore the differences between three different simulation structures.

4.3.1 Simulation Design from Previous Work

The original simulation architecture stems from the initial development of HIDRA and the work conducted prior to this research (McDermott 1999). Recalling from Section 2.1, the haptic loop of HIDRA demands the greatest in processing power, as haptic updates must occur with a frequency of no less than 1000 Hz, thus ensuring the stability of haptic feedback. For this reason, the GHOST SDK controls the timing of all haptic processing to assure that this frequency requirement is satisfied. In fact, the GHOST SDK triggers an update of the haptic loop exactly every 1 millisecond, as more frequent updates would undesirably and unnecessarily flood the computer's processor with excessive computations. As such, after initialization of the haptic scene, GHOST maintains all control over the execution timing of the haptic loop. This feature is common to all three variations of the simulation architecture discussed in this section.

In Figure 4.1, a representation of the initial simulation architecture is presented. As mentioned, GHOST creates a separate high-priority process to handle haptic interactions at the beginning of the simulation. The graphics and collision routines are maintained by the GLUT window management. Upon execution of GLUT's main graphical loop, the routine never returns control to the calling program. As a result, all further routines are carried out using GLUT callback routines that execute on a periodic basis. The original version of HIDRA utilizes a single graphics update callback to execute both the graphic display and collision detection routines. During the update, object positions are first queried from the haptic loop, which maintains all object states. After retrieval of these positions, collision detection and response are performed, and the response to any collisions is returned to the haptic loop for processing. Finally, the graphical display of objects is refreshed using the object positions previously retrieved. User interaction is provided through the haptic interfaces, the keyboard and mouse, and a visual display.

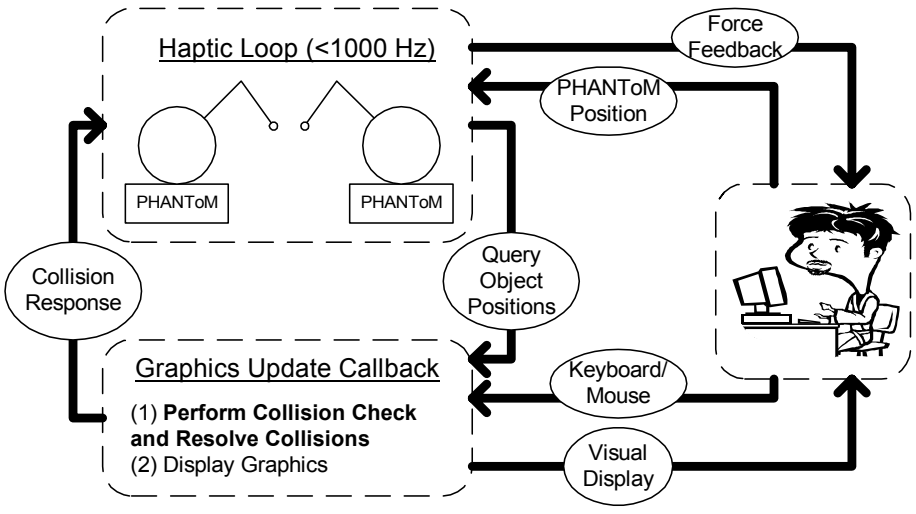


Figure 4.14: Original Simulation Architecture

4.3.2 Separation of Collision and Graphic Loops Using GLUT Timers

The main limitation of the original simulation architecture, which will be realized in Section 4.3.4, is the fact that the execution of collision loop is linked directly to the graphical display using the graphics update callback of GLUT. As with all computer-generated environments, the necessary graphics refresh rate to maintain a realistic visual environment is typically around 30 Hz. Ideally, the collision detection and response processes should be capable of much faster execution frequency. Given the original simulation layout, both processes are compromised as the collision loop runs slower than desired and the graphics refresh rate is faster than necessary. As a result, another version of HIDRA was created using GLUT timer callbacks to separate the collision and graphics loops.

The separation between the collision and graphics loops, achieved using GLUT timer callbacks, is depicted in Figure 4.15. Again, the haptic loop runs as a separate process and is maintained by GHOST. In this simulation configuration, however, the collision loop and graphics display are executed using individual callbacks. GLUT timer callbacks allow the simulation to specify a predetermined refresh rate for execution of a given routine. In this configuration, the collision loop is executed on a continual basis, assuring that the collision routines are performed as fast as possible. The frequency of calls to the graphics update loop, on the other hand, is set to 30 Hz, the typical rate necessary for a realistic visual sensation. Upon execution of the collision loop, object positions are queried from the haptic loop, collision detection and response is performed, and any response information is reported back to the haptic loop. The graphics update routine obtains the most recent object positions from the collision loop for object display,

rather than querying the haptic loop again, as this is a more time-consuming process. The limitation of using GLUT timer callbacks is that only one callback may be executing at a time, meaning the collision and graphics loops are still somewhat dependent on each other as they cannot run in parallel. Interactions between the user and the simulation remain the same.

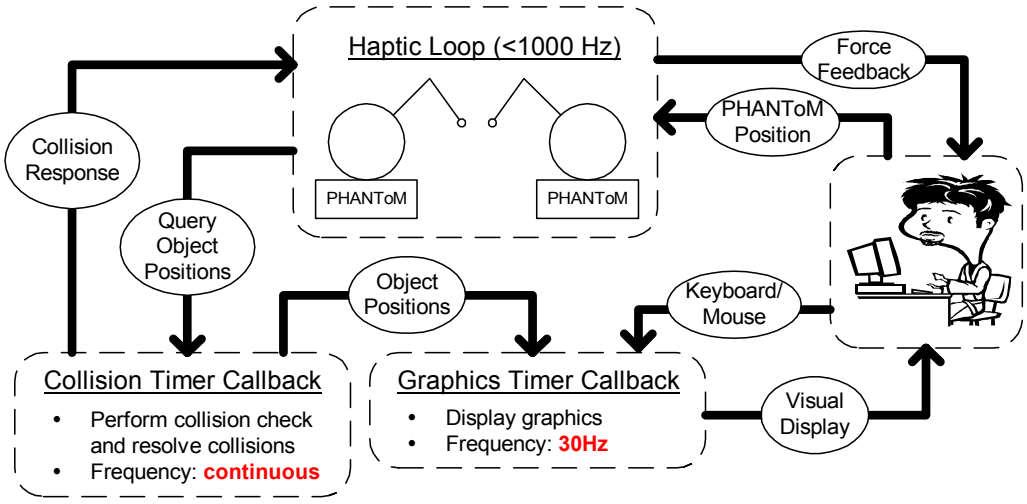


Figure 4.15: Simulation Configuration using GLUT Timer Callbacks

4.3.3 Integrating POSIX Threads for Improved Performance

The last version of the simulation implementing a unique overall configuration of the haptic, collision, and graphics loops uses POSIX® threads[1]. POSIX® threads, also know as Pthreads, allow an executable to run multiple tasks simultaneously. Unlike either of the previous implementations, the various components of the simulation can be run in parallel, maximizing the performance of each loop. In all versions of the simulation, the haptic loop is essentially running in parallel with the collision and graphics loops. Although the previous version of the simulation uses separate GLUT timer callbacks, the

[1] POSIX is a registered trademark of the IEEE and is based on the IEEE POSIX 1003.1c-1995 standard.

collision loop and graphics loop can never be executing at the same time, as one process will be queued and will wait for the other to complete before beginning. Removing this dependence between the collision and graphics loops by using Pthreads improves overall simulation performance.

The overall simulation architecture using Pthreads, shown in Figure 4.16, is very similar to the previous implementation. In this configuration, however, the collision loop is no longer controlled by a GLUT timer callback. Instead, the execution of the collision routine is controlled by the haptic loop using Pthreads. The maximum frequency necessary for calling the collision loop is bounded by the speed at which the haptic loop operates. For instance, assume that the haptic loop maintains a cycle time exactly equal to 1 millisecond, during which the positions of all objects are updated. Then collision detection and response only need to be calculated at the same frequency, namely 1000 Hz. If the collision loop is executed more frequently, object positions will not have changed from one iteration to the next. This means that the results for collision detection and response will be the same, and precious computer processing power is wasted.

This problem is alleviated with the Pthread implementation of HIDRA. A new thread is created to perform collision detection and response processes based on the following two conditions: (1) the haptic loop has just completed updating object positions and (2) all previous threads for collision detection and response have completed processing. This assures that the collision loop is executed as frequently as possible, but no faster than the haptic loop updates object positions. There is one exception to this rule that is required as a result of haptic dynamic loading (discussed in Section 2.2.3), which unloads the haptic representation of an object if the PHANToM virtual fingertips are

greater than a certain distance from the object. If all objects in the scene are motionless, and their haptic representation is unloaded, the haptic loop remains idle since object positions will not change. The collision loop, therefore, must be executed manually to perform a proximity check between the PHANToM representations and the haptic objects in order to determine when the objects should be loaded back into the haptic scene. This form of fault checking is performed using a lower frequency GLUT timer callback that checks to make sure the haptic loop is still running. If not, a separate thread is created to execute the collision loop, thus performing broad phase collision detection between the PHANToMs and the haptic objects. Since objects are motionless when this occurs, there is no effect on simulation performance.

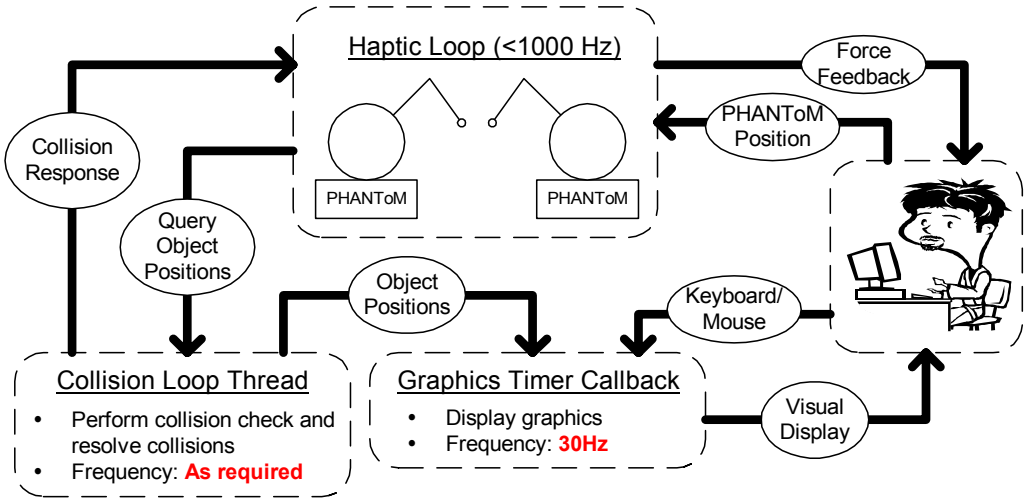


Figure 4.16: Simulation Configuration using POSIX Threads

4.3.4 Performance Comparison and Discussion of Simulation Architectures

Given the three versions of the simulation, differing in overall configuration of the haptic, collision, and graphics loops, a study was performed using the simple ring and

shaft (see Figure 4.11). The goal was to determine the effect each simulation architecture had on the efficiency of the collision loop. As mentioned in the previous section, the collision loop ideally executes exactly one time for each update of the objects positions by the haptic loop. Additional calls to the collision routines are unnecessary and less frequent calls could result in undetected collisions prior to inter-object penetration and simulation failure.

For each version of the simulation, the task performed was the assembly of the ring and shaft. Prior to assembly, the shaft was anchored and remained motionless during the entire sequence. The following data was recorded: time between successive iterations of the collision loop and number of haptic loop iterations between collision loop iterations. For all scenarios, haptic loop partitioning (Section 2.2.3) was enabled which means that the calculations required to update object positions, including computing the forces, accelerations, and velocities, was performed over a total of ten haptic loop iterations. This means that object positions are updated approximately every 10 milliseconds, and collision detection and response should be performed, ideally, at the same frequency.

In Figure 4.17, the number of completed haptic cycles between successive iterations of the collision loop is shown as a frequency of occurrence. For the sake of this discussion, a completed haptic cycle is defined as the time when object positions are updated, signifying the completion of 10 iterations of the haptic loop. For instance, the first column in the figure shows the percentage of time that zero haptic cycles were completed between successive iterations of the collision loop. The second column represents the frequency that one haptic cycle was completed between collision loop

iterations, etc. This data is shown for all three versions of the simulation discussed above, denoted as the Original, GLUT Timers, and Pthread versions.

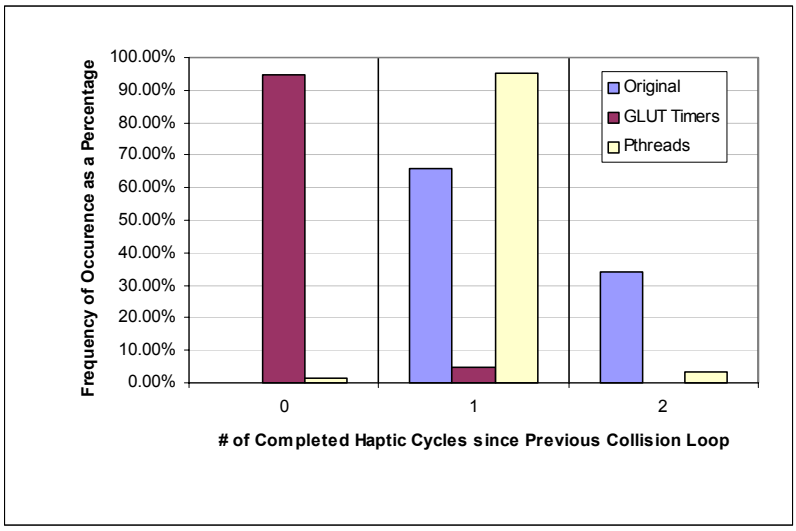


Figure 4.17: Number of Completed Haptic Loop Cycles between Successive Collision Loop Iterations as a Frequency of Occurrence.

Given the data in Figure 4.17, several differences between the versions of HIDRA become apparent. These differences are discussed below.

- In the original version of the simulation, one completed haptic cycle between successive iterations of the haptic loop occurs about 65% of the time and two completed haptic cycles occurs about 35% of the time. This demonstrates that the time between collision loop iterations is too large, and is a direct result of the fact that the frequency of collision loop iterations is directly dependent on the graphics update callback frequency.
- On the other end of the spectrum, the iteration frequency of the collision loop in the GLUT Timer version of the simulation appears to be too fast. Almost 95% of

all collision loop cycles are unnecessary since zero haptic cycles were completed and objects positions had not changed.

- As expected, the Pthread version of HIDRA performed most efficiently, with respect to number of completed haptic cycles between collision loop iterations. Slightly more than 95% of the time, exactly one haptic cycle was completed between iterations of the collision loop.

The ideal situation is that there is always one completed haptic cycle between collision loop iterations. The reality, however, is that this ideal situation is difficult to achieve given the nature of real-time simulation. The Pthread version of HIDRA had the highest percentage of one completed haptic cycle between collision loop iterations when compared to the other two versions. This suggests that the Pthread simulation architecture provides the capability for the most efficient integration of the collision detection and response routines with the rest of the simulation.

In addition to the data collected above, the average time between successive collision loop iterations was computed and is shown in Table 4.1. First, notice that the GLUT Timer simulation version had an average collision loop iteration time of 1.4 milliseconds, much less than the haptic cycle time of 10 milliseconds, again highlighting the wasted processing time. The average collision loop cycle time for the Pthread version of HIDRA was 10.3 milliseconds. This is very close to the haptic cycle time and supports the conclusion that the Pthread implementation performed most efficiently. The original simulation architecture had an average collision loop cycle time of 13.9 milliseconds, which also corresponded to the graphics refresh rate.

Table 4.1: Average Collision Loop Iteration Time for each Simulation Architecture

	Average Collision Loop Cycle Time (ms)
Original	13.866
GLUTTimers	1.398
Pthreads	10.315

The data presented in this section supports the notion that Pthreads, or thread programming in general, are a very efficient method for integrating collision detection and response algorithms within the rest of a simulation. It should be noted that the simulation architecture using GLUT timer callbacks could possibly be improved by reducing the predetermined frequency of timer callbacks for the collision loop to 10 milliseconds, or the approximate time period of one complete haptic cycle. This would surely increase efficiency of the configuration by reducing excessive calls to the collision loop. However, there are two major problems with this strategy. Although haptic cycle completion time is expected to be around 10 milliseconds, this value can vary and can never really be predetermined exactly. As such, using a GLUT timer callback to schedule the collision loop would still be guesswork and less efficient than using Pthreads, which are initialized upon completion of the haptic cycle by the haptic loop itself. Second, and more importantly, the collision loop will never be able to run in parallel with the graphics display update as long as GLUT timer callbacks are controlling the execution of both loops.

Many other improvements are possible to speed up computation and improve efficiency such as a faster computer or a multiple computers, one being dedicated solely to graphics display, etc. However, with respect to the software architecture of the simulation, thread programming provides one of the most flexible options to integrate

various programming loops such as collision detection into a real-time simulation. From this point on in this dissertation, all references to HIDRA will refer to versions implementing Pthreads for improved simulation architecture.

4.4 A PERFORMANCE COMPARISON BETWEEN V-CLIP AND SWIFT++

In Chapter 3, the concepts of physically based modeling and collision detection for impulse-based simulations were introduced. Also, a discussion of current collision detection libraries included a qualitative comparison of the most promising libraries for implementation in an assembly and disassembly simulation with haptic feedback. In this section, a quantitative comparison of the two most promising collision detection libraries resulting from that review, V-Clip and SWIFT++, will be presented. The goal in this study was to make timing comparisons between V-Clip and SWIFT++ as collision detection algorithms in HIDRA and analyze the performance of each library, including computational speed and usability. The expectations were that the capability of SWIFT++ to return multiple collision points would result in better performance of object interactions. In addition, the constraint maintenance scheme and other techniques for supporting collision detection will be tested, as the overall performance of the assembly task will be compared to the same assembly scenario used for testing in previous work (McDermott 1999). In following sections, the experimental setup and testing procedure is described, followed by a discussion of the results.

4.4.1 Setup and Description of Quantitative Comparison

For performance comparison, we created 3 assembly scenarios. Each task consisted of sliding a ring on a shaft. Although seemingly trivial, the interaction between non-convex objects becomes much more complex in a real-time virtual environment,

especially with haptic feedback as external forces created by the user are unpredictable. The difference between each scenario rests in the complexity of the ring and shaft. In increasing order of geometric complexity, we have ring and shaft pairs with a square hole, a hexagonal hole, and a round hole (Figure 4.18, Figure 4.19, and Figure 4.20). To give an idea of the size of each object, the overall width of the ring is 70 millimeters, and the shaft height is 75 millimeters. There is a 2-millimeter gap between each ring and shaft combination on all sides when assembled and centered.

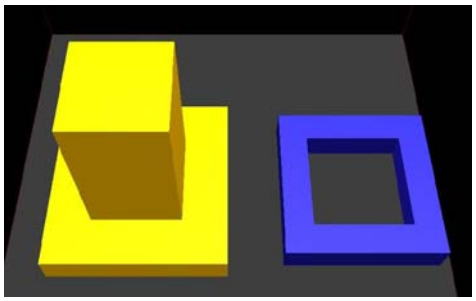


Figure 4.18: Square Ring and Shaft

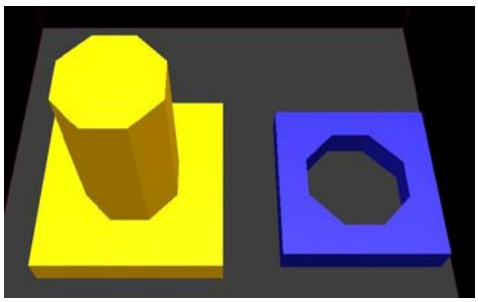


Figure 4.19: Hexagonal Ring and Shaft

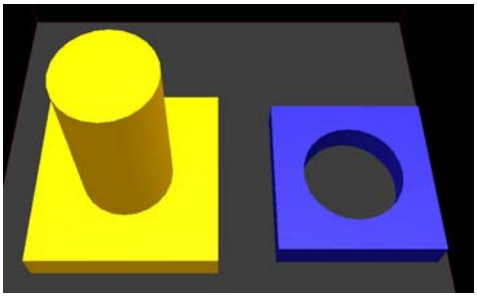


Figure 4.20: Round Ring and Shaft

Each of these parts, as described in Section 2.2.1, were created in ProEngineer and exported to VRML 1.0 format. The objects were then decomposed into convex pieces. The first notable difference between V-Clip and SWIFT++ was in preparing the virtual objects for HIDRA, as each library requires different formats for reading object geometries. SWIFT++ requires that a decomposer program, included with the library, process all non-convex objects, whereas the task must be done manually within HIDRA when using V-Clip. The resulting number of convex pieces created through each decomposition process is shown in Table 4.2. Clearly, decomposition performed by SWIFT++ results in more convex components, but objects like the round ring require several minutes of manipulation for use with V-Clip. The difference in the number of convex pieces created for each part, and its effect on collision detection performance, will be discussed in more detail when we analyze the results of our study.

Table 4.2: Convex Decomposition of Virtual Objects

Parts	# Convex Pieces	
	V-Clip	*SWIFT++*
Square Ring	4	16
Square Shaft	2	12
Hexagonal Ring	10	24
Hexagonal Shaft	2	20
Round Ring	32	57
Round Shaft	2	55

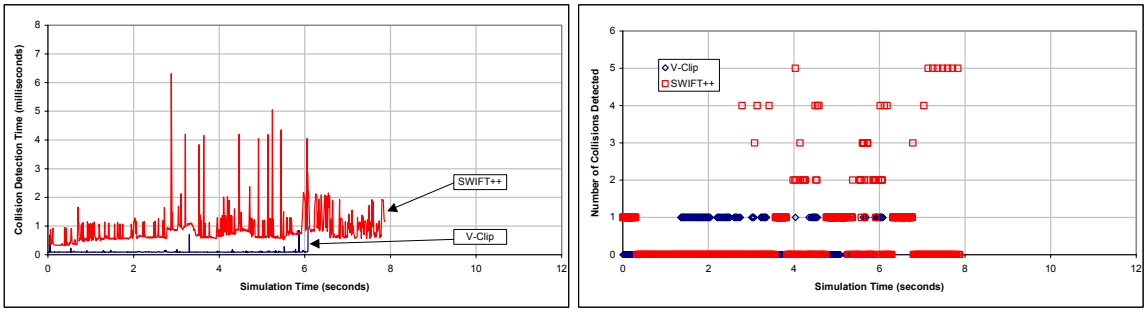
After the preprocessing described above, the assembly sequence of placing each ring on its corresponding shaft was performed. For all tasks, the shaft was anchored to the ground and the ring was constrained to translation only. An effort was made to 'jiggle' the ring as it slid down the shaft, attempting to cause object intersection and, hence, simulation failure. In essence, we were trying to test each collision detection algorithm to its limits for each scenario. In support of virtual object interactions, dynamic constraints and slowdown of objects in close proximity were active (see Sections 4.1 and 4.2).

4.4.2 Results and Analysis

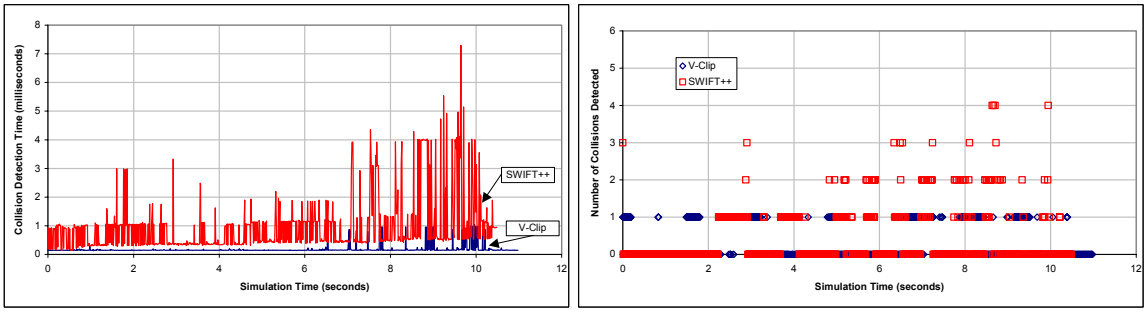
Each collision detection library was queried for exact separation distance and closest points. V-Clip provides the exact global distance between two objects and the associated closest points. SWIFT++ also reports exact distance and associated collision points, but may return such information for multiple collision pairs between the same two objects, correlating to locally closest points, which is possible since the objects are non-convex (see Section 3.2.4).

The results for all scenarios are shown in Figure 4.21. In this figure, each subsection (a-c) represents data from one of the three scenarios described above (i.e., square, hexagonal, and round ring and shaft assembly). For each scenario, the graph on the left shows raw timing data taken directly from the simulation. All times were

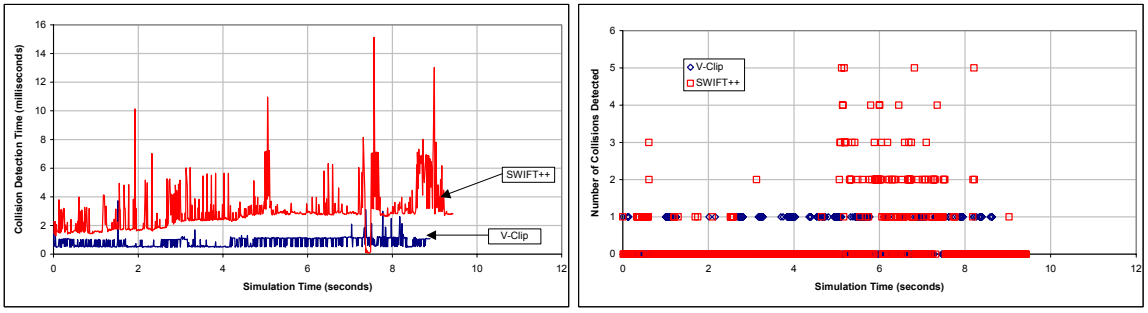
collected using an SGI specific clock timer, *clock_gettime()*, with a resolution of 80 nanoseconds. During each collision loop, the clock time was captured immediately before and after the query to each collision detection library, so the data displayed corresponds to the time required by the library to determine exact distance and closest points. All data was stored in an array for output at the completion of the scenario to avoid time-consuming operations associated with writing to a file. The graphs in the right hand column show the number of closest point pairs found by the given collision detection algorithm for each scenario. As mentioned, the V-Clip library will only find the globally closest point pair, whereas SWIFT++ may report more than one pair. The abscissa for all plots corresponds to elapsed simulation time since first collision. Notice that this value ends at a slightly different value for all scenarios but is irrelevant to our analysis. Now that we have clarified what the data in Figure 4.21 represents, the results will be scrutinized.



(a) Square Ring and Shaft



(b) Hexagonal Ring and Shaft



(c) Round Ring and Shaft

Figure 4.21: Simulation Data for Assembly Scenarios

As can be seen in Figure 4.21, the query time required by SWIFT++ to determine the exact separation distance(s) is consistently more than that required by V-Clip. There are several factors leading to this difference in computation time. First and foremost, SWIFT++ searches for all local minimum distances, whereas V-Clip only finds the location of global minimum distance. Looking at all scenarios in Figure 4.21, we can see that SWIFT++ returned as many as five simultaneous collision points during the assembly sequence. However, the frequency of such occurrences was somewhat limited

as multiple collisions occurred during only 7-9% of the queries to SWIFT++, increasing slightly with complexity of the objects. Furthermore, the existence of any collision under the 0.25-millimeter collision threshold discussed in Section 4.1.1 occurred in approximately 20-26% of the queries. In the version of HIDRA implementing V-Clip, the frequency of collisions detected was between 30-35%. The increase in collision frequency is attributed to our supplemental constraint maintenance scheme, which employs motion constraints between two colliding objects. When multiple collisions are detected using SWIFT++, constraints are added along all collision axes, further limiting motion and subsequent collisions. When only one collision constraint is possible, as with V-Clip, subsequent collisions are more likely to occur.

Another factor that may have increased the query time for SWIFT++ is the collision representation of each object (see Table 4.2). Each part, the ring and the shaft, are composed of a number of convex pieces. In all cases, the number of convex pieces required by SWIFT++ is much more than that required by V-Clip. This can be attributed to the method by which the collision representations are constructed. As mentioned in Chapter 3, SWIFT++ requires that each non-convex object be decomposed into convex pieces using a supplemental program that comes with the library, and reducing the number of components is not possible. Using V-Clip, manual decomposition is required, but the number of pieces making up each object is drastically reduced. There is clearly a trade-off between an automated program that delivers a larger number of components versus manual decomposition, which may eventually become impractical with increasing scene complexity. It should be mentioned that the preprocessing techniques required by

SWIFT++ for non-convex objects are designed to allow more efficient execution of the library.

Referring to Figure 4.21(c), an abnormality exists in the *Collision Detection Time* for the assembly of the round ring and shaft when using the SWIFT++ version of HIDRA. Between 6 and 8 seconds *Simulation Time*, the query time for SWIFT++ drops to a very small value, just above zero. This decrease represents a penetration between the round ring and shaft, and occurred as the ring reached the base of the shaft. SWIFT++ very quickly detected that the two objects were intersecting, and returned control to the simulation immediately, without reporting any distance or closest points. Fortunately, during this assembly sequence, with assistance from the constraint maintenance scheme, separation between the two objects was achieved and the simulation continued without failure. This penetration was caused by the relatively large amount of time required by SWIFT++ to detect multiple collisions between the ring and shaft. This difficulty would only be compounded for larger, more complex assembly scenarios. The assembly scenarios using the V-Clip version of HIDRA were completed without incident, and average query times were lower by several factors. A comparison of the query times for the two libraries tested will be discussed next.

In Table 4.3, we show the average query times during those queries when collisions are detected between the ring and shaft. These averages do not include queries when no collisions are detected, although those times are very similar in all cases. We present the data in Table 4.3 to show the difference between total time required and time required per collision found. This value will always be the same for V-Clip (column 1) because its algorithm only returns one pair of closest points for any two objects. Since

SWIFT++ sometimes reports multiple collision pairs, the query time per collision (column 3) is lower than the overall average query time (column 2). Under the scenarios tested, V-Clip consistently performs at a faster rate than SWIFT++, even on a 'per collision' basis. Again, this can be attributed to the fact that SWIFT++ must descend all convex nodes of the non-convex objects to check for local minimum distances under the desired collision threshold rather than searching for the smallest separation distance globally. In essence, there is a penalty in the form of computation time for the capability to find simultaneous collisions between two objects.

Table 4.3: Average Query Time When Collision(s) Detected

Scenario	Average Query Time (milliseconds)		
	V-Clip	SWIFT++	SWIFT++ (per collision)
R&S	0.102	0.886	0.727
Hex R&S	0.166	0.815	0.667
Round R&S	0.912	2.810	1.894

4.4.3 Discussion

The assembly sequences conducted in this comparison contained two non-convex objects. We could have conducted similar experiments using only convex objects. However, the utility of doing this would be severely limited since most objects in reality are non-convex in nature. In terms of our application, assembly does not make much sense without at least one non-convex object and typically more.

Aside from a selection process, the study comparing V-Clip and SWIFT++ was helpful in addressing Research Questions 1.1 and 1.2. The implementation of each library and subsequent quantitative comparison highlighted the usability and functionality of the two collision detection libraries in the following ways:

- The implementation of SWIFT++ in HIDRA is less complex than V-Clip, since the former is an N-body collision detection algorithm and the latter is a pair-processing algorithm. When using an N-body algorithm, the simulation must only update the object positions and query the library for collision information one time. When using pair-processing algorithm, on the other hand, the simulation must query the library for collision information for each pair in the scene. The main advantage an N-body collision detection library has is that broad phase collision detection can be incorporated into the algorithm. In SWIFT++, this is accomplished using simple cubic bounding volumes. As mentioned previously, for the V-Clip version of the simulation, we implemented bounding spheres. Although slightly more programming is required when using a pair-processing algorithm, the advantage is greater flexibility for the simulation.
- As one would expect, there is a substantially large computation penalty for the capability to find simultaneous collisions between two non-convex objects. In the assembly experiments discussed above, a small collision detection time proved to be more beneficial than multiple collisions as the V-Clip version of HIDRA performed flawlessly and the SWIFT++ version had minor difficulties with the round ring and shaft.
- The method by which objects are decomposed into convex pieces can become very important as assemblies increase in complexity. Manual decomposition, as required by V-Clip, would eventually become impractical, and an automated decomposition algorithm, such as that used with SWIFT++, would be necessary.

By conducting the experimental comparison, we have highlighted some of the trade-offs in using either V-Clip or SWIFT++, or similar collision detection libraries.

With respect to Research Question 1.2, the value of the constraint maintenance scheme and other collision detection were readily apparent during the assembly scenarios. During previous work, the performance of V-Clip alone, for the assembly of the relatively simple square ring and shaft, was sometimes unreliable, as unrecoverable object intersections would result in simulation failure. By implementing and activating these supplemental techniques for object interaction, the simulation has become much more robust as the assembly scenarios discussed above can be completed with, for all intensive purposes, a 100% success rate (using the V-Clip version of HIDRA). The assembly scenarios to be presented in Chapter 6 will also demonstrate the value of the supplemental techniques developed, as the assemblies carried out were previously unfeasible.

The experiments and analysis carried out in this section was not only helpful in deciding which particular library is most well suited for HIDRA, but also highlights the advantages and drawbacks of the usability and functionality provided by each collision detection library. During this comparison, the V-Clip version of our simulation provided the necessary capabilities to model object interactions without undesired intersections while requiring much less computation time. This proved to be the most important factor when comparing collision detection libraries for our real-time haptic simulation. However, as the ability to model larger and more complex objects increases, manual decomposition may no longer be practical and an automated preprocessor would be needed.

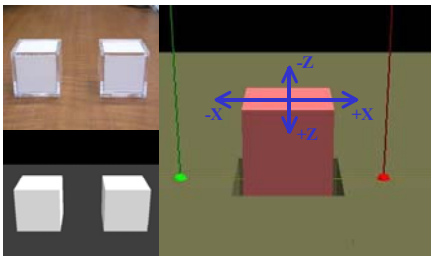
4.5 CHAPTER SUMMARY

In this chapter, focus continued on Research Question 1 and the modeling of interactions between virtual objects in HIDRA. In Section 4.1, a method for constraint maintenance between virtual objects, specifically designed to supplement collision detection in an impulse-based simulation, was introduced. Discussion of the constraint scheme included HIDRA's definition of a collision and the maintenance and application of constraints between the objects. Additional techniques in support of collision detection, including user-defined constraints, velocity slowdown, and more efficient object interactions with the simulation workspace, were described in Section 4.2. In Section 4.3, the method of integrating collision detection and response into the HIDRA simulation was put to the test. Three methods were implemented, compared, and tested for efficiency, and thread programming was determined to be the best option for constructing the high-level simulation architecture for improved object interactions. Finally, in the previous section, a quantitative comparison was performed between two leading collision detection libraries, V-Clip and SWIFT++. The comparison not only highlighted the performance of each library during a simple assembly scenario, but also exposed some of the advantages and disadvantages of their differing implementations.

In Chapters 5 and 6, the focus of this dissertation will shift to Research Question 2, as experiments are conducted to assess the value of haptic feedback in simulation for assembly and disassembly simulation.

CHAPTER V

CHARACTERIZATION EXPERIMENTS IN HIDRA



With the completion of Chapter 4 and the work involved in directly addressing Research Question 1 and improving object interactions, HIDRA can be utilized as a test-bed application to test the hypotheses relating to Research Question 2 and the usefulness of haptic feedback in simulation. Prior to addressing the usefulness of haptic feedback for assembly and disassembly evaluation, however, characterization experiments will be conducted. In this chapter, two experiments will be described, each one assessing the utility of haptic feedback for particular aspect of virtual environments and simulation.

5.1 GOALS FOR CHARACTERIZATION EXPERIMENTS

Although the main application focus in developing is assembly and disassembly analysis, this chapter will focus on preliminary investigations into the effectiveness of haptic feedback in a virtual environment. There are many things that haptic feedback can add to a simulation, including the sensation of weight, inertia, friction, compliance, etc. In addition, by incorporating object interactions into a simulation, haptic feedback can allow you to feel the interaction between two objects as you move one through space. In this chapter, two experiments will be conducted, each corresponding to a different research question.

The first experiment will characterize weight sensation in a virtual environment by comparing the ability to perceive weight differences in the virtual world with that in the real world. The main goal will be to determine whether the difference in weight between two virtual objects can be perceived through haptic feedback as well as and in comparable time to the perception of the weight difference between two real objects. At the same, we will attempt to determine whether the weight of a virtual object under a virtual gravity constant of 9.8 m/s^2 is perceived to be the same weight as a real object with identical mass.

In the second experiment, the value of haptic feedback for sensation of motion tolerances will be studied. Tests will determine whether the force feedback provided through the PHANToM haptic devices can assist a user in determine the direction of greater allowable motion for a peg constrained to move within a hole. Also, the value of stereoscopic viewing for the same purpose will be studied concurrently.

The discussion of the weight sensation experiment will be contained entirely in Section 5.2, and the motion tolerance experiment in Section 5.3. In Section 5.4, a summary of the chapter will be provided.

5.2 A STUDY ON WEIGHT PERCEPTION

The implementation of haptic technology in a virtual environment provides the capability to sense weight, which is not possible in a purely visual environment. The experiments in this section explore the effectiveness of weight sensation in the virtual environment when compared to that in the real world. Ideally, the sensation of weight and the ability of a user to sense a weight difference between two objects in a virtual environment with haptic feedback are the same as with real objects. The goal of the experiments described is to evaluate weight sensation in a virtual environment by answering Research Question 2.1:

> *RQ 2.1 – How does the perception of weight in a virtual environment with haptic feedback compare to that in the real environment?*

and the corresponding hypothesis:

> *H 2.1 – The perception of weight and ability to distinguish between weights in a virtual environment closely matches that in the real environment.*

To test this hypothesis, three experiments were performed. The experimental procedures, findings, and validation of results are presented in the next three sections. In Section 5.2.3, the second experiment will be revisited for reasons to be explained. Finally,

feedback from experiment participants and a summary of the results as they relate to the sub-hypothesis will be discussed.

Initial motivation for this experiment was partially inspired by a similar experiment on weight sensation using a haptic device. Gurocak *et al.* designed a force feedback device known as AirGlove and performed weight sensation experiments (Gurocak *et al.* 2003). The AirGlove is a device attached to the users hand, and five thrusters provide force sensation. In their experiment, five participants were asked to compare two identically sized cubes with mass differences ranging from 50-600 grams. The experiments were duplicated in the real and a virtual environment. The study found that decision time was much smaller for real cubes. In addition, all mass differences were correctly identified for the real cubes. On the other hand, cubes were identified correctly in only 60% of the trials of virtual cubes with a mass difference of 50 grams. A limitation of this haptic device is that the thrusting device and resulting forces are applied to the back of the user's hand. It is known that weight discrimination is finer when objects are pickup up using the fingers rather than the whole hand. In the following sections, we conduct a similar experiment to measure the fidelity of weight sensation in a virtual environment. In our experiments, we use HIDRA and the PHANToM haptic devices, which apply forces to the user's fingertips. The results of this experiment demonstrate this increased sensitivity when using the fingers.

5.2.1 Methods and Procedures

In this section, the methods and procedures for experiments on weight sensation will be presented.

Experiment 1 – Real Cubes

This experiment was designed to test the ability of a person to distinguish small weight differences between two cubes of the same size.

Setup: The experimental setup consisted of 12 cubes measuring 50.8 millimeters along each side. One cube was designated as the base cube with a mass of 100 grams. The mass of each of the remaining cubes, designated as test cubes, ranged between 80 grams and 120 grams in 4-gram increments (i.e., 80g, 84g … 116g, 120g). Therefore, one of the test cubes weighed the same, 100 grams, as the base cube. To eliminate any false perception of weight, all cubes were the same in appearance and color. A picture of the real cubes used in this experiment is shown in Figure 5.1.

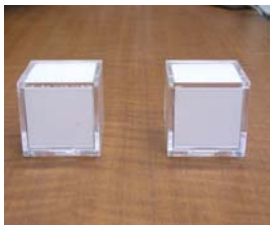


Figure 5.1: Two Real Cubes

Procedure: The user was presented with two cubes at a time, the base cube and a test cube. When asked, the user picked up one cube at a time using the thumb and forefinger. The comparison always began by lifting the base cube first, although either cube could be lifted as many times as required. In addition, the same hand was used to lift both cubes throughout the experiment. The user was to determine whether the test cube was heavier, lighter, or equal to the base cube in weight. This procedure was repeated for all test cubes. The decision time and response were recorded for analysis. Each

experimental subject was given the same set of cubes in a random order. The typewritten instructions given to each participant is available in Appendix A.1.

Experiment 2 – Virtual Cubes

This experiment was designed to test the ability of a person to distinguish small weight differences between two cubes in a virtual environment and to compare this with performance in a real environment (Experiment 1).

Setup: This experiment consisted of 12 virtual cubes measuring 50.8 millimeters along each side. Again, the mass of the base cube was 100 grams and the test cubes ranged from 80-120 grams in 4-gram increments. During this experiment, the user inserted his thumb and forefinger into the thimble of each haptic device of our dual-PHANToM setup. The virtual cubes were modeled in HIDRA and manipulated by the user's virtual fingertips, represented as small spheres. An image of the virtual cubes is shown in Figure 5.2.

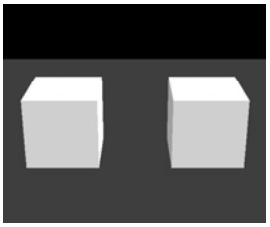


Figure 5.2: Two Virtual Cubes

Procedure: The procedure from Experiment 1 was duplicated, wherein the user judged if the test cube was heavier, lighter, or equal to the base cube in weight. The mass of the virtual test cube was modified between comparisons to simulate a new cube. The order of the test cubes was again randomized, and decision time and response were

recorded. The same set of instructions as Experiment 1 was given, with the caveat that the cubes were modeled in the virtual environment (Appendix A.1).

Experiment 3 – Real vs. Virtual

The objective of this experiment was to determine how closely the sensation of weight in a virtual environment matched that in the real world.

Setup: The user was presented with one real cube and one virtual cube, each with a mass of 100 grams. The appearance of each cube was identical and the dimensions matched those given in the previous experiments. The real cube was designated as the base cube, while the virtual cube was the test cube. The weight of the test cube was modified between trials by changing the gravity constant in the virtual environment. A total of fifteen gravity values were tested ranging from 6.3 to 13.3 m/s^2 in increments of 0.5 m/s^2 and including 9.8 m/s^2.

Procedure: Similar to the first two experiments, the user was asked to compare the weight between the two cubes, specifying whether the virtual cube was heavier, lighter, or the same weight as the real cube. However, in this task the subject used both hands, one for the virtual environment and one for the real cube. This was necessary to eliminate the time required to don the haptic devices during a comparison, which could affect memory of weight. To compensate for any effect of weight sensation difference between hands, one-half of the subjects used their dominant hand for the virtual environment while the other half used their non-dominant hand. The user compared the two cubes for all gravity values, which were randomized for each experiment. Decision time and response were recorded for analysis. The instructions for this experiment are provided in Appendix A.2.

5.2.2 Results and Findings

Statistical analysis was performed on the data collected in the experiments described in the preceding section (refer to Appendix C.1 for a description of all statistical methods utilized during the analysis of all experiments in this dissertation). In all experiments, the dependent variables recorded were participant response (less than, greater than, or equal to) and response time, or time required to make decision. These variables were analyzed for their dependence on the following factors:

- Cube presentation order – Test cubes presented earlier vs. later in sequence.
- Magnitude of the weight difference between the base and test cubes.
- Environment – Real vs. Virtual (comparing Experiments 1 and 2 only).
- Positive vs. Negative weight deviations of test cube compared to base cube.
- Dominant vs. Non-dominant hand for lifting test cube (Experiment 3 only).

The analyses conducted test the significance of each factor on the response time and correct decision percentage. All statistical calculations were performed in the Minitab analysis software.

A total of 14 subjects completed all three experiments. All participants were graduate level engineering students, and only a few had any experience with virtual reality or haptic technology.

Analysis of Experiments 1 and 2

The combined data for all users is shown in Figure 5.3 and Figure 5.4 and listed in Appendix B.1. In Figure 5.3, we can see that there are consistently a higher number of correct responses when the user was asked to judge between two real cubes rather than two virtual cubes. In addition, there seems to be a significant increasing trend of correct

responses for the real cubes as the mass of the test cube moves away from the 100-gram mass of the base cube. In fact, the data suggests that one has a very high probability of successfully detecting a 20-gram difference between two cubes near a mass of 100 grams. This trend is less noticeable, if at all, for the comparisons between two virtual cubes.

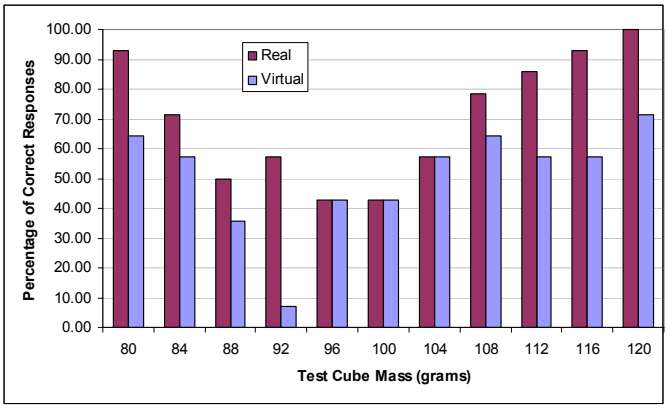


Figure 5.3: Percentage of Correct Responses versus Test Cube Mass

In Figure 5.4, the average response time is plotted against the test cube mass. The time required to compare the real cubes was less than that for the virtual cubes in all cases. In both experiments, there is a decrease in the amount of time required to make a decision as the mass of the test cube approaches the limits of the test range. This trend is slightly more pronounced for the comparison of real cubes.

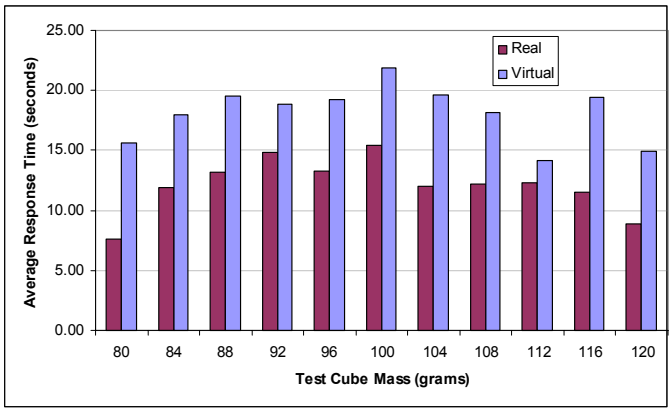


Figure 5.4: Response Time versus Test Cube Mass

Statistical analysis was performed to determine the significance of cube order and weight variations on the response variables. Following convention, statistical significance was declared for p-values less than 0.05, indicating a 95% confidence that the given independent variable affects the response variable. Table 5.1 shows the results of logistic regression for response correctness, a binary variable. As evidenced by high p-values, cube order had no effect in either environment, real or virtual, on the ability of the participants to distinguish between the two cubes. This suggests that learning was not a factor affecting performance during either experiment.

For the comparison of two real cubes, the magnitude of the weight difference between the cubes significantly affected the user's ability to make the correct judgment (p-value < 0.0005). In fact, user's identified the heavier of two real cubes with a success rate of 96% at a mass difference of 20 grams and only 50% of the time given a 4-gram mass difference. However, this relation was not statistically significant in the virtual environment. This implies that the ability of participants to sense a 20-gram mass

difference is no more significant than their ability to distinguish a 4-gram difference with a base mass of 100 grams.

Table 5.1: Logistic Regression with Correct Response in Determining Weight Difference as the Dependent Variable

Factor	Cube Set	Z-statistic	p-value	Odds ratio
abs(dWeight*)	Real	4.30	< 0.0005	1.15
	Virtual	1.93	0.054	1.05
Cube order	Real	0.39	0.697	1.02
	Virtual	-0.61	0.541	0.97

* 'dWeight' = 'Test Cube Weight' - 'Base Cube Weight'

In Table 5.2, we show the results of analysis of variance for the same two factors with the natural log of response time as the dependent variable. The natural log of response time was used for analysis because it provided a more accurate fit of the data, generating more meaningful results. Again, the only significant factor affecting response time is the magnitude of the weight difference in the real world. Participants required a significantly shorter decision time as the weight difference between the cubes increased. Response time was not affected by cube order in either environment.

Table 5.2: ANOVA with ln(Response Time) in Determining Weight Difference as the Dependent Variable

Factor	Cube Set	Degrees of Freedom	F-statistic	p-value
abs(dWeight*)	Real	5	3.14	0.010
	Virtual	5	1.41	0.225
Cube order	Real	10	1.28	0.246
	Virtual	10	0.95	0.491

* 'dWeight' = 'Test Cube Weight' - 'Base Cube Weight'

The total percentage of correct responses and average response time for all comparisons in the first two experiments are shown in Figure 5.5 and Figure 5.6, respectively. As shown, the users made a correct decision in approximately 70% of the comparisons for the real cubes and only 50% for the virtual cubes. At the same time, the response time increased substantially for comparisons in the virtual environment. Also shown in the figures is the standard error, indicating an estimated range for the true mean of the corresponding response variable.

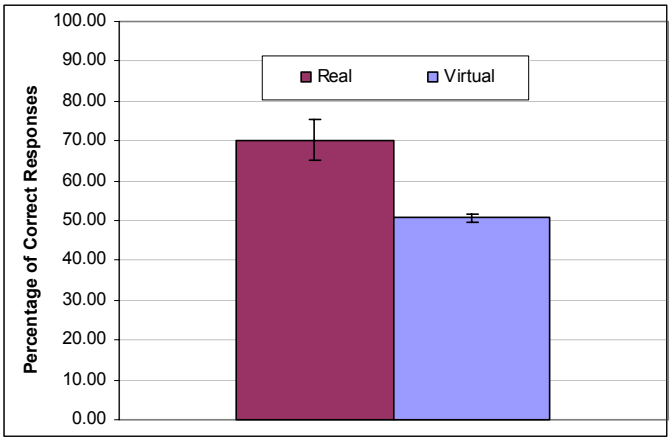


Figure 5.5: Total Percentage of Correct Responses (with standard error) in Weight Sensation Study, Summarized by Environment

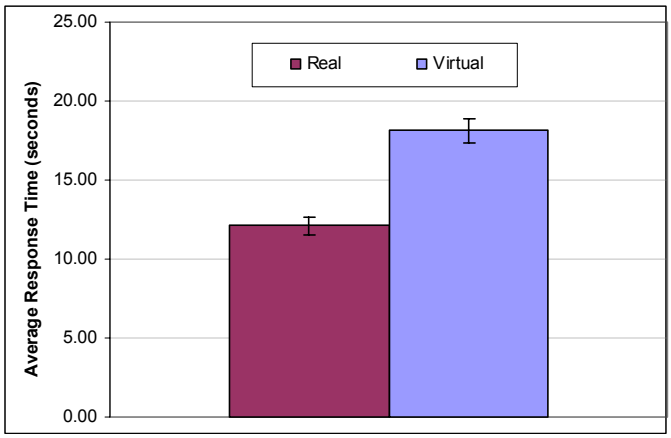


Figure 5.6: Average Response Times (with standard error) in Weight Sensation Study, Summarized by Environment

Analysis was performed on the data depicted in Figure 5.5 and Figure 5.6 to determine the statistical significance of the discrepancies seen. To compare the percentage of correct responses between real and virtual, a hypothesis test was conducted using the z-statistic to determine whether the two populations were the same (Table 5.3). Although the z-statistic is traditionally reserved for variables with a normal distribution, the relatively large number of observations permits its use for a binary variable (Hayter 1996). The p-value for this test shows that there is a statistical difference between the two environments. The estimated rate of correct responses using real cubes is 19% greater than that for virtual cubes.

Table 5.3: Comparison of Correct Responses in Judging Weight Difference between Real and Virtual Environments

Environment	Number of Observations	Success Rate	p-value	Estimated Difference (Real - Virtual)
Real	154	0.70	< 0.0005	0.19
Virtual	154	0.51		

In Table 5.4, we present the results of a 2-sample t-test for comparing the response time between the real and virtual environments. Based on the p-value shown, there is statistical significance for the dependency of response time on environment. It is estimated that a virtual comparison will take an average of six seconds longer than a comparison between real cubes.

Table 5.4: Comparison of Response Time in Judging Weight Difference between Real and Virtual Environments

Environment	Number of Observations	Mean	St. Dev.	p-value	Estimated Difference (Real - Virtual)
Real	154	12.10	7.38	< 0.0005	-6.02
Virtual	154	18.12	9.51		

When studying the data in Figure 5.3, we noticed that the experimental subjects tended to make more correct judgments for test cubes that were heavier than the base cube rather than lighter. We analyzed the data to test whether this trend was statistically significant. The results of this analysis are shown in Table 5.5. In fact, users were 20% more likely to make a correct judgment on the weight difference for test cubes heavier than the base, real and virtual, and the p-values suggest that this effect is genuine.

Table 5.5: Comparing Test Cubes Heavier and Lighter than the Base Cube

Cube Set	sign(dWeight*)	Number of Observations	Success Rate	p-value	Estimated Difference (Negative - Positive)
Real	Negative	70	0.63	0.008	-0.2
	Positive	70	0.83		
Virtual	Negative	70	0.41	0.016	-0.2
	Positive	70	0.61		

* 'dWeight' = 'Test Cube Weight' - 'Base Cube Weight'

Analysis of Experiment 3

The data for Experiment 3 is shown in Figure 5.7 and Figure 5.8. There were expected to be a higher percentage of correct responses as the virtual gravity constant moved away from 9.8 m/s^2. In Figure 5.7, we notice a trend that occurred in the previous experiments. There are higher rates of correct judgments for gravity constants larger than 9.8 m/s^2, which equate to heavier test cubes. The data shown in Figure 5.8 does not indicate a strong dependence of response time on virtual gravity constant. The average response time across all trials was approximately 19 seconds, which is slightly greater than Experiment 2, comparing two virtual blocks.

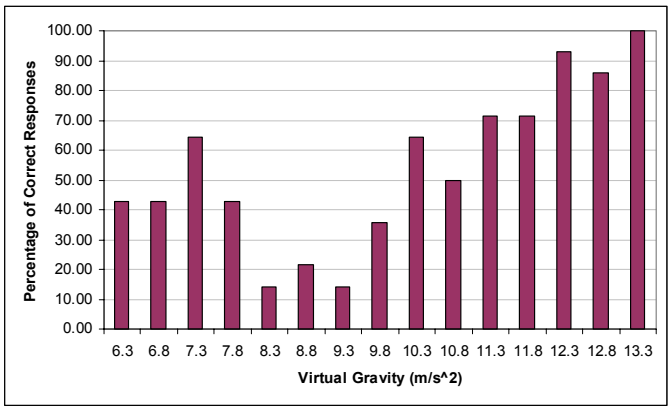


Figure 5.7: Percentage of Correct Responses versus Virtual Gravity Constant

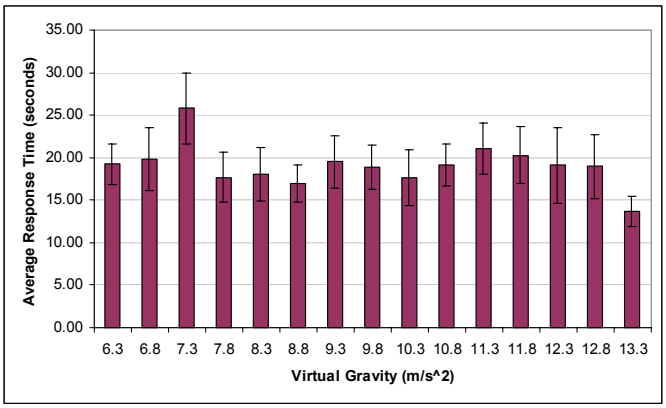


Figure 5.8: Response Time versus Virtual Gravity Constant

As with the previous experiments, statistical analysis was performed to determine any significance between the response variables and the various controlled factors. The order of gravity values presented to the users had no effect on the response variables. The magnitude of the difference between the real and virtual gravity constants was a significant factor influencing response correctness but not response time. This may be misleading, however, as most of the correct responses were for trials with large virtual gravity constants. In fact, analysis demonstrates that a user's ability to correctly judge weight differences depends greatly on whether the virtual gravity constant is larger or smaller than 9.8 m/s^2 (Table 5.6).

Table 5.6: Comparing Positive and Negative Deviations of Virtual Gravity

sign(dGrav*)	Number of Observations	Success Rate	p-value	Estimated Difference (Negative - Positive)
Negative	98	0.35	< 0.0005	-0.4
Positive	98	0.77		

* 'dGrav' = 'Virtual Gravity Constant' - 9.8 m/s^2

As mentioned, this experiment involved the use of both hands for weight comparison. During the study, half of the participants used their dominant hand for lifting

of the virtual cube, while the other half used their non-dominant hand. Analysis of this factor showed no significant difference in performance between the two groups, as evidenced by the high p-value in Table 5.7.

Table 5.7: Comparing Dominant versus Non-dominant Hand for Rate of Correct Responses

Hand for Virtual Environment	Number of Observations	Success Rate	p-value	Estimated Difference (Negative - Positive)
Dominant	105	0.53	0.782	0.0
Non-dominant	105	0.55		

* 'dGrav' = 'Virtual Gravity Constant' - 9.8 m/s^2

5.2.3 Revisiting Experiment 2 – Comparison of Two Virtual Cubes

When analyzing the data for Experiments 1 and 2, the trends in response time and number of correct responses were much more prevalent for the real cube set when compared to the virtual cubes. In fact, it is clear that there is a high probability that one can detect a 20-gram mass difference between two real cubes near 100 grams. However, for two cubes in a virtual environment, this threshold has not been determined. In this section, Experiment 2 will be revisited, and the threshold for detecting a mass difference between two cubes in a virtual environment will be explored.

The setup and procedures for this experiment are the same as for Experiment 2, with the exception that the mass of the test cubes are within one of two ranges: 55 to 75 grams and 125 to 145 grams, in 5-gram increments. The mass of the base cube remains at 100 grams, and the dimensions of all cubes remain the same. The data for this experiment was combined with the original data comparing two virtual cubes and is shown in Figure 5.9 and Figure 5.10. In Figure 5.9, the percentage of correct responses versus test cube

mass is displayed, with the vertical lines showing the division between the original Experiment 2 data and the extended results. Here, we notice a more distinct trend in the number of correct responses. The data suggests that the detectable mass difference between two cubes in a virtual environment is around 40 to 45 grams for a 100-gram base cube mass, about twice as much as that for the comparison of real cubes (Experiment 1). There are two unexpected data points in Figure 5.9, those corresponding to the virtual test cubes with a mass of 55 and 70 grams. The percentage of correct responses for these test cube masses does not follow the trend of the other measurements, as values are low. One possible explanation is that the ability to perceive weight becomes less accurate as the mass of the virtual cube approaches zero. One potential cause of this is that the grip force used to lift the cube using the haptic interfaces approaches or exceeds the gravitational force on the cube, making weight sensation more difficult. However, the exact reason for this discrepancy remains unknown.

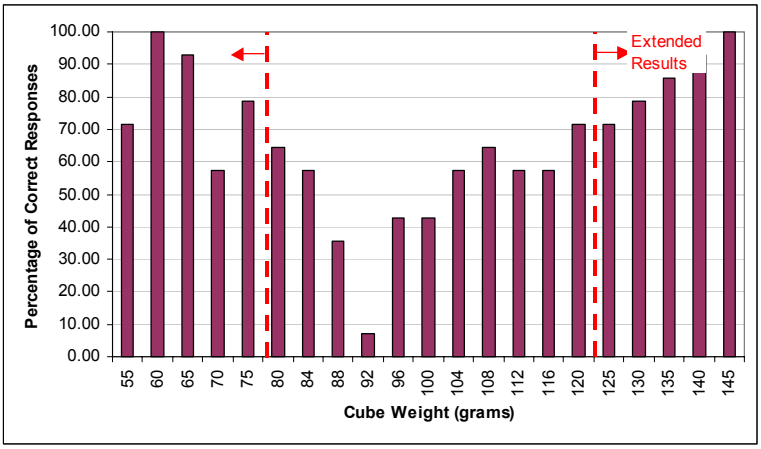


Figure 5.9: Percentage of Correct Responses versus Virtual Test Cube Mass (Experiment 2 Extended Results)

In Figure 5.10, the average response time for comparing two virtual cubes is plotted versus the test cube mass. Again, the extended results are shown with the original data collected for Experiment 2. Here, there is clearly a noticeable trend in response time. When the test cube mass is close to the mass of the base cube, 100 grams, the average response time of participants is near 20 seconds. As the difference in mass between the two cubes approaches 45-grams, the response time is nearly halved at close to 10 seconds. This is believed to represent a greater confidence by the participant in detecting the weight difference between the cubes as the mass difference increases.

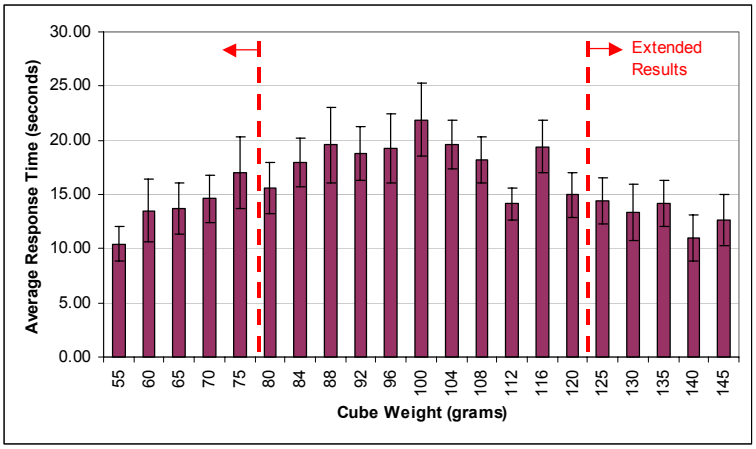


Figure 5.10: Response Time versus Virtual Test Cube Mass (Experiment 2 Extended Results)

As before, statistical tests were performed to determine the significance of the results. In Table 5.8, the results are displayed for logistic regression analysis of response correctness versus the magnitude of the mass difference between the two cubes and the order in which the test cubes were presented to the participant. As expected from the figures above, the magnitude of the mass difference between the base and test cube had a

significant effect on the user's ability to perceive a weight difference. In Table 5.9, results from analysis of variance are presented, with the natural log of response time as the dependent variable and the same independent variables. Unlike the previous results from Experiment 2, the response time was significantly affected by the magnitude of the mass difference with the extended results added. Cube presentation order had no effect on either response correctness or response time.

Table 5.8: Logistic Regression with Correct Response as Dependent Variable (Weight Sensation Experiment 2 Extended Results)

Factor	Z-statistic	p-value	Odds ratio
abs(dWeight*)	6.04	< 0.0005	1.07
Cube order	-0.96	0.335	0.96
* 'dWeight' = 'Test Cube Weight' - 'Base Cube Weight'			

Table 5.9: ANOVA with ln(Response Time) as the Dependent Variable (Weight Sensation Experiment 2 Extended Results)

Factor	Degrees of Freedom	F-statistic	p-value
abs(dWeight*)	10	3.98	< 0.005
Cube order	10	0.88	0.552
* 'dWeight' = 'Test Cube Weight' - 'Base Cube Weight'			

Lastly, the success rate in correctly identifying the mass difference between test cubes heavier and lighter than the base cube was analyzed (Table 5.10). As with previous experiments, participants were more likely to correctly identify a test cube that is heavier than the base cube, rather than a lighter one, by an estimated 13%. The apparent reason for this will be discussed in Section 5.2.6.

Table 5.10: Comparing Test Cubes Heavier and Lighter than the Base Cube (Weight Sensation Experiment 2 Extended Results)

sign(dWeight*)	Number of Observations	Success Rate	p-value	Estimated Difference (Negative - Positive)
Negative	140	0.61	0.021	-0.129
Positive	140	0.74		

* 'dWeight' = 'Test Cube Weight' - 'Base Cube Weight'

5.2.4 Validity of Experimental Procedure and Analysis

The weight sensation experiments were designed with three controlled factors: weight values for the test and base cubes, the comparison environment, and the hand used to pick up each cube (Experiment 3 only). In Experiments 1 and 2, the dominant hand was used for all comparisons. All other factors used for data analysis were either randomized, as with cube presentation order, or directly computed from the controlled factors. To control participant knowledge about the experiment, the only information given was in the instructions listed in Appendix A.

During the analysis of the experimental data, four statistical techniques were used: analysis of variance, 2-sample t-test, logistic regression, and a 2-proportion comparison test. The first two techniques are for continuous variables, while the last two are for binary variables. The experimental factors studied were deemed influential in the response time and correctness based on statistical significance using a critical p-value of 0.05, which is typical in practice.

When using the statistical techniques described above, the output was analyzed to assure that the model used was valid for the data. For instance, when using analysis of variance, it is assumed the response variable is normally distributed and the variance of the distribution is equal at all levels of the predictor variables. As is typically done,

residual analysis was used to determine whether the analysis was appropriate for the data. For instance, a plot of the residuals versus the fitted response values provided a method to check for constancy of the error variance. Also, a normal probability plot was analyzed to assure normality of the error terms. In fact, it was this residual analysis that resulted in the data for the response time to be analyzed using the natural log of the response time. Analysis of the statistical results in this way guaranteed validity of the model used to interpret the experimental data obtained. Refer to Appendix C.3 for a more detailed description of the process used to validate the appropriateness of the statistical models used for all experiments in this dissertation.

Cleary, the results presented in this study are truly valid only for the HIDRA simulation and, in particular, the dual-PHANToM experimental setup. For example, the experiments presented by Gurocak *et al.* using the AirGlove recorded a 60% correct response for identifying a 50-gram weight difference between two cubes in a virtual environment with haptic feedback (Gurocak *et al.* 2003). In the experiments discussed in the previous section, a 96% correct response rate was recorded for detecting a smaller weight difference of 20 grams between two virtual cubes. One possible factor leading to this performance difference between participants in the two experiments is the haptic device used. In our simulation, using the dual-PHANToM setup, forces are applied to the user's fingertips, whereas forces are applied to the user's hand with the AirGlove. The ability to perceive weight is highly dependent on the part of the body used to pick up an object. In this case, lifting with the fingertips provides higher fidelity for weight sensation than does lifting with the whole hand.

Another factor leading to the different results in these experiments is the average mass of the two cubes compared. In the experiments described above, the base mass remained constant at 100 grams, and all other test masses were relatively close to this. In experiments using the AirGlove, the average mass of the two cubes varied greatly depending on the weight difference to be tested. For instance, the 50-gram comparison was performed with cube masses of 650 and 700 grams, while the 100-gram comparison was completed using 250 and 350-gram cubes. The average mass of the two cubes being compared surely has an effect on the ability to distinguish a given weight difference. For example, a participant would be much more likely to correctly distinguish a 50-gram weight difference for cubes with masses around 100 grams rather than masses closer to 1 kilogram. It was for this reason that the experiments described in this section were designed using a constant base cube mass, arbitrarily chosen to be 100 grams, which can easily be lifted using the fingers. Following this train of thought, the results presented in this experiment are only valid for two cubes with masses around 100 grams. Additional experiments would need to be performed to determine the effect of cubes at higher or lower masses.

5.2.5 Feedback from Experiment Participants

After completing all three experiments, participants were asked to provide feedback. A summary of the observations is provided below.

- Most participants felt that the weight differences were very small in all trials, but the comparison was easiest for the real cubes (as evidenced in the data). In addition, users felt that judgments became easier as the trials progressed, though the data doesn't suggest this.

- Participants used various techniques to judge weight including: momentum, holding cubes steady, lifting slow or fast, grip force, slippage (in virtual environment), and different lifting motions (i.e., bending at wrist, elbow, shoulder, etc.).
- Many users found it difficult at first to judge depth in the virtual environment, making it difficult to change cubes quickly. This surely affected response time.
- A few participants thought that motion in the virtual environment was constrained, and others felt a small vibration in the haptic devices.
- The interface to the PHANToM device is a thimble, which was too small or too large for many of the users. Rubber bands were used to tighten the thimbles and create a snug fit when thimble was too large.
- Interestingly, one participant commented that it was easier to find cubes that were heavier than the base cube rather than lighter.

The items listed above highlight numerous uncontrolled factors that may have affected the judgments and decision times of the participants during the experiments. However, the experiment was designed to not control these factors. For example, participants were allowed to judge the weight of the cubes to the best of their ability regardless of technique used. Also, the depth perception difficulty experienced by users in the virtual environment undoubtedly increased decision time, but this is an undesired effect of working in a virtual environment. Stereoscopic vision could have been used but probably would not have helped, as evidenced by the results of all other experiments to be discussed.

Many people commented that the PHANToM thimble interface was either too loose or too tight, even though the thimble was designed with some flexibility. This highlights a limitation of the thimble interface between the user and the PHANToM. To compensate for this, rubber bands were placed around each thimble when necessary to provide a snug fit. For participants who noted that the thimble was too tight, nothing could be done, short of designing a new interface.

Some even observed that to make the comparison between the real and virtual cubes fair, the real cube could have been picked up with fingers inserted into the thimbles of the PHANToMs. In this fashion, users would not have the advantage of added sensitivity from the finger pads when lifting the real cube. Designing the experiment this way would eliminate this difference in sensory information between lifting a real cube versus a virtual cube using the PHANToM interfaces. However, this is a genuine and detrimental effect of using HIDRA, or any virtual environment with force feedback, that should be included in any experiment.

5.2.6 Discussion of Findings in Weight Sensation Experiments

This study was designed to compare the perception of weight in a virtual environment versus the same sensation in the real world and answer Research Question 2.1. The major findings resulting from Experiments 1 and 2, including the extended results are listed below.

- The ability to detect a mass difference between two cubes is better in the real world, when compared to the virtual environment. When asked to compare the weight of two cubes, participants not only responded correctly more often in the real environment, but also required less time.

- The threshold for detecting a mass difference between two cubes in the virtual environment is about twice that in the real world: ~40 grams versus ~20 grams.
- Some of the deviation in response time between the two environments can surely be attributed to the difficulty of depth perception in the virtual environment and the associated increase in time required to grasp the cubes.

It is believed that these results can be extended to objects other than two cubes, so long as the comparison includes two objects differing only in weight. As stated before, the comparison made is only truly valid for cubes with a mass near the base cube mass of 100 grams. It is believed that similar results would be found for experiments with a different base cube mass, although the detection thresholds would surely be different (i.e., larger threshold for a greater average cube mass).

In Experiment 3, we tried to determine whether weight sensation in the virtual environment matched that in the real world. The results for this experiment were inconclusive. At first glance, it appears that objects in the virtual environment are perceived to be heavier than the corresponding real object. However, all three experiments demonstrated a tendency for the users to judge the second weight as heavier. The users guessed that the test cube was heavier than the base in 47% of all trials, whereas judgments of lighter than and equal to were evenly split at about 26%. In fact, the origin of this discrepancy was an overlooked phenomenon known as the time-order error, in which, given two objects of equal weight, one typically perceives the second as being heavier (Hellstrom 1985). Although this had an effect in all of our experiments, our results comparing the real and virtual environments for detecting weight differences remain valid. That is, users can detect weight differential between two cubes in the real

environment at a higher success rate and in faster time than in the virtual environment. The noticeable effect on Experiment 3 is that the virtual gravity constant appears to be less than gravity in the physical world, or 9.8 m/s^2. This can probably be attributed to the time-order error.

Overall, weight sensation in the real world is more effective and easier than that in the virtual environment. In the next section, results from a second experiment characterizing the capabilities provided by HIDRA will be presented.

5.3 MOTION TOLERANCE EXPERIMENT IN A VIRTUAL ENVIRONMENT

The next experiment conducted during this research is a motion tolerance experiment. Similar to the weight sensation experiment, participants are asked to judge the range of motion of an object in two directions. This is the first experiment conducted to determine the usefulness of haptic feedback over a purely visual simulation environment. This experiment is designed to address the following research question:

> *RQ 2.2 – How does haptic feedback affect the ability of a user to detect motion tolerances in a virtual environment?*

and the corresponding hypothesis:

> *H 2.2 – In a virtual environment, haptic feedback improves a user's ability to detect differences in the range of motion of an object when compared to a purely visual simulation.*

To test this hypothesis, the motion tolerance experiment was performed. A detailed description of the experiment, including methods and procedures, analysis of results, and discussion of the findings, is presented in this section.

5.3.1 Methods and Procedures

This experiment was designed to test the ability of a person to distinguish the difference in allowable range of motion for a peg constraint to movement in two directions. A total of 8 human subjects participated in this experiment.

Setup: The experimental setup consists of a square peg resting in a square hole in the floor of the workspace. The peg measures 40 millimeters in width (x-axis), 40 millimeters in depth (z-axis), and 80 millimeters in height (y-axis). The depth of the hole in the floor is 40 millimeters. During this experiment, the motion of the peg is constrained to motion along the x and z axes only (i.e., the peg cannot be removed from the hole). A picture of the peg resting in the hole in the floor is shown in Figure 5.11, with the axes showing the allowable directions of motion. The participant viewed the peg as if sitting in front of the setup at a table, and the viewing angle remained fixed throughout the experiment.

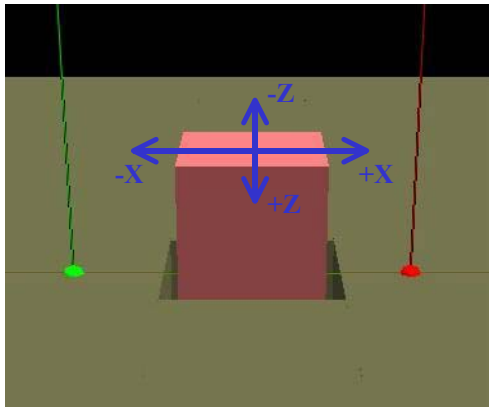


Figure 5.11: Setup for Motion Tolerance Experiment

There were ten predefined hole sizes that were tested during the experiment. The graphical representation shown in Figure 5.12 depicts five of these hole sizes that vary in

dimension along the x-axis. The colored square in this top-down view represents the peg, and the bordering squares represent the hole in the floor. For these five holes, the motion tolerance of the peg along the z-axis is ±5 mm. The size of each hole differs along the x-axis with motion tolerances of ±6 mm, ±7 mm, ±8 mm, ±9 mm, and ±10 mm. This same series of dimensions were created with elongation along the z-axis to create the other five holes.

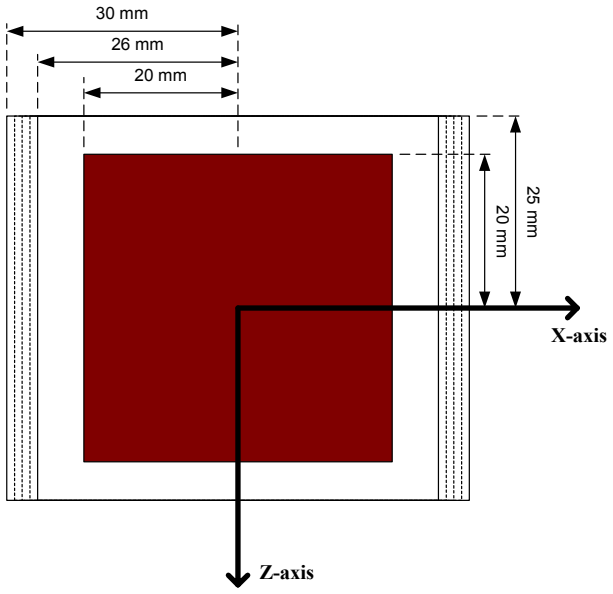


Figure 5.12: Varying Hole Size for the Motion Tolerance Experiment

Procedure: The user was presented with the setup as described above and instructed to move the peg from side-to-side (along the x-axis) and front-to-back (along the z-axis), comparing the allowable range of motion between the two directions. The participant was only allowed to grasp the peg on the two sides perpendicular to the x-axis. As soon as a decision was made, the user was asked to state which direction allowed the greater range of motion, and the response was recorded. Between judgments, the size

of the hole in the floor was randomly modified to one of the ten predefined dimensions. During this changeover, the graphics were disabled to prevent the user from making judgments based on seeing the hole size change. This same series of judgments, in different random orders, were completed using four versions of the simulation, created through combinations of haptic feedback versus no haptic feedback and 2D monitor viewing versus 3D viewing with the CrystalEyes shutter glasses. The typewritten instructions given to each participant is available in Appendix A.3.

5.3.2 Analysis of Results

Similar to the weight sensation study, two dependent variables were recorded during the experiment: participant judgment on the direction of greater allowable motion and response time. This data was analyzed for its statistical dependence on the following independent variables:

- 2D or 3D? – 2D monitor viewing or 3D viewing using the CrystalEyes shutter glasses.
- Haptics? – Whether or not haptic feedback was present.
- Magnitude of the tolerance difference between the two axes. During this analysis, tolerance difference is defined as the positive x-axis motion tolerance minus the positive z-axis motion tolerance. For instance if the motion tolerance values for a given hole are ±5 mm along the x-axis and ±6 mm along the z-axis, then the motion tolerance difference is equal to -1 mm. For the case when tolerance values are ±8 mm for the x-axis and ±5 mm for the z-axis, the tolerance difference is +3 mm.

- Sign of the motion tolerance difference (i.e., whether motion along the x-axis is greater or motion along the z-axis is greater).
- Simulation environment sequence – As mentioned, four versions of HIDRA were tested (2D/non-haptic, 2D/haptic, 3D/non-haptic, 3D/haptic). The ordering of these environments was randomized for each participant using a latin square design to prevent learning from affecting the analysis of the other factors.
- Tolerance presentation order – the order in which each hole size was presented to the user.

A tabular listing of the data collected during this experiment can be found in Appendix B.2. Graphical representations of the data collected for each dependent variable are shown in Figure 5.13 and Figure 5.14. In Figure 5.13, the percentage of correct responses versus motion tolerance difference is shown for each of the four simulation environments. The first thing to notice is that user's were 100% accurate in identifying the axis of greater allowable motion for tolerance differences equal or greater in magnitude than 4 millimeters in all environments. Overall, user's performed the best in the environment using haptic feedback and 3D visualization and the worst in the environment with no haptic feedback and 2D visualization. Further analysis later in this section will clarify the data shown in this figure.

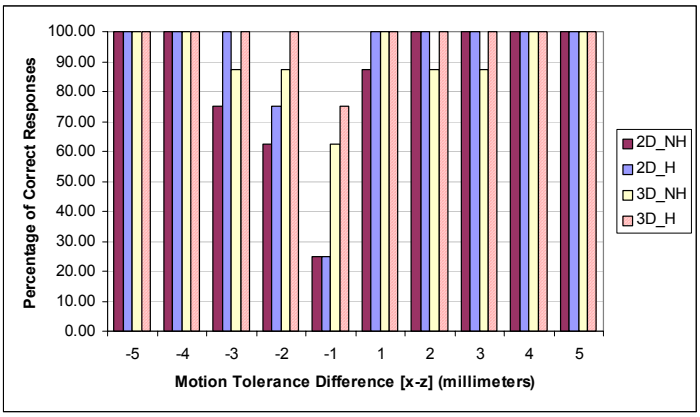


Figure 5.13: Percentage of Correct Responses versus Motion Tolerance Difference

The average response time versus motion tolerance difference is shown in Figure 5.14 for all simulation environments. The only immediately noticeable trend apparent in this figure is that decision time appears to be reduced as the magnitude of the motion tolerance increases, with the exception of a few data points.

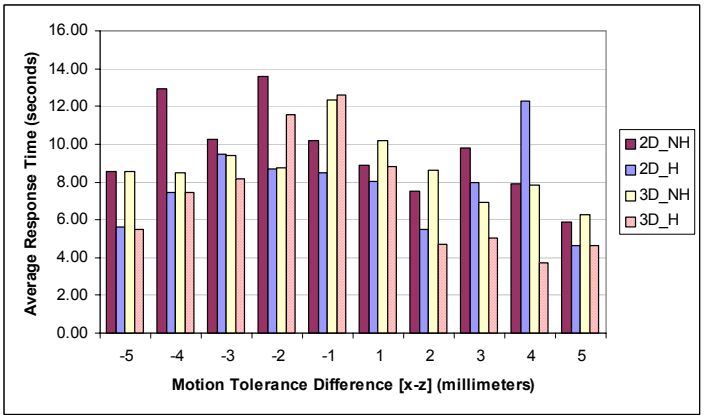


Figure 5.14: Response Time versus Motion Tolerance Difference

In order to gain a clearer picture of the effects of haptic feedback and type of visualization, the data was summarized further and is shown in Figure 5.15 and Figure

5.16. In these figures, it appears that both 3D visualization and haptic feedback improve the rate of correct responses and reduce the time required to make the decision, although the stereoscopic view seems to have a greater positive effect.

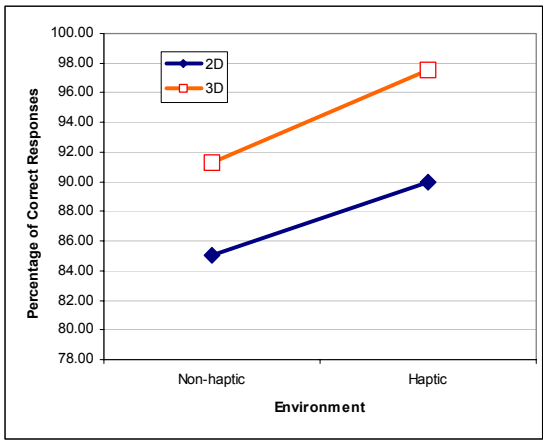


Figure 5.15: Percentage of Correct Responses for Motion Tolerance Study Summarized by Type of Visualization and Haptic Environment

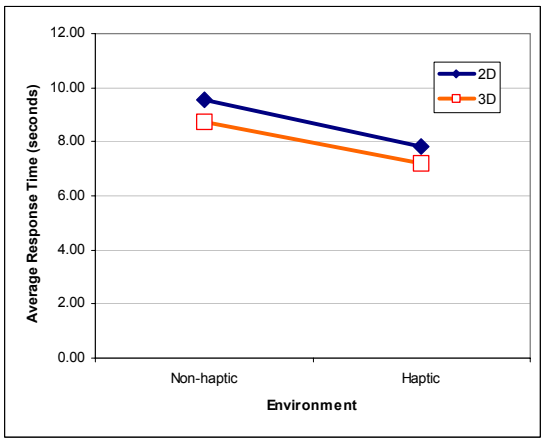


Figure 5.16: Response Time for Motion Tolerance Study Summarized by Type of Visualization and Haptic Environment

A two-way analysis of variance was performed on the data to determine the statistical significance of the type of visualization and the presence or absence of haptic feedback on the response time. The ANOVA was completed using the natural log of the response time since model validation analysis showed this to be a better fit. The results of this analysis are summarized in Table 5.11. As before, a factor is considered significant if its corresponding p-value is less than 0.05. In this experiment, the type of visualization, 2D or 3D, had no statistically significant effect on the response time. The presence of haptic feedback, on the other hand, did significantly reduce the time required to make a decision.

Table 5.11: Two-way ANOVA for Motion Tolerance Experiment – ln(Response Time) with Visualization and Haptic Feedback as Factors

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	1.23	0.268
Haptics?	1	18.99	< 0.0005
Interaction	1	0.02	0.901

In addition, a one-way analysis of variance was performed on the natural log of the response time with various other independent variables as factors. The results of these analyses are shown in Table 5.12. Although the results are summarized in a single table, the calculations for each factor were computed separately.

Table 5.12: Summary of One-way ANOVA Calculations for Motion Tolerance Experiment – ln(Response Time) versus Various Factors

Factor	Degrees of Freedom	F-statistic	p-value
Abs(Tol. Diff.)	4	5.29	< 0.0005
Sign(Tol. Diff.)	1	24.9	< 0.0005
Environment Sequence	3	6.44	< 0.0005
Presentation Order	9	1.7	0.089

The magnitude of the motion tolerance difference, shown as *Abs(Tol. Diff.)*, was a major factor affecting decision time. This is somewhat evident in Figure 5.14, as the response time seems to grow smaller with increasing magnitude of the motion tolerance difference. The sign of the motion tolerance difference, shown as *Sign(Tol. Diff.)*, also significantly affects the participants response time. This is evidence in Figure 5.17 as the response time is reduce by an average of around 2 seconds for trials where the allowable range of motion is greater along the x-axis rather than the z-axis. It is suspected that this decrease in response time for holes with a larger x-axis tolerance results from the angle at which the setup is viewed. Given the viewing position in front of the setup, the depth of the hole on the computer monitor appears smaller than its actual size. The width of the hole on the monitor, on the other hand, is displayed at its normal size. Therefore, participants were able to view changes in width more easily than changes in depth, and holes sizes with a large width were identified more quickly.

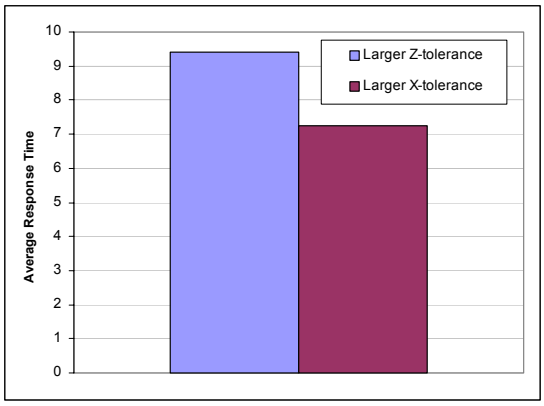


Figure 5.17: The Effect that the Direction of Greater Allowable Motion has on Response Time

As mentioned when describing the procedure for this experiment in Section 5.3.1, the ordering of the simulation environments tested for each subject was randomized using a latin square design. The low p-value in Table 5.12 for this factor indicates that it significantly affected the response time of the participants. As shown in Figure 5.18, the average response time required by participants during their first set of judgments, and thus first environment tested, was around 11 seconds. This was drastically reduced during the second set of judgments when using the second environment in the randomized sequence and continued to decrease slowly through the final environment in which they were tested. This demonstrates that the participants were gradually learning, or becoming faster at making judgments, as the trials progressed. The randomization of the environments prevents this learning from affecting the overall results and analysis of the other factors. The tolerance presentation order, or sequence of holes within each environment, did not have a significant effect on the response time.

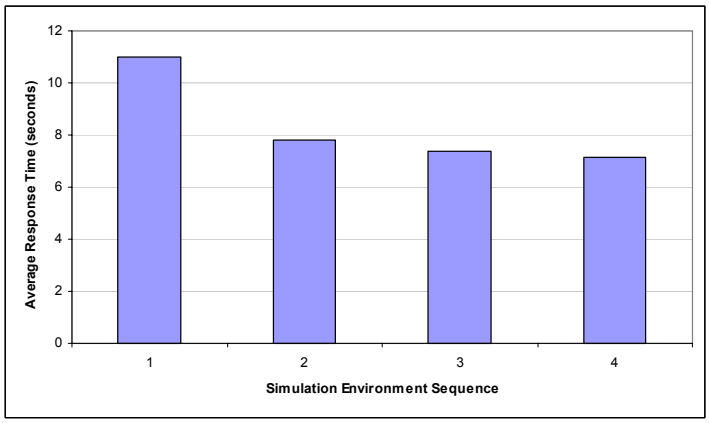


Figure 5.18: Effect of Simulation Environment Order on Response Time for Motion Tolerance Experiment

It has been shown that numerous factors affect the response time of the participant in the motion tolerance experiment. The remaining analysis will determine the effect theses factors had on the ability of the user to correctly judge the difference in allowable range of motion. With respect to analyzing correctness, the response has only two possible outcomes, correct or incorrect. Therefore, binary logistic regression will be used to analyze this variable. The result of a logistic regression analysis of the response correctness versus the type of visualization and the presence or absence of haptic feedback is shown in Table 5.13. As suggested by a low p-value, the type of visualization was a significant factor in the participant's ability to judge the range of motion effectively. Haptic feedback, on the other hand, did not have a significant effect in the user's ability to determine the direction of greater motion. Referring back to Figure 5.15, the presence of haptic feedback actually did increase the rate of correct responses, but statistical significance was not apparent. This lack of significance of haptic feedback in affecting response correctness was surely caused by the variability in the data. It is

expected that testing of additional subjects would reduce the variability and result in haptic feedback becoming statistically significant.

Table 5.13: Logistic regression of Correct Response versus Visualization Type and Haptic Feedback for Motion Tolerance Study

Factor	Z-statistic	p-value	Odds ratio
2D or 3D?	2.10	0.036	2.42
Haptics?	1.74	0.082	2.04

The last set of statistical tests performed on the data collected for this experiment was aimed at determining the effect of the various other factors on response correctness. The results of binary logistic regression with the response correctness as a dependent variable and the magnitude and sign of the motion tolerance difference as factors is shown in Table 5.14.

Table 5.14: Logistic regression of Correct Response versus Magnitude and Sign of the Tolerance Difference for Motion Tolerance Study

Factor	Z-statistic	p-value	Odds ratio
Abs(Tol. Diff.)	4.89	< 0.0005	3.26
Sign(Tol. Diff.)	-3.73	< 0.0005	0.10

Clearly, both factors significantly affect the ability to judge the difference in allowable motion correctly. Referring to Figure 5.13, increasing magnitude of the motion tolerance results in higher probability of response correctness. In Figure 5.19, the effect of the sign of the motion tolerance difference, which corresponds to holes with either a larger range of motion along the x-axis or along the z-axis, can be seen visually. Participants were approximately 14% more likely to respond correctly when the range of

motion along the x-axis was larger. The simulation environment sequence and tolerance or hole presentation ordering had no significant effect on the user's ability to correctly judge the direction of greater motion.

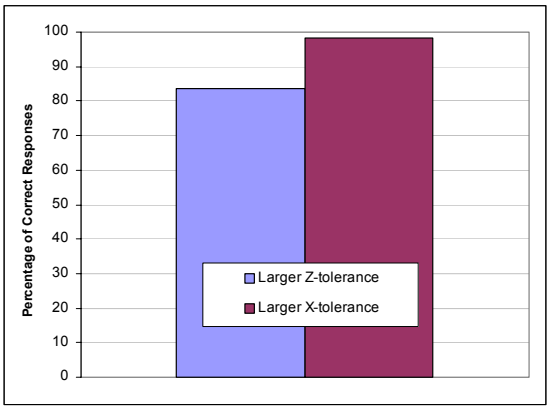


Figure 5.19: The Effect that the Direction of Greater Allowable Motion has on the Percentage of Correct Responses

5.3.3 Validity of Experimental Procedure and Analysis

As with the weight sensation experiment, slight modifications to the experimental design, such as changing the relative size of the peg and hole, may affect the amount of tolerance difference a user of the simulation could detect. However, it is expected that all other results with respect to the effect of the various factors on the dependent variables, response time and response correctness, would remain the same. There were only two controlled factors during the experiment: the presence or absence of haptic feedback and the type of visualization. The remaining factors, hole presentation order and environment sequencing, were randomized, the latter using a latin square design.

For this experiment, the statistical analysis techniques used were similar to those used for the weight tolerance experiment. The one additional technique implemented was

the two-way ANOVA, which allows the analysis of a continuous response variable versus two predictor variables and includes the testing of an interaction effect between the two. The validity of the ANOVA model is assured through residual analysis. Similar to the weight sensation experiment, the original ANOVA models used participant response time as the response variable. However, residual analysis showed that not only were the error terms from this analysis non-normal, but also did not have a constant variance. Various transformations of the response time were tested, and, for the data in this experiment, a natural log transformation of the response time produced the best fit of the data to the model, resulting in near constant error variance and a normal distribution of the error terms (see Appendix C.3).

5.3.4 Discussion of Findings in Motion Tolerance Experiments

The motion tolerance experiment was designed to determine the improvement that haptic feedback provides in detecting the differences in range of motion of an object when compared to a purely visual simulation (Research Question 2.2). The two factors selected to define improvement were the time required to make a judgment and the ability to correctly identify the direction of greater allowable motion. Below is a listing of the major finding resulting from this experiment.

- Haptic feedback reduced the time necessary to make a judgment on the direction of greater allowable motion. Although it appears through plots of the data that haptic feedback improved the user's ability to make the correct judgment, this was not statistically significant based on the threshold p-value of 0.05 (corresponding to a 95% confidence). However, the probable reason for this result is the large variability in the data collected, most of which is not explained by the

independent factors recorded. It is expected that additional trials of this experiment would reduce the overall variability of the data, resulting in the significance of haptic feedback for response correctness.

- Participants were much more capable in correctly identifying a greater range of motion along the x-axis than the z-axis. Additionally, judgments were made in faster time when the elongation of the hole was along the x-axis. This is believed to be a result of the difficulty in depth perception when working in a virtual environment. In the setup of the experiment, the x-axis corresponded to motion from side-to-side, while the z-axis corresponded to front-to-back motion. The width of the hole as seen on the screen directly correlated to the dimension along the x-axis. For the z-axis, however, this is different. Since the scene was being viewed from an angle in front of and above the surface of the virtual table, the depth of the hole on the monitor appeared to be much shorter than it actually was, causing user's to judge the motion along the z-axis in this way. The effects of this were more apparent when using the 2D monitor viewing versions of the simulation, but still had a very slight effect when using the 3D shutter glasses. Another factor that may have affected this phenomenon was the fact that the peg was grasped on the sides perpendicular to the x-axis.
- 3D visualization using the CrystalEyes shutter glasses was effective in improving the judgment of the difference in range of motion along the axes. This is attributed to improved depth perception.
- The last, somewhat unexpected result is that learning occurred from one set of trials to the next. Each group of trials, consisting of the same set of hole sizes in a

different random order, corresponded to a separate simulation environment (i.e., 2D/Haptic, 2D/Non-haptic, 3D/Haptic, and 3D/Non-haptic). Users improved by reducing the decision time from one environment to the next as the progressed through the trials. Users also improved their rate of correct response from one group of trials to the next, though not with statistical significance. As mentioned, the sequence of the virtual environments presented to the user was randomized using a latin square design to prevent the effects of learning from contaminating the analysis of the other factors.

- Based on the analysis of the experimental data, haptic feedback does appear to improve one's ability to detect differences in the range of motion of an object in a virtual environment. Overall, the best results, including reduced decision time and increased rate of correct responses, were attained when using the combination of haptic feedback and the 3D stereoscopic shutter glasses.

5.4 CHAPTER SUMMARY

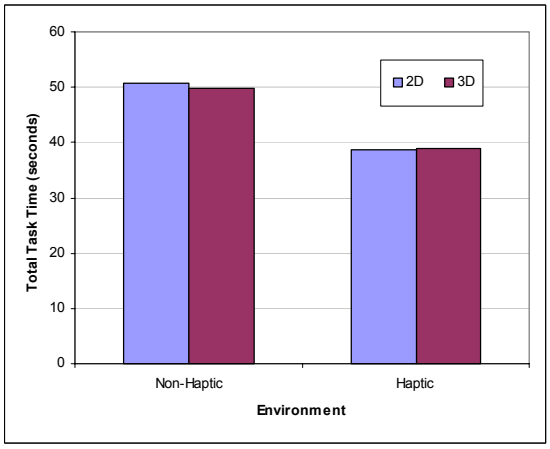
In this chapter, two experiments were discussed, a weight sensation study and a motion tolerance study. Each experiment was design to characterize the capability of haptic feedback in a virtual environment and determine its usefulness. Human subject volunteers conducted a series of tasks during each experiment, and pertinent data was recorded. Analysis of the data led to findings, some unexpected, that help to understand the capabilities and limitations of haptic feedback in a virtual environment. The weight sensation experiment revealed that the ability of a person to perceive weight in a virtual environment is less efficient than that in the real world. However, without haptic feedback, weight sensation would not even be possible in a virtual environment. Through

the second experiment, we found that haptic feedback can be helpful in determining the variations in the range of motion of an object in a virtual environment, although 3D visualization seems to more advantageous for this purpose.

The results of these experiments provide direct evidence for addressing Research Questions 2.1 and 2.2. In Chapter 7, a review of all the research questions and corresponding findings, including the acceptance or rejection of each hypothesis, will be provided. In the next chapter, the focus of experimentation will be directed toward addressing the usefulness of haptic feedback for assembly and disassembly simulation and Research Question 2.3. Three more experiments, conducted using the HIDRA simulation, will be discussed and a participant survey will be presented.

CHAPTER VI

EXPERIMENTS IN ASSEMBLY AND DISASSEMBLY USING HIDRA



In the previous chapter, two experiments were conducted in an effort to answer Research Questions 2.1 and 2.2. In this chapter, the details and results of three more experiments involving human subject volunteers will be discussed. All experiments described in this chapter will be directed toward answering Research Question 2.3, the last in this dissertation. In the Section 6.1, the goals for these assembly and disassembly experiments will be discussed. A relatively simple peg-in-hole experiment will be presented in Section 6.2. Next, a more complex task will involve the assembly of model train track pieces to form a continuous segment (Section 6.3). The final experiment will simulate the replacement of an old brake pad in an automotive brake assembly (Section

6.4). Finally, the results of a survey of the experimental participants will be provided in Section 6.5. In Section 6.6, a chapter summary will be provided.

6.1 GOALS FOR ASSEMBLY AND DISASSEMBLY EXPERIMENTS

The experiments presented in the previous chapter were designed to characterize the HIDRA simulation by exploring weight sensation and perceivable motion tolerances of virtual objects. These experiments were directly linked to testing the hypotheses of Research Questions 2.1 and 2.2. In this chapter, three more experimental studies will be discussed. The first is a peg-in-hole insertion experiment that will test the basic ability of a user in HIDRA to assemble and disassemble an object pair. The next experiment, assembly of a toy train track, will provide a more complex example of HIDRA's capabilities to model virtual objects and their interactions, while evaluating the importance of haptic feedback and other attributes of the simulation. Lastly, an application more relevant to the engineering domain will be studied. The final experiment will involve the replacement of an automotive brake pad, as modeled by the HIDRA simulation.

The three experiments discussed in this chapter were all designed with the intention of testing the final research question of this dissertation, Research Question 2.3:

> *RQ 2.3 – Does haptic feedback provide a significant improvement over a purely visual simulation for assembly and disassembly operations?*

and the corresponding hypothesis:

> *H 2.3 – Haptic feedback can provide significant improvements over a purely visual simulation for assembly and disassembly operations by reducing task completion time.*

In addition to answering this research question, many more interesting findings will result from the experiments. Also, the mere completion of these experiments will demonstrate improved performance of the simulation, with respect to object interactions, as a result of the research conducted to answer Research Question 1. Feedback collected from experimental participants will provide additional evidence to preferences and observations while using the HIDRA simulation.

6.2 PEG-IN-HOLE INSERTION EXPERIMENT

The first experiment conducted to assess the viability and usefulness of haptic feedback for assembly and disassembly analysis is a peg-in-hole study. Partial motivation for this experiment came from previous research conducted by Gupta (Gupta and Zeltzer 1995; Gupta *et al.* 1997). In this study, a very similar peg-in-hole experiment was carried out using a virtual environment with haptic feedback. VEDA, as it is known, allowed the interaction with virtual objects through a pair of PHANToM interfaces set up in the exact same configuration as HIDRA. The main limitation of this research, however, is that objects and their motions were only modeled in two dimensions. The results of the study showed that the introduction of haptic feedback in the virtual environment provided an improvement, through reduced assembly time, in the ability of the user to insert the peg into the hole. With HIDRA and the ability to model virtual objects in three dimensions, the experiment described in this section will confirm or deny the usefulness of haptic feedback for peg-in-hole insertion in a 3D environment.

6.2.1 Methods and Procedures

A total of 8 experimental subjects participated in this experiment each completing all aspects of this study. The setup and procedures followed during the execution of the peg-in-hole insertion task are described below.

Setup: The experimental setup of consists of two virtual objects, a square peg and the floor with a square hole cut out. The virtual peg measures 40 millimeters in length by 40 millimeters in width by 80 millimeters in height. The hole in the floor measures 44 millimeters in length by 44 millimeters in width by 40 millimeters in depth. Therefore, when the peg is inserted into the hole, there is a 2-millimeter tolerance around all sides and the center of the peg coincides with the surface of the floor in the vertical direction. An image of the initial experimental setup for the peg-in-hole insertion is shown in Figure 6.1.

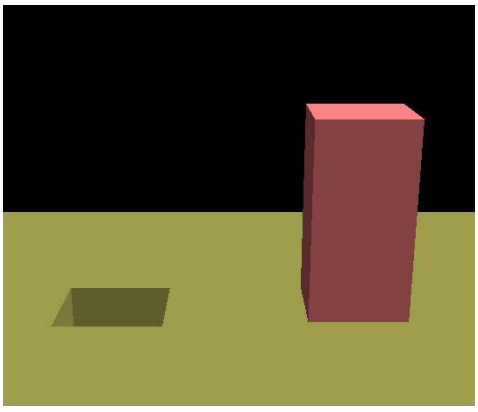


Figure 6.1: Setup for Peg-in-Hole Insertion Experiment

Procedure: During the task, the user was presented with the peg in the initial configuration shown in the figure above. When instructed, the user was asked to grab the peg from its resting position, lift it from the floor, and insert it into the hole until it was

fully seated. After the peg was fully inserted, the user then removed the peg from the hole and returned it to its initial resting position. This sequence of insertion and removal was completed five times in a row. Following a repeated measures experimental design, the entire procedure was completed by each participant in four variations of HIDRA, each a different combination of 2D monitor viewing versus 3D visualization with shutter glasses and haptic feedback versus only visual feedback. Refer to Appendix C.2 for a discussion on repeated measures design.

6.2.2 Analysis of Results

During this experiment, task completion time was recorded for analysis. The task completion time was further broken down into insertion and removal times for each of the five repetitions. The following independent variables were used for statistical analysis:

- 2D or 3D?
- Haptics?
- Simulation environment sequence – The ordering of different environment variations presented to the user.
- Repetition – As mentioned, the user completed the peg insertion and removal procedure five times in succession. This variable refers to this aspect of the experiment, wherein the entire procedure involves five repetitions.

The raw data collected during this experiment is available in Appendix B.3. A graphical depiction of the data is shown in Figure 6.2 and Figure 6.3. In Figure 6.2, the time required to insert the peg into the hole is plotted against repetitions during the task for all environment variations. The removal time versus repetition is shown in Figure 6.3.

The general trend in the data is an overall reduction in insertion and removal times as the user proceeds through the five repetitions. Also, it appears that the 3D, haptic environment generally produces the lowest recorded times.

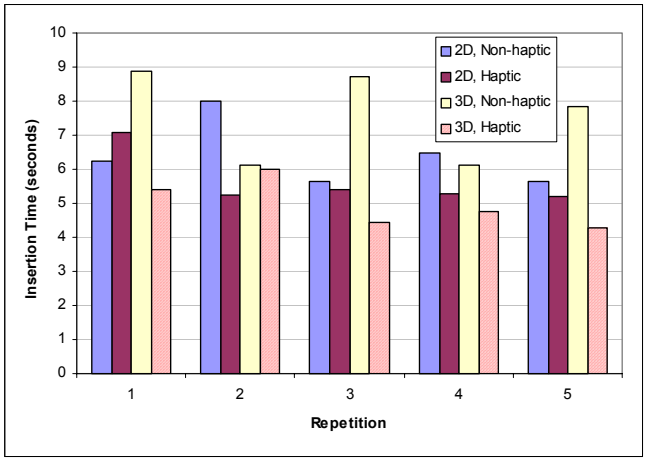


Figure 6.2: Insertion Time versus Repetition for Peg-in-Hole Experiment

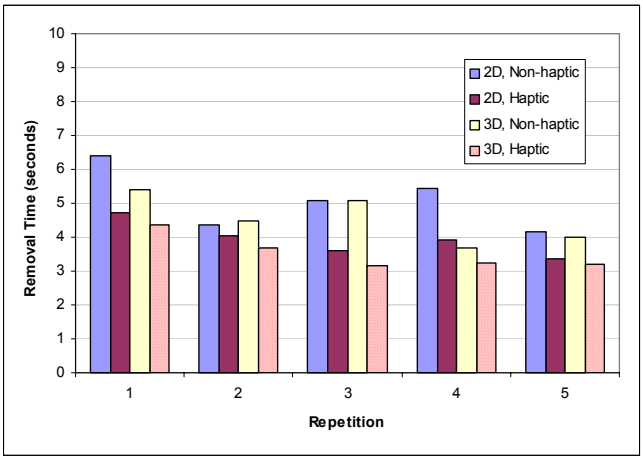


Figure 6.3: Removal Time versus Repetition for Peg-in-Hole Experiment

The data was analyzed to determine the statistical significance of the type of visualization, haptic feedback, the environment sequence, and repetition number as they

affect insertion time, removal time, and repetition time. Repetition time refers to the total time to complete insertion and removal for one repetition. As with the experiments in Chapter 5, analyzing the natural log of each of these times versus the controlled variables provided the best fit of the data to the ANOVA model. In Table 6.1, the result of a balanced ANOVA comparing the natural log of insertion time to the controlled variables is presented.

Table 6.1: Balanced ANOVA Comparing ln(Insertion Time) versus the Controlled Variables for Peg-in-Hole Experiment

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	0.09	0.759
Haptics?	1	11.19	0.001
Environment Sequence	3	0.72	0.543
Repetition	4	1.32	0.266

As can be seen, the only control variable that significantly affects insertion time is haptic feedback. In fact, the presence of haptic feedback reduced the insertion by an average of 1.7 seconds for all experiments. This reduction can be seen graphically in Figure 6.4, as the average insertion time for all repetitions is plotted against the availability of haptic feedback for 2D and 3D visualization. Interestingly, the insertion time with 2D viewing was less than that with 3D viewing for the non-haptic simulation environment. Just the opposite was true when haptic feedback was present. The reason for this is unknown.

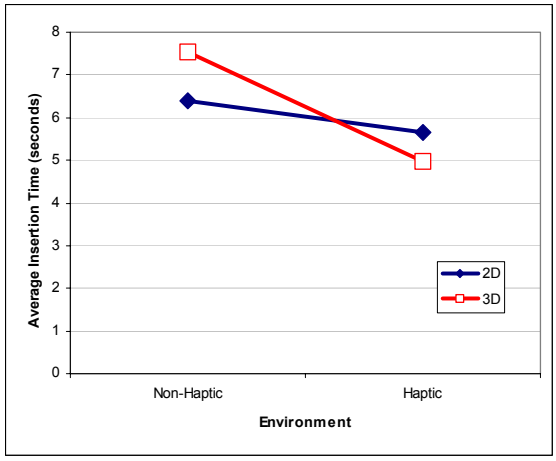


Figure 6.4: Insertion Time versus Haptic Environment for Peg-in-Hole Experiment

The data in Table 6.2 represents the results of a Balanced ANOVA for the natural log of removal time versus the same controlled variables. Again, haptic feedback remains significant for the removal of the peg from the hole. In addition, the environment sequence is a statistically significant factor. This suggests that learning occurred as the experiment progressed from one simulation environment to the next, and participants became more capable of removing the peg in shorter time. Similar results were found during the motion tolerance study. Visualization mode and repetition were not significant factors for the peg removal time.

Table 6.2: Balanced ANOVA Comparing ln(Removal Time) versus the Controlled Variables for Peg-in-Hole Experiment

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	3.86	0.051
Haptics?	1	10.42	0.002
Environment Sequence	3	4.12	0.008
Repetition	4	1.95	0.105

The dependence of removal time on availability of haptic feedback is shown in Figure 6.1. Notice that, unlike for the insertion time data, trials with 3D visualization require less time than their 2D counterparts for removal of the peg, though not statistically significant.

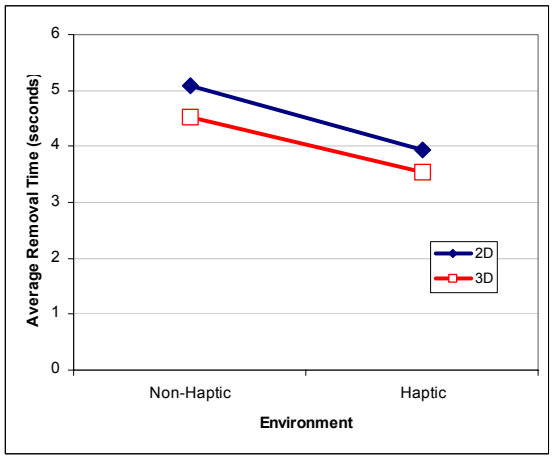


Figure 6.5: Removal Time versus Haptic Environment for Peg-in-Hole Experiment

In Figure 6.6, average peg removal time is plotted against environment sequence. Over the course of the four environments, the average removal time decreased from 5.1 seconds to 3.7 seconds. This signifies learning as the users progressed through the trials from one environment to the next. Similar results with respect to the environment sequence were not found with insertion time or total repetition time.

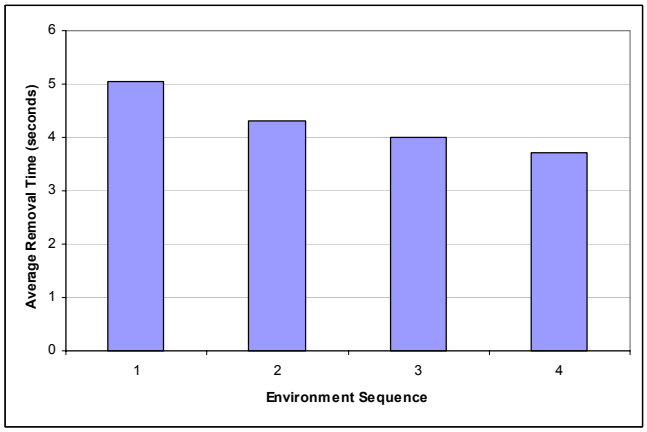


Figure 6.6: Removal Time versus Environment Sequence for Peg-in-Hole Experiment

The last statistical analysis performed for this experiment is a balanced ANOVA for the natural log of the total repetition time (Table 6.3).

Table 6.3: Balanced ANOVA Comparing ln(Repetition Time) versus the Controlled Variables for Peg-in-Hole Experiment

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	1.44	0.232
Haptics?	1	19.14	< 0.0005
Environment Sequence	3	1.87	0.137
Repetition	4	2.49	0.046

As with the other analyses haptic feedback had a significant effect on the total repetition time. This dependence on haptic feedback is highlighted in Figure 6.7. We also see the same interaction between the type of visualization, 2D or 3D, and availability of haptic feedback for the total repetition time as we saw for the insertion time. This is not surprising, however, since the two values are directly related.

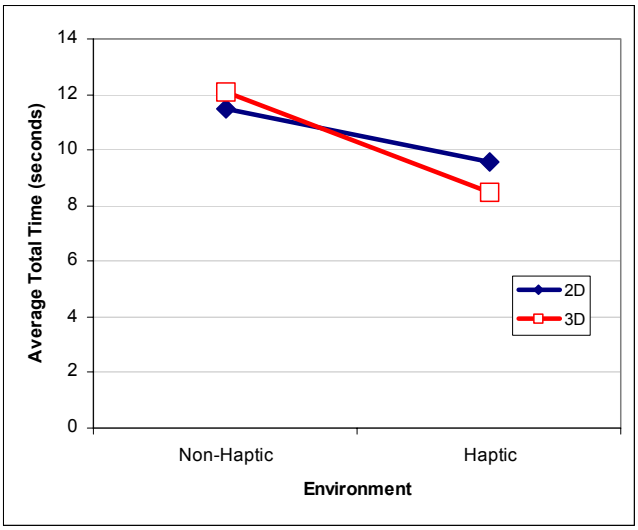


Figure 6.7: Repetition Time versus Haptic Environment for Peg-in-Hole Experiment

In addition to the availability of haptic feedback, insertion/removal repetition was also considered to be statistically significant. The dependence of total insertion and removal time on repetition is shown in Figure 6.8, as broken down into its components. Over the course of the five iterations of peg insertion and removal, the total time reduces on average from approximately 12.1 seconds to 9.4 seconds. The reduction is most likely explained as learning occurring during the task. Although both the insertion and removal time also reduce from one repetition to the next, these reductions were not found to be significant, probably due to the large variability of the data. Additional trials would most likely find that the reduction of insertion and removal times is also significantly significant with respect to repetition and user learning.

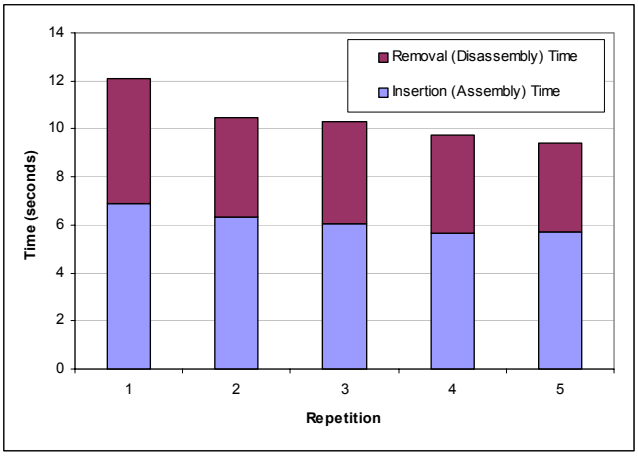


Figure 6.8: Insertion and Removal Time versus Repetition for Peg-in-Hole Experiment

6.2.3 Discussion and Validation of Results

In the motion tolerance experiment, several factors were being tested, type of visualization, availability of haptic feedback, environment sequence, and repetition, all with respect to their influence on task completion times. A discussion of the significant findings resulting from this study is provided below.

- First haptic feedback significantly influenced the time required by participants to insert the peg into the hole and remove it from the hole. In fact, of all the control variables tested, the presence of haptic feedback was the only factor to reduce peg insertion time.
- From one environment to the next, the time required by participants to remove the peg from the hole, a disassembly operation, was significantly reduced. However, this was not the case for insertion time. The reason for environment sequence influencing removal time and not insertion time is not yet understood.

- Lastly, the overall peg repetition time, to include insertion and removal, was significantly affected by the repetition count. Remember, within one environment the peg was inserted and removed five times in succession, or repetition. This gradual reduction in repetition time from the first repetition to the last corresponds to short-term learning in that, within a given sequence of repetitions, the user became more efficient and faster at the task.

Overall, the experiment was a success in answering the questions it was designed to answer. Many tasks during the assembly of a product involve peg-in-hole insertion of some form or another. For instance, inserting a bolt through a hole is a slightly more complex peg-in-hole task since the bolt is round. However, there is no reason to believe that the results of this experiment, in particular the reduction in task time with presence of haptic feedback, would not also be true for more complex insertion or removal procedures.

During processing of the data collected, the balanced ANOVA technique was utilized to analyze the significance of several factors on insertion, removal, and repetition time. After completion of the analysis, residual analysis was conducted to assure that the data was appropriately used with respect to the ANOVA model. It was this analysis that resulted in comparisons being made with the natural log of time, rather than time itself, as this provided a response variable with a normal distribution (see Appendix C.3).

6.3 TRAIN TRACK ASSEMBLY

Although the results of the peg-in-hole experiment provide a good set of results to assist in the response to Research Question 2.3, the experiment is a relatively simple procedure with respect to assembly and disassembly. The next study undertaken includes

multiple dynamic objects with much more interesting geometric features, which ultimately challenge HIDRA's capability to model object interactions effectively during real-time simulation. In this experiment, participants were asked to assemble four sections of a model train track. The setup, experimental procedure, and analysis and discussion of the results for the train track assembly experiment are discussed in this section.

6.3.1 Methods and Procedures

In this section, the setup and procedure for the train track assembly experiment will be discussed.

Setup: The objects for this experiment consisted of four sections of a model train track. This experiment included assembly scenarios in the real world and HIDRA's virtual environment. The virtual representation of the objects were created using measurements from a real set of model train track sections. Images of the real and virtual objects used during this experiment are shown in Figure 6.9 and Figure 6.10, respectively.

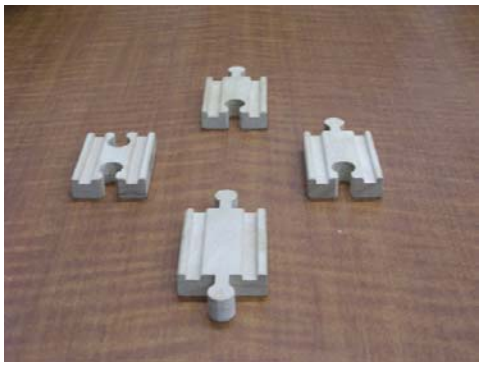


Figure 6.9: Initial Configuration for Train Track Assembly Experiment – Real

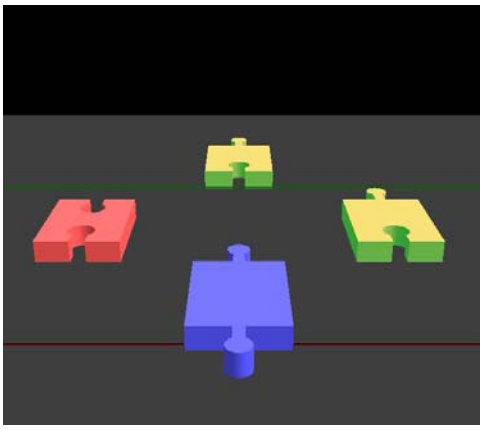


Figure 6.10: Initial Configuration for Train Track Assembly Experiment – Virtual

In all, there were three variations of the train track segments, referred to as male-male, female-female, and male-female. The dimensions for the male-female track section is illustrated in Figure 6.11. The main body of the object measures 40 millimeters by 52 millimeters with a thickness of 12 millimeters. As can be seen the male segment of the track measures 2.5 millimeters smaller than the female segment in all dimensions. Given the 0.25-millimeter collision threshold, as defined in Section 4.1.1, this corresponds to a motion tolerance of 1 millimeter along all sides of the piece. All other sections, male-male and female-female, were modeled using the appropriate dimensions in this figure. These dimensions were measured from the real train track pieces and used to create the virtual representations. For simplicity, the grooves in the train track sections were not modeled.

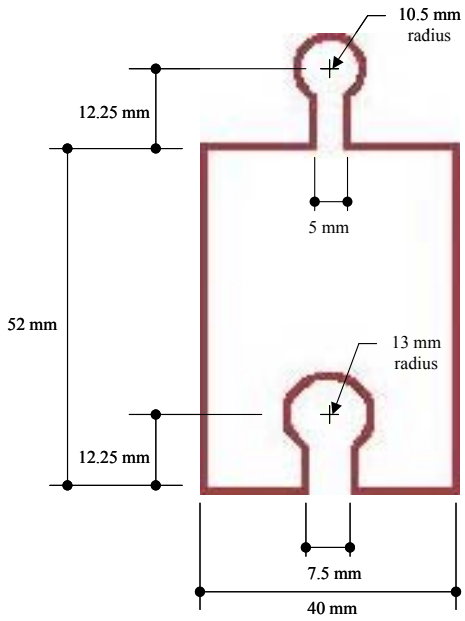


Figure 6.11: Dimensions for Train Track Section

Procedure: For this experiment, the user was presented with the train track pieces positioned in the initial configuration shown in the figures above. The goal was to assemble each segment of the track to form a continuous section. During completion of this experiment, the male-female track section furthest from the user was anchored, preventing movement of this object. When instructed, the participant began the assembly process, starting with the male-male track section closest to the user, followed by the female-female segment to the left, and finishing with the male-female segment to the right. The task was considered complete when all sections of the track were assembled to form a straight line. For reference, the image in Figure 6.12 is provided to show the final configuration of the train track for the virtual environment. The specific instructions given to each experimental participant is provided in Appendix A.5.

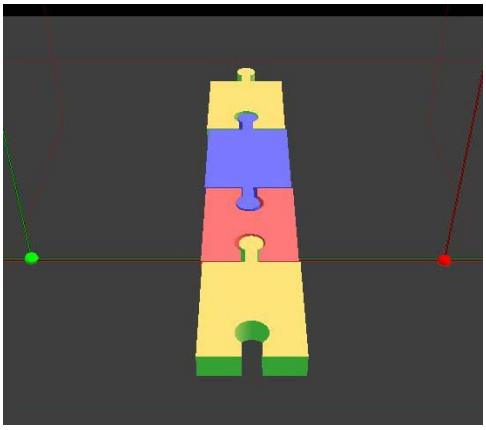


Figure 6.12: Final Configuration for Train Track Assembly Experiment – Virtual

In this experiment, and this experiment only, the participants were allowed to utilize HIDRA's simulation fly-through (see Section 2.3.1) to the extent that they could rotate the viewing angle up and down. This assisted in the assembly process by allowing a better line of sight since the train sections assembled by sliding together from the top. Without this feature, the difficulty in this assembly sequence would have been greatly magnified. The procedure described above was completed using the real train track pieces and their virtual representation, for which the assembly scenario was completed using all combinations of visualization and availability of haptic feedback (see Appendix C.2).

6.3.2 Analysis of Results

Similar to the peg-in-hole experiment, task completion time was recorded during the assembly of the train track. The task completion time is composed of two quantities: time required to grasp a given piece and time to assemble that piece (collected for all sections of the track). In the analysis, the grasping time and assembly time are combined

to derive a quantity referred to in this section as piece time. The controlled variables utilized for analysis are:

- Real World vs. Virtual Environment
- 2D or 3D?
- Haptics?
- Environment sequence
- Track Piece – The user was required to assemble three track pieces. This quantity refers to track piece being assembled.

In all, 15 subjects participated in this experiment, all of which were previously involved in the weight sensation experiment. The data collected during the train track assembly experiments is provided in Appendix B.4. The task completion time for each individual track segment versus environment is graphically depicted in Figure 6.13. From this figure, it comes as no surprise that assembly time in the real environment is far less than in any variation of the virtual environment. In addition, the total grasp and assembly time for the first track segment is always greater than the other two. The lowest piece time for all sections in the virtual environment occurs with 2D visualization and haptic feedback.

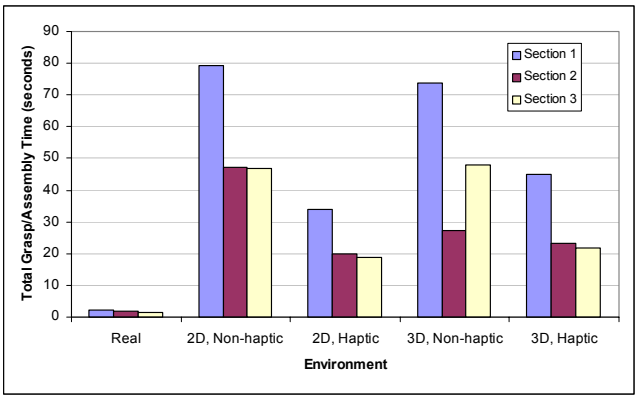


Figure 6.13: Total Grasp and Assembly Time for Each Track Section versus Environment

The first statistical analysis performed on the data was a comparison between the average task completion time in the real environment and that in all virtual environments. The result of a 2-sample t-test comparing the real and virtual environments is provided in Table 6.4. As suspected, the difference between task completion time between the two environments is statistically significant. In fact, users completed the task in an average of 6 seconds in the real world and 121 seconds in the virtual environment, a difference of 115 seconds. There are believed to be two main reasons for this extreme discrepancy. The first, as previously discussed, is the reduced ability to manipulate objects in a virtual environment. Another factor that probably increased task completion time was depth perception and the reduced ability to quickly change viewing angle. To clarify, users typically found it easier to grasp objects when viewed at a low viewing angle, or a line of sight near the same plane as the floor of the workspace. When assembling objects, however, it is easier to view the objects from above, at a large viewing angle. In the real environment, one can change their viewing angle almost immediately and without

thought. In the virtual environment, on the other hand, adjusting the viewing angle while manipulating an object is more difficult, causing an increase in task completion time.

Table 6.4: Comparing Total Task Completion Time between Real and Virtual Environments

Environment	Number of Observations	Mean	St. Dev.	p-value	Estimated Difference (Real - Virtual)
Real	15	5.93	1.53	< 0.0005	-115.297
Virtual	60	121.20	75.40		

The remaining analysis involves task completion times in the virtual environment. Data collection for the task was broken down into components, grasp time and assembly time, in an attempt to gain more insight into the influence of the control variables. In Table 6.5, the results of a balanced ANOVA are shown with the natural log of grasp time as the response variable. As with all other experiments, the natural log provided the best fit of the data to the ANOVA model, namely that the response variable follows a normal distribution.

Table 6.5: Balanced ANOVA Comparing ln(Grasp Time) versus the Controlled Variables for Train Track Assembly

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	0.13	0.724
Haptics?	1	2.44	0.120
Environment Sequence	4	2.66	0.034
Track Piece	2	11.52	< 0.0005

As seen in Table 6.5, neither the type of visualization nor the availability of haptic feedback significantly influenced the grasp time. The time required to grasp each object

was significantly influenced by environment sequence and the track piece. The graph in Figure 6.14 shows the dependence of grasp time on track piece, as the grasp time for the first track section is much lower than the other two. The most likely reason for this is that the user was concentrated on picking up the first track section at the beginning of the experiment, and nothing else. In addition, remember that the viewing angle was typically very high (top-down view) during the assembly of each track section. When completing the assembly of one track section, most users typically rotated their viewing angle to a much lower position (front view) to allow for easier grasping of the next piece. This extra time required to change viewing angle essentially increased the time required to grasp the next piece of the track. In the real environment, this phenomenon would be less likely to occur since changing one's viewing angle is more natural and efficient.

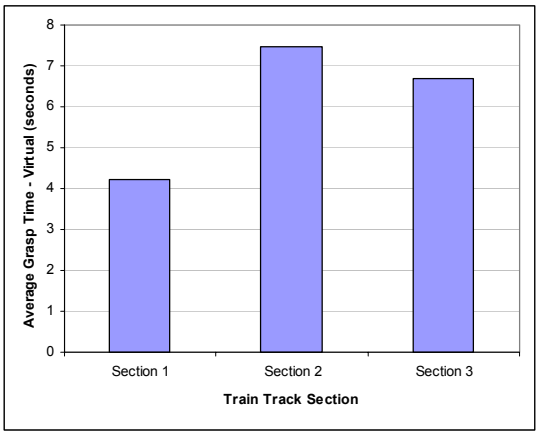


Figure 6.14: Average Grasp Time versus Train Track Section

In Figure 6.15, the dependence of grasp time on environment sequence is illustrated. For the first environment encountered by each user, the average grasp time for all track pieces is about 10 seconds. This time is cut in half by the second environment in

the sequence and levels off, ending with an average grasp time of about 4 seconds for the final environment faced. The same appears to be true for assembly time of each piece, as the time required for assembly drops from 45 seconds in the first environment and levels off at about 25 seconds from that point forth. This drastic reduction in grasp and assembly times most likely corresponds to each user becoming acquainted with the fit of each train track together and the ability to change the viewing angle.

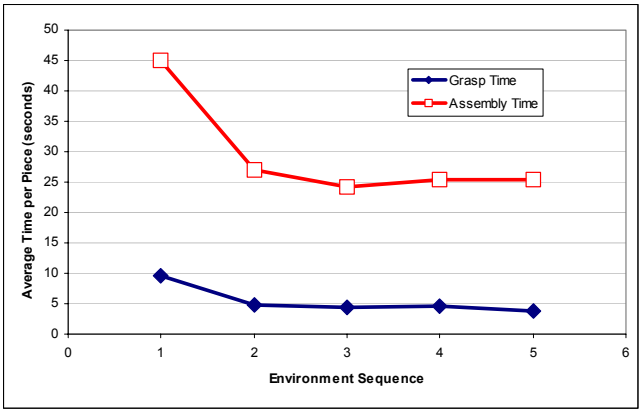


Figure 6.15: Average Track Piece Grasp and Assembly Times versus Environment Sequence

In Table 6.6, a summary of the results for a balanced ANOVA for the natural log of assembly time is provided. Again, the type of visualization was not a significant factor in predicting assembly time. Also, despite the large reduction in average assembly time from the first environment to the rest (Figure 6.15), environment sequence was not determined to have statistical significance in affecting assembly time. This is due to the large variability in the data collected.

Table 6.6: Balanced ANOVA Comparing ln(Assembly Time) versus the Controlled Variables for Train Track Assembly

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	0.02	0.888
Haptics?	1	48.54	< 0.0005
Environment Sequence	4	0.75	0.559
Track Piece	2	23.45	< 0.0005

Contrary to the analysis for grasp time, the availability of haptic feedback did have a significant effect. This dependence is illustrated through the graph in Figure 6.16. The average assembly time for all scenarios without haptic feedback is 47 seconds and with haptic feedback is 22 seconds. The presence of haptic feedback was successful in reducing the average assembly time by more than 50%, a drastic improvement. One cause of this difference is that during the non-haptic assembly scenarios, users were unable to feel the collisions between two objects. When haptic feedback is active and two objects collide, the user is able to feel that the motion of the object they are handling is inhibited. Without haptic feedback, however, users sometimes continued to push an object in a direction that was not possible due to a collision with another object. The improved sense of object collisions in the haptic environment surely had a great influence on the assembly time of the train track pieces.

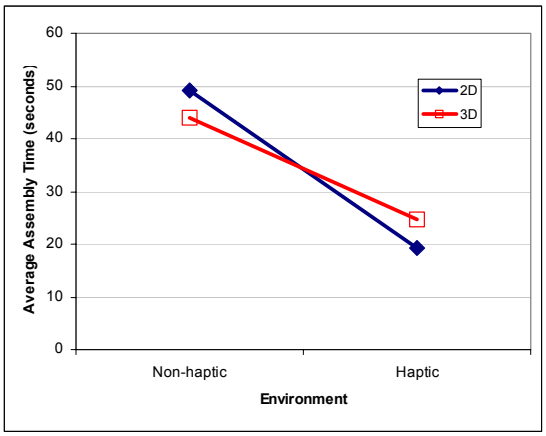


Figure 6.16: Average Track Piece Assembly Time Summarized by Type of Visualization and Haptic Environment

As with the grasp time, the average assembly time was significantly affected by the track section being assembled, but in a different way. As can be seen in Figure 6.17, the average assembly time for the first track segment was about twice as much as that for the second and third segments. It is possible that this reduction in time corresponds to learning during the assembly of the track sections, but this is not likely. A more probable explanation for this large discrepancy in the assembly time of each track piece is the position of the sections in the simulation environment during the assembly. Given the experimental setup, assembly of the first track piece occurs much deeper, or further from the user, in the virtual environment. As a consequence, viewing the assembly of the first segment was probably more difficult for two reasons. First, objects further from the user appear smaller, making the tight fit appear visually even tighter. Second, the depth of the assembly in the environment required that the user achieve an even greater viewing angle, when compared to the other sections, to effectively line up the first section for assembly. With a larger viewing angle (near top-down view) comes a difficulty in

determining the height of the object off the floor, which introduces additional complications for assembling the first train segment. It is believed that the greater depth of the first segment assembly is a major factor in the increased assembly time.

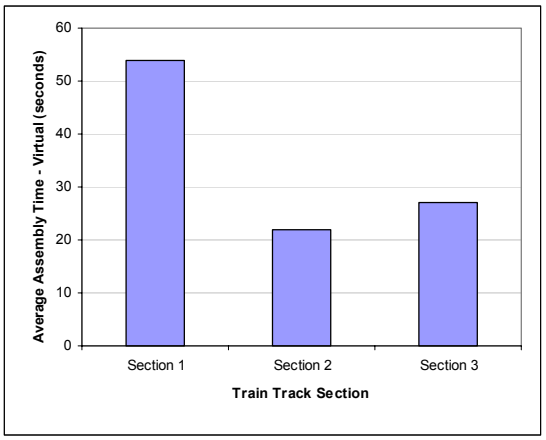


Figure 6.17: Average Assembly Time versus Train Track Section

Results from the last balanced ANOVA, comparing the natural log of piece time with the control factors, is shown in Table 6.7. Remember, piece time refers to the summation of grasp and assembly times. As with all other analyses, the total piece time was not significantly affected by the type of visualization. Also, the environment presentation sequence was determined to be statistically insignificant. Although both the grasp and assembly times seem to be greatly reduced from the first environment to the rest (refer to Figure 6.15), this factor was probably deemed insignificant due to the high variability of the data collected.

Table 6.7: Balanced ANOVA Comparing ln(Piece Time) versus the Controlled Variables for Train Track Assembly

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	< 0.005	0.999
Haptics?	1	48.49	< 0.0005
Environment Sequence	4	1.03	0.391
Track Piece	2	17.08	< 0.0005

Once again, the availability of haptic feedback was evaluated to be a statistically significant factor for piece time. The average piece time is plotted in Figure 6.18 versus the availability of haptic feedback for 2D and 3D visualization. The average time to grasp and assemble each track segment is cut in half when haptic feedback is activated, from approximately 160 seconds to 80 seconds.

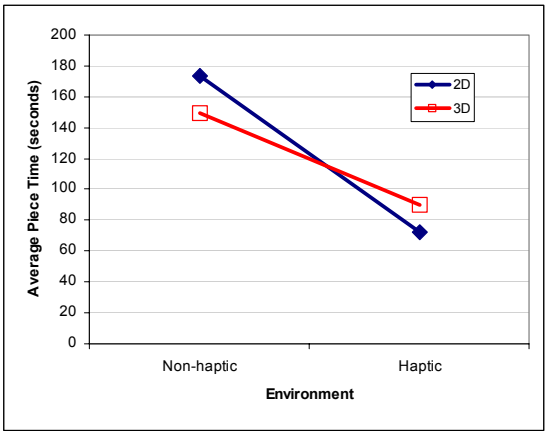


Figure 6.18: Average Track Piece Grasp and Assembly Time Summarized by Type of Visualization and Haptic Environment

The track segment being assembled is also an influential factor affecting average piece time. Similar to the data for assembly time (Figure 6.17), the total time required to

grasp and assemble the first track segment is far more than that for the other two segments. This is not surprising since the major component of overall piece time is assembly time. The same probable cause described earlier for assembly time applies here as well.

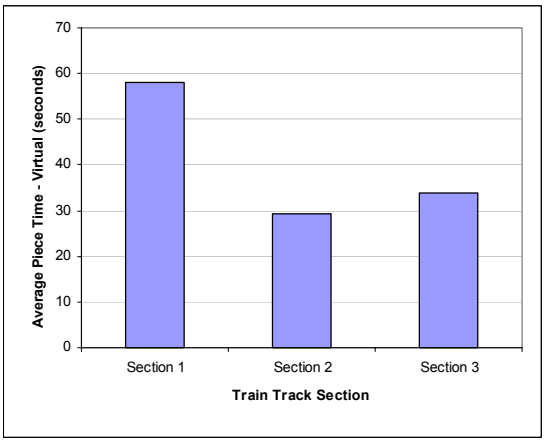


Figure 6.19: Average Grasp and Assembly Time versus Train Track Section

6.3.3 Discussion and Validation of Results

Although the train track experiment, as the others in this chapter, was designed with the explicit purpose of answering Research Question 2.3, many other interesting findings have resulted from an analysis of the data. Below is a discussion of the key findings during this study.

- First and foremost, haptic feedback has clearly been shown to reduce task completion time over a purely visual simulation. The assembly time and total piece time response variables were, on average, cut in half with the addition of haptic feedback to the simulation.

- Although only statistically significant for grasp time, the graphical representation of the data (Figure 6.15) suggests that the environment sequence, or order in which participants used each environment, affected all times. Furthermore, after a drastic reduction in grasp and assembly times from the first environment to the second, the task completion times leveled off. The average grasp and assembly times were probably much higher for the first environment as a result of the user becoming acquainted with the procedure and developing techniques to facilitate the assembly of the track pieces.
- The task times also statistically varied with which track piece was being assembled, although not in a way consistent with learning. Instead, the grasp time for the first section was much lower than for the other segments and the assembly and total piece times were much higher. The lower grasp time for the first piece is attributed to the fact that this corresponded to the beginning of the simulation and initial grasping was not inhibited by modification of the viewing angle or assuring assembly completion of the previous piece. For the assembly and total piece times, the larger task times for the first track segment are believed to be associated with the location of the assembly. Since the first section was assembled further from the user, visualization was more difficult, and more extreme viewing angles were required to get a good view of the assembly procedure.
- Not surprisingly, the assembly of the train track required more time in the virtual environment than in the real world. However, during the design of this experiment, it was not expected that task completion times in the virtual environment would average approximately 2 minutes. As with the weight

sensation experiment described in Chapter 5, this undeniably demonstrates that completion of tasks in a virtual environment, with or without haptic feedback, are less efficient and require more time than in the real world. Through time and research this gap will be reduced.

In hindsight, the train track assembly was the most difficult experiment for participants to complete. The increased difficulty associated with the need to change viewing angle for easier assembly probably contributed greatly to this.

The train track experiment was designed to determine the usefulness of haptic feedback for assembly simulation, and for that it was successful. During analysis of the data collected, a few interesting characteristics appeared in the data, including rather unique task completion times for the first track section when compared to the other two. As mentioned, this is believed to be associated with the depth of objects in the field of view. Also, a substantial reduction in track piece time from the first environment to the second was noticed in the data. Factors such as these and their believed causes will require further experiments to prove or disprove. Overall, this experiment successfully achieved its goal of evaluating the usefulness of haptic feedback for assembly.

For this experiment, two statistical analysis methods were utilized to test the significance of several factors on task completion time and its derivatives. A 2-sample t-test was implemented to compare data from the real world and the virtual environment. For all other analyses, a balance ANOVA with four factors was used. In order to insure that the ANOVA model was appropriate, residual analysis was conducted to confirm the normality and constant variance of the resulting error terms. Through this analysis, as

with other experiments, the natural log of task time was determined to provide the most appropriate fit to the ANOVA model (see Appendix C.3).

6.4 AUTOMOTIVE BRAKE PAD REPLACEMENT

The final experiment conducted in this research is the simulation of the replacement of an automotive brake pad. As with the previous two studies, the goal of this experiment is to provide evidence that will allow a response to Research Question 2.3. In support of the overall goal of this research, the procedure will involve both an assembly and disassembly component, while providing an example with relevance to engineering.

6.4.1 Methods and Procedures

In this section, the setup and procedure for the simulated brake pad replacement will be discussed.

Setup: The virtual environment for the brake pad replacement experiment consists of five virtual objects: the rotor, the caliper, the piston, the old brake pad, and the new brake pad. For ease of implementation and so that HIDRA can efficiently model the interactions between objects, a simplified model of the components was used. For the same reasons, not all of the components in a real automotive brake assembly are modeled, but enough to allow the testing desired. Also, only one-half of the brake assembly is modeled. An image of the components as they are initially presented to the user is shown in Figure 6.20. The individual parts are labeled by number as follows: 1 – old brake pad, 2 – new brake pad, 3 – caliper, 4 – piston, and 5 – rotor. As with the train track experiment, there is a 1-millimeter tolerance between each brake pad and the caliper in height. There is slightly more room for movement from side-to-side.

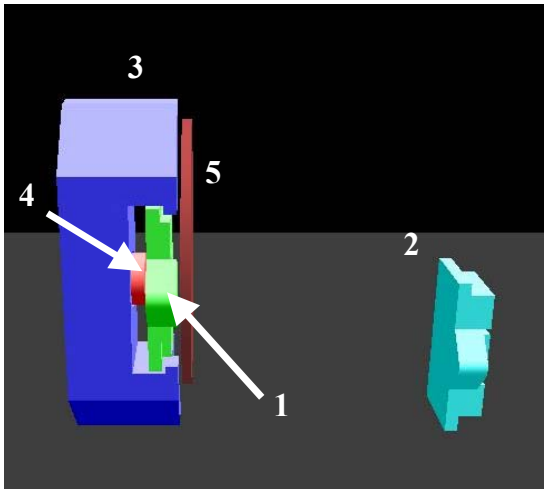


Figure 6.20: Initial Setup for Brake Pad Replacement Experiment

Procedure: During this task, the participant was presented with the components of the brake assembly as shown above (Figure 6.20). The procedure requires that the user replace the old brake pad, originally within the caliper, with the new brake pad sitting to the side. To do this, the old brake pad is first removed from the assembly by grasping the small tab along its leading edge and pulling straight toward the user. After disassembly is complete, the old brake pad is set to the side. The next step in the procedure involves pushing the piston back until it is flush with the caliper. Otherwise, the new brake pad, since it is thicker, will not fit into the slot formed by the caliper and the rotor. As a final step, the new brake pad is inserted into the caliper in the same way the old brake pad was removed. The final configuration of the components after replacement of the brake pad is shown in Figure 6.21. During the entire procedure, the caliper and the rotor were anchored to that they remain motionless. A total of 12 participants completed this procedure in all variations of HIDRA, defined by combinations of the type of visualization and availability of haptic feedback (see Appendix C.2). Each of the

participants had already completed the weight sensation experiment and the train track assembly experiment.

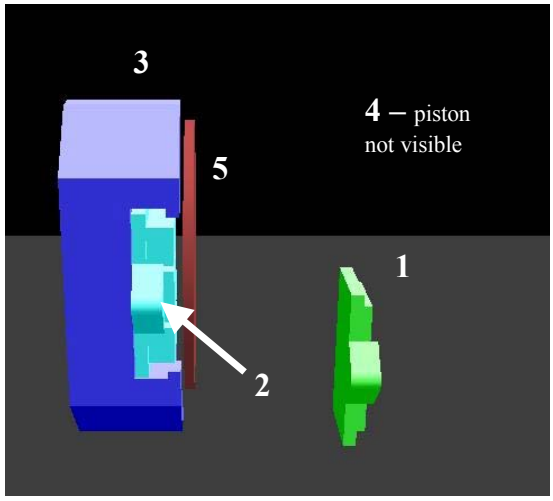


Figure 6.21: Final Configuration for Brake Pad Replacement Experiment

6.4.2 Analysis of Results

During the replacement procedure, six timing variables were recorded. These include time to grasp old brake pad, time to remove old brake pad, time to locate the piston for pushback, time to push the piston back, time to grasp the new brake pad, and time to assemble the new brake pad. A listing of the raw data collected during the brake pad replacement experiment is provided in Appendix B.7. These time values, along with the total replacement time, were analyzed for their dependence on the following controlled variables:

- 2D or 3D?
- Haptics?
- Environment sequence

A graphical representation of the average total time required to grasp/locate and assemble/disassemble/pushback each component during the brake pad replacement is shown in Figure 6.22 for all environment combinations. Also shown from this figure, we can see that the majority of the task completion time involves the assembly/disassembly of the two brake pads. Pushing back the piston required significantly less time.

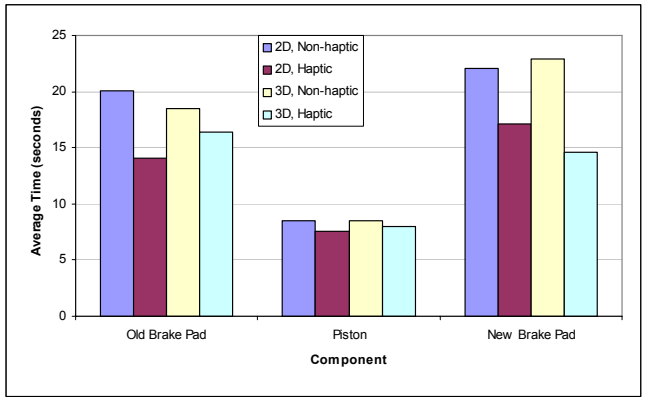


Figure 6.22: Total Object Manipulation Time including Grasp/Location Time for Components in Brake Pad Replacement Experiment

During the statistical analysis of the data, it was shown that neither the time required to locate the piston nor the time required to push it back were significantly affected by the control variables. For this reason, the analysis of the timing data will not be discussed further.

As with the other experiments in this chapter, analysis of variance was carried out on the natural log of each of the response variables, since this provided the most appropriate fit to the ANOVA model, namely that the response variables be normal. In Table 6.8, the result of a balanced ANOVA for the grasping time of the old brake pad is

presented. We can see from the p-values that neither the type of visualization nor the availability of haptic were significant factors affecting grasp time of the old brake pad.

Table 6.8: Balanced ANOVA Comparing ln(Old Brake Pad Grasp Time) versus the Controlled Variables for the Brake Pad Replacement Experiment

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	1.66	0.204
Haptics?	1	0.1	0.757
Environment Sequence	3	3.5	0.024

Environment sequence, on the other hand, was determined to be statistically significant for its influence on the grasping time. In Figure 6.23, the average grasp time for the old brake pad is plotted against environment sequence. The total reduction in average grasping time of this component was about 2 seconds, from 6.2 seconds in the first environment to 4.2 seconds in the last. This reduction in grasping time probably corresponds to the participants becoming more acquainted with, or essentially learning, the brake assembly setup in the virtual environment. Similar reductions occurred in some of the other timing results, including the total task completion time, but none were considered significant as will be shown below.

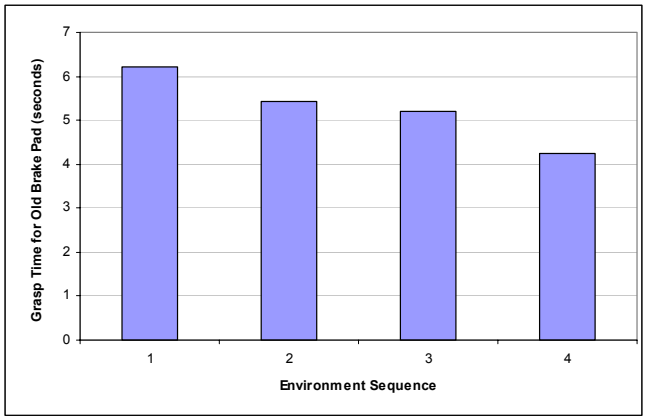


Figure 6.23: Average Grasp Time for the Old Brake Pad versus Environment Sequence

The result of a balanced ANOVA for the disassembly time of the old break pad versus the control variables is shown in Table 6.9. Again, the type of visualization was not a statistically significant factor in disassembly time of the old brake pad. Unlike the results for grasping time, disassembly time was also not affected by the environment sequence.

Table 6.9: Balanced ANOVA Comparing ln(Old Brake Pad Disassembly Time) versus the Controlled Variables for the Brake Pad Replacement Experiment

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	0.04	0.844
Haptics?	1	13.37	0.001
Environment Sequence	3	0.49	0.690

Haptic feedback was determined to be a significant factor affecting old brake pad disassembly time. In Figure 6.24, the average disassembly time for the old brake pad is

plotted against the availability of haptic feedback. Overall there is about a 4 second reduction in disassembly time for this component when haptic feedback is active.

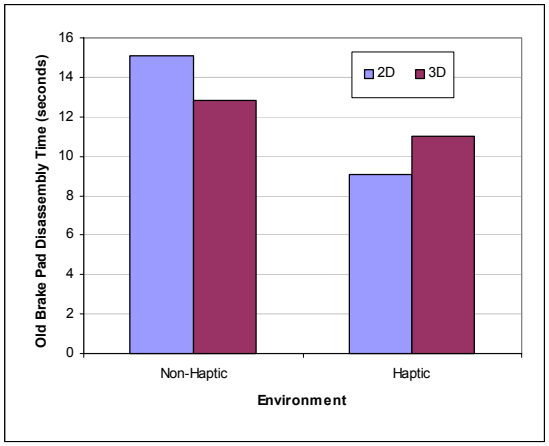


Figure 6.24: Average Disassembly Time for the Old Brake Pad versus Availability of Haptic Feedback

The next set of response variables analyzed was the grasping and assembly time for the new brake pad. Again, a balanced ANOVA was carried out for each of these variables against the control factors. As can be seen in Table 6.10, none of the control factors significantly affected the time required to grasp the new brake pad.

Table 6.10: Balanced ANOVA Comparing ln(New Brake Pad Grasp Time) versus the Controlled Variables for the Brake Pad Replacement Experiment

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	0.53	0.472
Haptics?	1	0.9	0.349
Environment Sequence	3	0.93	0.434

As with the disassembly time for the old brake pad, the type of visualization and environment sequence had no effect on the assembly time of the new brake pad, but the availability of haptic feedback did.

Table 6.11: Balanced ANOVA Comparing ln(New Brake Pad Assembly Time) versus the Controlled Variables for the Brake Pad Replacement Experiment

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	0.01	0.922
Haptics?	1	12.45	0.001
Environment Sequence	3	0.55	0.649

In Figure 6.25, the average assembly time of the new brake pad is plotted against the availability of haptic feedback for both 2D and 3D visualization. The data in this figure shows a decrease of 6 seconds for assembly time of the new brake pad from the non-haptic to the haptic version of the simulation, corresponding to a 36% reduction in required time.

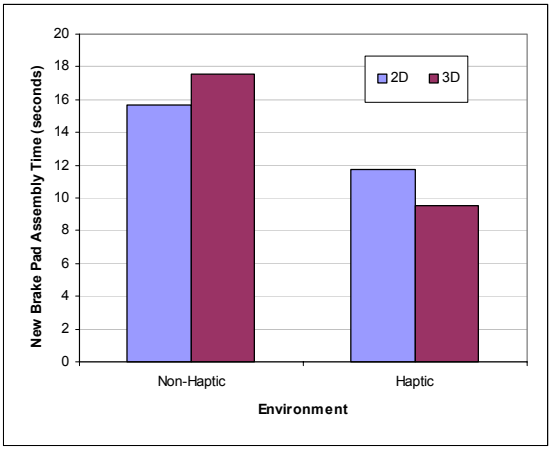


Figure 6.25: Average Assembly Time for the New Brake Pad versus Availability of Haptic Feedback

The last response variable analyzed for the brake pad replacement experiment was the total task completion time. The result of a balanced ANOVA for this variable against the control factors is shown in Table 6.1.

Table 6.12: Balanced ANOVA Comparing ln(Total Task Time) versus the Controlled Variables for the Brake Pad Replacement Experiment

Factor	Degrees of Freedom	F-statistic	p-value
2D or 3D?	1	< 0.005	0.968
Haptics?	1	14.52	< 0.0005
Environment Sequence	3	1.2	0.320

Again, the only significant factor influencing task completion time is the availability of haptic feedback. In Figure 6.26, the average time required to complete the brake pad replacement is plotted against the availability of haptic feedback for both types of visualization. In this figure, there is a clear difference between completion times with haptic feedback when compared to no haptic feedback. The average time for brake pad

replacement in the non-haptic version of HIDRA was approximately 50 seconds, whereas the task was completed in an average of 39 seconds with haptic feedback. Also, notice that in each environment, haptic and non-haptic, the difference in completion time between 2D and 3D visualization is negligible.

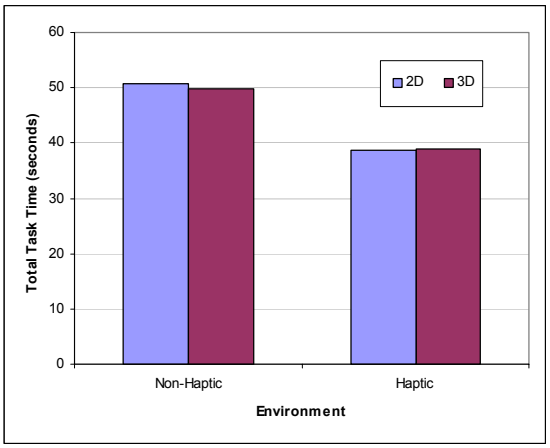


Figure 6.26: Average Task Completion Time for the Brake Pad Replacement Experiment versus Availability of Haptic Feedback

6.4.3 Discussion and Validation of Results

In this section, a discussion of the experimental results will be provided. Below are the significant findings resulting from the brake pad replacement experiment.

- As if a recurring theme, haptic feedback once again improved performance of the experimental participants in the form of reduced task completion times. In this experiment, this trait was shown for both assembly and disassembly tasks.
- The type of visualization, 2D or 3D, was not determined to be significant in any of the analyses. This was also common among all three experiments presented in this chapter. One would expect that stereoscopic vision would improve depth

perception, and thus make assembly and disassembly tasks easier and faster. However, this was not the case. One possible reason for this lack of improvement when using 3D visualization is that the added depth perception provided by the shutter glasses is not realistic enough. However, most people who use the shutter glasses comment on the improved ability to see in 3D, so this is not a likely reason. A more likely explanation is that the positioning guides (discussed in Section 2.3.2) implemented in the 2D version of HIDRA greatly enhanced the user's ability to perceive depth without stereoscopic vision, closing the gap between the 2D and 3D versions of the simulation. This is not to say the 3D visualization is not useful in a virtual environment. In fact, referring back to the motion tolerance experiment, 3D visualization was a significant factor for increasing the rate of correct responses when determining the motion tolerance difference of the peg. In the experiments discussed in this chapter, the improvement in using 3D visualization was not apparent.

- Neither 3D visualization nor haptic feedback improved grasping time of the objects in this experiment. It is not necessarily surprising that haptic feedback did not affect grasping time, since haptic feedback will not actually be received until an object is grasped. The probable reason why type of visualization did not affect grasping time was discussed previously, relating to the use of positioning guides in the 2D environment.
- Early in the analysis, environment sequence was shown to a significant factor influencing the grasp time of the old brake pad. As mentioned, this is most likely attributed to learning, in the sense that, from one environment to the next,

participants became more familiar with the simulation configuration and could locate and grab the old brake pad more easily.

- A very important result, not directly related to this experiment, is the simple fact that the assembly and disassembly procedures involved in the train track experiment and this brake pad replacement experiment were possible. Versions of HIDRA prior to this research would not have been able to handle the object interactions necessary for simulation these scenarios. This provides indirect evidence of the success of the supplemental techniques for object interactions developed in Chapter 4.

Most of these findings have commonalities with the previously discussed experiments. This is a positive sign, in the evidence provided by each experiment can be combined to form more compelling arguments. A discussion of all of the findings from all experiments will be provided in the next chapter and related to the research questions and hypotheses. Similar to all previous experiments, the validity of the ANOVA model used to analyze the data was insured through an analysis of the residuals (see Appendix C.3).

6.5 A SURVEY OF EXPERIMENT PARTICIPANTS

Following the completion of one or more of the assembly and disassembly experiments described in this chapter, participants were asked to fill out a survey. The survey was designed to get feedback from users of HIDRA about their experiences while completing the experiment(s). Several of the questions were aimed at determining the level of expertise users had with computers, virtual environments, haptics, and stereoscopic shutter glasses. The goal of the remaining questions was to gauge the

realism or usefulness of different aspects of the simulation, including haptic feedback, object interactions, and 3D visualization.

A summary of the responses to the questions in the survey is shown in Table 6.13. The individual responses from each experimental subject are supplied in Appendix B.6. Also, the actual questionnaire given to each participant is provided in Appendix A.7. In addition to the survey questions, users were provided space to give open feedback. Most of the feedback received in this way was associated with one of the questions. This open feedback will be included in the discussion of the appropriate question.

Table 6.13: Summarized Responses from Experimental Participant Survey

Survey Question	Average Response
1. How would you rate your knowledge and experience with computers?	2.9
2. How would you rate your knowledge and experience with virtual reality environments?	1.6
3. Have you ever used haptic technology or haptic devices?	41% Yes / 59% No
4. How would you rate your knowledge with haptic technology and experience in using haptic devices?	1.4
5. When compared to your experiences manipulating objects in real life, how *similar* were manipulations of virtual objects through haptic feedback?	2.6
6. When compared to your experiences manipulating objects in real life, how *easy* were manipulations of virtual objects through haptic feedback?	2.5
7. How realistic do you feel the interactions between virtual objects were (i.e., Virtual Object #1 interacting with Virtual Object #2)?	3.0
8. Have you ever used 3D shutter glasses?	29% Yes / 71% No
9. How much of an improvement did the 3D shutter glasses provide over 2D monitor viewing for interacting with the virtual environment?	2.4
10. How much of an improvement did haptic feedback provide over no haptic feedback for interacting with the virtual environment?	3.8

For most of the questions in the survey, participants were asked to respond on a scale of 1 to 4, with four being the highest or best based on the given question. Questions 3 and 8 required a 'yes' or 'no' answer, along with a brief explanation when 'yes' is chosen. An explanation and discussion of each question and the corresponding responses is provided below.

1. When asked about their knowledge and experience with computers, the participants responded with an average of 2.9, meaning that as a group they felt more proficient than the average computer user. (Scale: 1=Novice, 4=Expert)
2. The participants' knowledge and experience with virtual reality environments was much lower, with an average response of 1.6. Most of the participants have never or don't regularly work or use virtual environments. (Scale: 1=Novice, 4=Expert)
3. 41% of the participants stated that they had used haptic devices in the past, though when asked for an explanation, almost all referred to seeing or playing with demos of HIDRA and haptic devices for gaming, such as a force feedback joystick.
4. The participants' average response to the next question is more indicative of their knowledge and experience with haptic technology. The group response of 1.4 indicated that the participants, as a whole, were novices with haptic technology and devices. (Scale: 1=Novice, 4=Expert)
5. Users were then asked to rate how similar manipulating objects in HIDRA were to the real world. The average response of 2.6 indicated that object manipulations were somewhat similar in the virtual environment as compared the real world. This question was asked to gauge how realistic the PHANToM devices are in

mimicking the real world. Some people noted that the PHANToM-human interface, a thimble, is either too loose or too tight, making it uncomfortable. In addition, the orientation of the dual-PHANToM setup required an awkward hand position. It was also noted that haptic feedback is very useful for indicating a stable grasp. (Scale: 1=Not Similar, 4=Very Similar)

6. The average response to this question, 2.5, indicates that ease of object manipulations does not match that in the real environment, but is manageable. Again, the awkward hand position was noted as making object handling more difficult. (Scale: 1=Not Easy, 4=Very Easy)
7. When asked how realistic object interactions were, participants responded with an average of 3.0. This indicates that users felt the interaction between objects in HIDRA to be more than moderately realistic. Compared to the response for Questions 5 and 6, participants viewed object interactions as being more realistic than the haptic sensation, or the manipulation of objects with the haptic devices. This spells good news for the supplemental techniques for object interaction discussed in Chapter 4, as the techniques implemented substantially improve object interactions while maintaining a favorable level of realism. One person noted a sticking sensation between the objects, however, which is attributed to velocity slowdown compensation. One participant noted that since the virtual objects were limited to translation, and not rotation, interactions with objects through haptic feedback did not match reality as closely. (Scale: 1=Not Realistic, 4=Very Realistic)

8. 29% of the participants responded that they had used shutter glasses before, but, again, almost all had only used them in virtual reality demonstrations.
9. When asked whether 3D visualization with the shutter glasses improved interactions with the virtual environment, the average response was 2.4. Many additional comments were received in this area. Some felt that 3D visualization was worse, some felt it was more helpful, while others still stated that they were unsure. Some participants specifically noted that the positioning guides in the 2D environment were helpful in lining up the fingers with virtual objects. One user noted that the somewhat low refresh rate of the shutter glasses was distracting at times. (Scale: 1=No Improvement, 4=Much Improvement)
10. Lastly, users were asked if haptic feedback provided an improvement when interacting with the virtual environment. With an average response of 3.8, most people found haptic feedback to be extremely useful. Some also noted that working in the virtual environment without haptic feedback could get frustrating. This high rating for the improvement that haptic feedback provides is echoed in the results of the experiments, as haptic feedback was almost always a significant factor in reducing task time. (Scale: 1=No Improvement, 4=Much Improvement)

Another comment received by several participants was that they thought they improved during the experiment and could improve further with more experience. This slight improvement was noticed in some of the experimental results. Overall, the feedback provided by the experiment participants was valuable in assessing the realism of HIDRA, and user preference. Given the results of the survey, it is safe to say that HIDRA is fairly realistic in all areas, more so in object interactions rather than the feeling of

haptic feedback. Also, haptic feedback was most definitely preferred, whereas user preference for 3D visualization was less dominant.

6.6 Effect of Supplemental Techniques Supporting Collision Detection on the Experiments

In Section 4.1 and 4.2 of this dissertation, several techniques designed to support collision detection and make object interactions more robust were discussed. In this section, a discussion of the influence that these supplemental techniques may have had on the results for each experiment is provided. Two of the techniques, bounding spheres for haptic dynamic loading and more efficient simulation workspace interaction, had no effect on any experiment and will be discussed first.

Haptic dynamic loading, discussed in Section 4.2.4, refers to a technique that reduces the computational load on the haptic loop by performing broad phase collision detection between the PHANToM interfaces and virtual objects in the scene. Using bounding spheres for this purpose rather than a PHANToM collision representation was implemented to reduce the computational load on the collision detection library used and has no effect on the interactions between virtual objects. Therefore, bounding spheres for haptic dynamic loading also has no direct effect on object interactions in any of the experiments conducted.

To prevent objects from moving to a position unreachable by the PHANToM interfaces, a simulation workspace must be defined, and all virtual objects must be confined to this workspace. To satisfy this need without an increased computational load on the collision detection library, a spring force was implemented to push objects back into the workspace if they travel outside. During the experiments in this dissertation, all

objects remained within the workspace boundaries, and, therefore, this technique also had no effect on object interactions.

In Table 6.14, we highlight the supplemental techniques that may have affected the user's performance in each experiment. In this table, a checked box indicates that there may have been influence, whereas a box with an $\mathcal{X}$ indicates that there was no affect of the supplemental technique on the given experiment.

Table 6.14: Highlighting the Supplemental Techniques for Collision Detection that may have Affected each Experiment

		Supplemental Technique for Object Interactions			
		Constraint Maintenace	User-defined Constraints	Elimination of Angular Velocity	Velocity Slowdown
Experiment	Weight Sensation	$\mathcal{X}$	✓	$\mathcal{X}$	$\mathcal{X}$
	Motion Tolerance	$\mathcal{X}$	✓	$\mathcal{X}$	$\mathcal{X}$
	Peg-in-Hole Insertion	✓	✓	✓	✓
	Train Track Assembly	✓	✓	✓	✓
	Brake Pad Replacement	✓	✓	✓	✓

In the weight sensation experiment, there were no object interactions. As such, most of the supplemental techniques had no possible influence on the experiment or the results. The only technique that may have had an influence was user-defined constraints. In this experiment, the cubes were confined to translation only to prevent unwanted rotations. It is suspected that this had minimal affect on the user's ability to judge the weight of the cubes, but we can't be sure. It is possible that rotational motion can assist in

judging the weight of a cube, such as feeling the moment of inertia, but probably not as much as simple translational lifting.

In the motion tolerance experiment, user-defined constraints were again implemented to prevent unwanted rotation of the peg as it rested in the hole. Also, if rotation of the peg were possible, the allowable range of motion for the peg would have changed, and the experiment or its results would not have much validity. During the motion tolerance experiment, the motion constraints on the peg were programmed into the simulation, and velocity slowdown was not utilized. For this reason, none of the other supplemental techniques could have influenced the experiment.

The remaining assembly and disassembly experiments conducted in this chapter, peg-in-hole insertion, train track assembly, and brake pad replacement, may have been influenced by all of the techniques listed in Table 6.14. Therefore, the affect of each technique on these experiments will be discussed for all experiments at the same time. For these experiments, as with all other experiments, user-defined constraints were utilized to constrain the rotation of all dynamic objects in the simulation. Surely, the time required to complete all scenarios would have been greater had object rotations been allowed, as objects could rotate out of alignment, making assembly or disassembly more difficult or impossible. Since the ability to handle object rotation with the dual-PHANToM setup is limited and all experimental participants were considered novices with the simulation, it was decided to eliminate object rotations to simplify each task. Having said that, the ability to complete each experiment in the real world would not have been significantly dependent on object rotations, since it is expected that users could quite easily hold the objects in the experiments steady, with negligible rotation.

Therefore, to eliminate the object handling limitation of the haptic setup, rotational constraints were used. Since all objects in the experiments were constrained to linear motion anyway, eliminating angular velocities during collisions had no further effect on the experiments or their results.

In the three assembly and disassembly experiments, the constraint maintenance algorithm was active, providing support for object interactions. Referring back to Section 4.1.4, one of the major limitations of the constraint maintenance scheme is the possible loss of friction between objects constrained to motion with respect to each other. This was most noticeable in the brake pad replacement experiment, during the insertion of the new brake pad into the caliper. Without friction and nothing to stop the motion of the brake pad on the backside of the caliper, it would continue sliding through the caliper after assembly. However, the procedure was considered complete when the new brake pad was pushed flush with the front edge of the caliper, and, therefore, this lack of friction did not affect the results. The other experiment in which lack of friction may have affected the results was in the peg-in-hole insertion procedure as friction between the peg and hole may have been lost during the procedure. However, the effects of velocity slowdown, discussed in the next paragraph, probably affected the assembly and disassembly more so than the lack of friction.

The supplemental technique that probably affected the results of the assembly and disassembly experiments the most was velocity slowdown for objects in close proximity. Since each experiment involved components that interacted in close proximity, the velocities of these components were definitely affected. For example, during the brake pad replacement procedure, removal of the old brake pad and insertion of the new brake

pad was somewhat impeded by velocity slowdown. The effect of this technique on all experiments is increased time required to complete the task, though the impact was probably minimal. Recalling the survey of experiment participants, one person actually felt a slight sticking sensation between the objects, which was attributed to velocity slowdown. Although no other comments were received concerning this sensation, if informed beforehand about the velocity slowdown technique, participants probably would have been able to perceive its effects.

6.7 CHAPTER SUMMARY

In this chapter, a description and the results of three experiments were presented. Volunteers participated in each of the experiments, and task completion times were recorded for analysis. First, the peg-in-hole experiment tested the simple assembly and disassembly procedure of inserting a peg into a hole, and then removing it. The train track assembly experiment provided a more complex study, in which users assembled the sections of a model train track to create on complete section. Lastly, the brake pad experiment offered an example more relevant to the engineering domain, as it simulated the replacement of an old brake pad.

The three experiments were designed and conducted in order to address and answer Research Question 2.3, which questions the usefulness of haptic feedback for assembly and disassembly simulation. In all, the experiments were successful in achieving this goal and more. To provide additional insight into the quality of HIDRA and the usefulness of haptic feedback, a survey was conducted of all experimental participants. The results of this survey provide additional support for the effectiveness of haptic feedback and quality of object interactions.

In Chapter 7, a response will be generated for all research questions and hypotheses based on the information provided in this dissertation. In addition, a discussion of the achievements resulting from this work, the limitations of the research, simulation, and experiments, and possible avenues of future work will be discussed.

CHAPTER VII

ACHIEVEMENTS AND RECCOMENDATIONS

In this dissertation, the development and testing of the HIDRA simulation environment, which incorporates haptic feedback for assembly and disassembly analysis, has been presented. In this chapter, we provide closure to the research conducted and discussed throughout the dissertation. In Section 7.1, an overview of the research questions developed in Chapter one is presented, followed by a response to each question through testing of its corresponding research hypothesis. A review of the contributions from this research is provided in Section 7.2. Finally, the limitations of this research and areas for future work are discussed (Section 7.3).

7.1 ANSWERING THE RESEARCH QUESTIONS

To this point, this dissertation has been focused on presentation, analysis, and discussion of the research conducted. In this section, focus is returned to the research questions, as the research findings will be used to answer these questions through their corresponding hypotheses.

7.1.1 Research Question Overview

As stated in Chapter 1, the main objective of this research was to evaluate the usefulness of haptic feedback for assembly and disassembly simulation in a virtual environment. The research was conducted in two stages: evaluation and improvement of object interactions within HIDRA and experimentation to determine the influence of haptic feedback during various procedures in the simulation. As such, two primary research questions were developed and are shown below.

> ***Question 1:*** *How can collision detection between arbitrary non-convex objects be improved to support the dynamic real-time simulation of such objects for haptic assembly and disassembly evaluation?*
>
> ***Question 2:*** *Does a haptically enabled simulation environment provide a significant improvement over a purely visual simulation for assembly and disassembly evaluation and related engineering issues?*

To answer these primary questions, the following hypotheses were proposed.

Hypothesis 1: *Significant improvements can be made over current collision detection libraries for the interaction between arbitrary non-convex objects for dynamic real-time simulation, particularly for haptic assembly and disassembly evaluation.*

Hypothesis 2: *A haptically enabled simulation environment does provide a significant improvement over a purely visual simulation for assembly and disassembly evaluation and related engineering issues.*

Each of the primary research questions was expanded further to provide more manageable questions that could be answered more directly through the research described in the preceding chapters. The sub-questions associated with each of these questions is provided below.

Question 1.1: *Do current collision detection libraries provide the usability and functionality necessary to efficiently handle dynamic object interactions, and, if not, how can they be improved?*

Question 1.2: *How can current collision detection libraries be supplemented to improve object interactions within HIDRA, while limiting detrimental effects and preserving the realistic quality of the simulation?*

Question 1.3: *How can a collision detection library be integrated with the HIDRA simulation to maximize the collision performance?*

Question 2.1: *How does the perception of weight in a virtual environment with haptic feedback compare to that in the real environment?*

Question 2.2: *How does haptic feedback affect the ability of a user to detect motion tolerances in a virtual environment?*

Question 2.3: *Does haptic feedback provide a significant improvement over a purely visual simulation for assembly and disassembly operations?*

Again, individual hypotheses were formulated for each research sub-question. These sub-hypotheses are listed below.

Hypothesis 1.1: *Public domain software libraries for collision detection provide the usability and functionality to simulate dynamic object interactions, while there are areas for improvement.*

Hypothesis 1.2: *Techniques such as constraint integration, velocity slowdown, and others can be implemented to improve object interactions in HIDRA.*

Hypothesis 1.3: *Programming HIDRA using threads offers the most flexibility and maximizes performance for integration of a collision detection library.*

Hypothesis 2.1: *The perception of weight and ability to distinguish between weights in a virtual environment closely matches that in the real environment.*

Hypothesis 2.2: *In a virtual environment, haptic feedback improves a user's ability to detect differences in the range of motion of an object when compared to a purely visual simulation.*

Hypothesis 2.3: *Haptic feedback can provide significant improvements over a purely visual simulation for assembly and disassembly operations by reducing task completion time.*

Given this refresher of the dissertation research questions and hypotheses, answers to each question will be formulated in the next section. The final response to each question will be based on testing of its associated hypothesis.

7.1.2 Answering Research Questions

The answer to each research question will be based on satisfying the corresponding hypothesis. In this section, all research questions formulated in this dissertation will be answered based on the results detailed in the previous chapters. In order to answer each primary research question, the sub-questions will be discussed first.

- **Answer to Research Question 1:**

The answer to the three supporting questions for this research question will be presented, followed by a response to the primary research question.

Answer to Research Question 1.1:

The response to this question is based mainly on the research presented in Chapter 3, specifically Sections 3.2.3, 3.2.4, and 3.3.2. In order to model object interactions in an impulse-based simulation, collision detection is necessary. To date, a small number of collision detection libraries have been developed for this purpose. In Section 3.2.3, public domain software libraries for collision detection were introduced. The components of a collision detection library necessary to simulate the interaction between dynamic objects were then discussed in Section 3.2.4, followed by a qualitative review and comparison of the most functional libraries. It was determined that at an absolute minimum a collision detection algorithm should be able to determine the exact distance between two objects and provide closest point information. In addition, the ability to return depth of penetration for intersecting objects, the ability to model non-convex objects, the ability to build the collision representation directly from the objects geometry information were positive features. Based on this review, it was determined that several, but not all, of the public domain collision detection libraries provide the necessary capabilities for dynamic object simulation using physically based modeling. However, depth of penetration was only available with one library, and the ability to model non-convex objects was infrequent. Overall, a small number of the more recently developed collision detection libraries provide the functionality necessary to model dynamic object interactions, but many significant improvements are still to be made.

All of the collision detection libraries provide some form of user interface, through function calls that update object positions and query the collision detection routine. However, methods of implementation differ between libraries. For instance,

some libraries utilize pair processing algorithms while others use n-body collision detection. In Section 4.4, the advantages and disadvantages of each in terms of usability were discussed. Also, the method in which the collision representations of objects are created varies between libraries. Although a few libraries provide support for non-convex objects, these objects must be treated differently, in the form of preprocessing, from their convex counterparts. This feature of collision detection libraries can be made more useful.

With respect to Hypothesis 1.1, it was shown that more recent public domain software libraries for collision detection do in fact provide the necessary functionality and usability to handle the interaction between dynamic objects in simulation. However, many areas for improvement were cited and discussed.

Answer to Research Question 1.2:

The research conducted to test this sub-hypothesis was presented in Chapter 4, Sections 4.1, 4.2, and 4.4. As discussed in Section 2.4, there are many characteristics of HIDRA that hinder the performance of collision detection libraries and response algorithms in effectively modeling object interactions. For instance, the complex computations involved in collision detection are commonly considered a computational bottleneck during real-time simulation. The introduction of haptic interaction within a simulation introduces additional complications for collision detection, as the forces on an object are unpredictable and can be very large, resulting unpredictable object motions and a reduced efficiency of the collision detection library. To support collision detection libraries and improve the success of HIDRA in modeling object interactions, several supplemental techniques were developed.

A constraint maintenance scheme was discussed in Section 4.1 that managed constraints between objects to reduce the dependency of object interactions in HIDRA on collision detection alone. Also, various other techniques including velocity slowdown compensation and user-defined constraints were introduced. The success of these techniques became apparent in Chapter 6 during the relatively complex train track assembly and brake pad replacement. Prior to the development of these supporting techniques, these scenarios were not possible, as object interactions would routinely fail. Furthermore, a survey of experimental participants (Section 6.5) determined that users felt object interactions in HIDRA were fairly realistic. In fact, object interactions were viewed as more realistic than haptic interaction with objects. Only one participant noted an unusual interaction between objects (a slight sticking sensation), resulting from the velocity slowdown simulation.

Overall, the supplemental techniques introduced in HIDRA to support collision detection not only improved object interactions, but did so with little detrimental effect as noticed by simulation users.

Answer to Research Question 1.3:

The research conducted to test Hypothesis 1.3 was presented in Section 4.3. The method in which a collision detection library is integrated within a real-time haptic simulation of HIDRA can significantly affect the effectiveness of the library. Three methods for implementing collision detection in HIDRA were considered, compared, and analyzed through simulation data. Based on this analysis, thread programming was determined the most effective way to implement collision detection for highest efficiency. Thread programming allows the separation of the three main components of a

haptic simulation of dynamic objects: collision detection, haptic interaction, and graphic display.

Overall Response to Research Question 1:

Based on a confirmation of each of the sub-hypotheses above, the overall response to Research Question 1 can be formulated. For an impulse-based simulation such as HIDRA, collision detection is a necessary and important component. The research presented in Chapters 3 and 4 has supported Hypothesis 1 in that significant improvements can be made over current collision detection libraries in the form of usability and functionality. In addition, the capability to support collision detection through constraint maintenance and other techniques can also improve object interactions. Lastly, an appropriate high-level simulation architecture will allow collision detection libraries to be most effective.

- **Answer to Research Question 2:**

As with the first research question, Question 2 has three supporting questions. Answers to these supporting questions will be presented first, followed by a response to the second primary research question.

Answer to Research Question 2.1:

Research Hypothesis 2.1 was tested with the weight sensation experiment in Section 5.2. During the experiment, users were asked to compare the weight of two cubes in the real world and in a virtual environment. The results, in the form of correctly identifying the heavier cube and response time, showed that users were more adept at weight perception in the real world when compared to the virtual environment. As such, this was the only research hypothesis in this dissertation that could not be confirmed.

Although it was determined that the ability to perceive weight differences in the virtual environment is less efficient, the ability to perceive weight at all would not be possible without haptic feedback. The second part of this research question was aimed at determining whether the perception of weight for an object in the virtual environment is the same as that for a real object with the same weight. The results for this experiment were inconclusive as a phenomenon known as the time-order effect, in which the second object lifted during a comparison of weight will generally feel heavier for objects of the same mass. Taking into account the time-order effect, it is believed that additional experiments would show that an object in the virtual environment feels the same in weight as an object with equal mass in the real world.

Due to the results of the weight sensation experiment, Research Hypothesis 2.1 cannot be accepted.

Answer to Research Question 2.2:

The research conducted to test this hypothesis was presented in Section 5.3. Similar to Research Question 2.1, an experiment was designed, participants were tested, and the results were analyzed to determine the influence of haptic feedback and other factors on the ability of a person to make judgments on the allowable range of motion for an object in two directions. During the experiment, two response variables were recorded: decision time and tolerance judgment. It was found that haptic feedback significantly reduced the time required by the user to make a judgment on the motion tolerances. However, haptic feedback was not influential in improving the rate of correct responses given by the participants, although analysis indicates that further testing may reduce the variability of the data, resulting in a dependence on haptics. The other major factor in the

experiment, type of visualization, affects the response variables in just the opposite way, suggesting an improvement in judgments for 3D visualization, but no reduction in decision time. In other results, the magnitude of the tolerance difference and direction of greater tolerance significantly affected both response correctness and decision time. Lastly, this experimented the first signs of learning occurring during the experiment as the decision time was reduced from one environment to the next.

With evidence from the motion tolerance experiment, Research Hypothesis 2.2 is accepted. Haptic feedback improves a user's ability to detect motion tolerance differences in a virtual environment by reducing the time required to make the judgment.

Answer to Research Question 2.3:

The research conducted to test the final research question was contained entirely in Chapter 6. To test this hypothesis and answer the research question, three experiments were conducted, all modeling assembly and/or disassembly in one way or another. The first experiment was the relatively simple insertion and removal of a peg from hole. In the next experiment, participants were asked to assemble sections of a model train track to form a continuous segment. Finally, an application more relevant to the engineering domain was studied, as users replaced a worn out brake pad in an automotive brake assembly. The predominant theme in all experiments was that the presence of haptic feedback improved the users ability to complete the task by reducing the time required for assembly/disassembly.

The results of these experiments provide ample evidence that haptic feedback, compared to a purely visual simulation, significantly improves assembly and disassembly

operations by reducing task completion time. Therefore, Research Hypothesis 2.3 can be accepted.

Overall Response to Research Question 2:

Based on the response to each of the sub-hypotheses above, the overall response to Research Question 2 can be formulated. The main goal for this research is to determine the viability of haptic feedback as an integrated component of virtual environment simulations for assembly and disassembly analysis. Through Research Questions 2.1, 2.2, and 2.3 and the experiments described in Chapters 5 and 6, sufficient evidence has been provided to suggest that haptic feedback does indeed improve a users ability to perform assembly and disassembly operations in a virtual environment. In addition, the general interaction with objects modeled in a virtual environment is improved through the implementation of haptics. Therefore, Research Hypothesis 2 is accepted.

7.2 ACHIEVEMENTS: REVIEW OF RESEARCH CONTRIBUTIONS

The expected contributions from this work were introduced in Section 1.4.3 of this dissertation. In this section, a discussion of the contributions that resulted from this work will be discussed, linking them to the chapters in which they were covered.

- *HIDRA* – The simulation developed during this and previous work (McDermott 1999) is the only known real-time assembly and disassembly simulation for arbitrary 3D objects that implements the dual-PHANToM interface and physically based modeling incorporating high-precision collision detection. Aside from combined two PHANToMs to allow grasping, HIDRA is one of the few assembly/disassembly simulations that provides the ability to grasp objects through haptic feedback. An overview of HIDRA was provided in Chapter 2, and

additional detail on object modeling techniques and the improvements made were presented in Chapters 3 and 4. Finally, a demonstration of the simulation's capabilities was provided through experimentation in Chapters 5 and 6.

- *Constraint Maintenance* – A method to integrate constraints in an impulse-based simulation of dynamic objects was presented in Section 4.1. Impulse-based simulations are not as efficient in modeling objects in close quarters, since an infinite number of collisions can occur between two objects in such a configuration. During the discussion of constraint maintenance, the equations to define and apply each constraint were provided. These constraints, implemented in conjunction with collision detection, improve the reliability of object interactions within HIDRA.
- *Techniques to Support Collision Detection* – In addition to constraint maintenance, a number of supplemental techniques was designed and implemented to provide more reliable object interactions in situations where a collision detection library alone fails. This set of techniques was discussed in Section 4.2.
- *A Review on Collision Detection* – Current collision detection and response algorithms are a necessary tool for the successful implementation of a simulation such as HIDRA. There are many features of such libraries that are essential to modeling object interactions and others that could make integration more effective and reliable. In Chapter 3, we provided a detailed review of current collision detection libraries, their capabilities, and their limitations with respect to

usability and functionality. This review offers a basis for which future research efforts can be directed to improve these libraries.

- *Empirical Evidence Supporting Haptic Integration* – The experimental studies presented in Chapters 5 and 6 provide direct evidence for the effectiveness of haptic feedback in HIDRA. The data shows that users consistently completed tasks in shorter time when haptic feedback was available. The results of these experiments support the argument for haptic integration in virtual environments for assembly/disassembly and others as well.
- *Experimental Participant Survey* - Aside from the numerical performance users during experimental testing, a participant survey was conducted to get feedback on the effectiveness and realism of different aspects of the simulation (Section 6.5). This is important because it takes into account the human factor. Although the data suggests that collision detection is improved and haptic feedback is useful, technologies are not accepted without the support of the people who use them. Participants overwhelmingly preferred haptic feedback as opposed to a purely visual environment during testing scenarios, supporting its integration into simulation. In addition, this survey was used to provide support for the other research contributions. For instance, the relative realism of object interactions indicated by the survey results confirms the successful integration of the supplemental collision detection techniques without alarming detrimental effects.
- *Haptic Technology* – Through completion of this dissertation, knowledge about the PHANToM haptic interfaces, and their use in a dual configuration, was gained. For instance, modeling haptic feedback using point force interaction can

severely limit the ability to manipulate dynamic objects. More discussion on the limitations of haptic technology will be discussed in the next section.

Having reviewed the research contributions of this dissertation, the final section of this document will discuss the limitations of this research and avenues for future work.

7.3 LIMITATIONS OF THE RESEARCH AND FUTURE WORK

With the answers to all research questions having been developed and a review of the contributions of this research, a critical evaluation of the work will be provided in this section. For ease of discussion, avenues for future work will be discussed as well since most areas for future work are directly related to the limitations of this research or the technologies used. Also, each limitation and/or future work will be grouped under on of several research areas.

- **The HIDRA Simulation**
  - One of the general simulation improvements discussed in Section 2.3 of this dissertation is allowing the user to fly through the virtual environment. This enhancement allows users to gain additional perspectives of parts during an assembly or disassembly procedure. The current method for controlling user position in the simulation is through the keyboard. Although seemingly straightforward, this can sometimes inhibit progress during a scenario, as the user's focus must be shifted toward the keyboard to modify viewing angle. A less distracting method could use the position of the PHANToM cursors in the simulation to automatically adjust viewing angle by maintaining a straight line between the user's viewing position, a point midway between the PHANToM cursors, and the center point of the simulation. Other possible methods of control

include controlling the viewing angle by a tracking device mounted on the user's head or voice recognition to command an orientation change.

- Currently, the HIDRA simulation obtains information for object representations via CAD transfer through VRML. When transferring data, much of the information with regards to assemblies is lost, such as assembly relationship between parts. One avenue of future work could be to extract these assembly relationships from the CAD file representing the assembly. Given these relationships, a technique for assisted assembly could be developed. For example, when attempting to insert a bolt through a hole, the centerline axes of the bolt and hole must line up. Given this alignment relationship from a CAD file representing the assembly, the simulation could exert 'invisible' forces on the bolt to line it up with the hole, in a form of assisted assembly.
- Typical assembly and disassembly procedures involve the use of tools to connect parts together or remove them from one another. Currently, HIDRA only allows assembly/disassembly of components through hand manipulation. The modeling of tools such as a screwdriver or wrench could expand the array of scenarios possible. Along the same lines, fasteners are important components in nearly all assemblies. HIDRA does not currently attempt model fasteners during the simulation. All experiments conducted during this dissertation ignored all fasteners. By integrating fasteners in HIDRA, more complex assemblies could be modeled, assuming that the computational power is available.
- One of the major difficulties users typically have when using HIDRA is with respect to the perception of depth. Some steps have been taken to alleviate this

problem such as positioning guides (Section 2.3.2) and stereoscopic vision using shutter glasses (Section 2.3.3). As an additional step, shadows could be added to the virtual environment for improved depth perception and increased realism. Again, this assumes that the computer is fast enough to handle the increased computational demand.

- **Object Interactions**
  - The constraint maintenance scheme developed in Section 4.1 is very effective in improving object interactions and the robustness of the simulation. However, currently, the method is implemented for translational constraints only. Future work could develop and integrate a technique for rotational constraints.
  - As discussed in Chapter 3, there are many areas in which current collision detection libraries can be improved. These include support for depth of penetration calculations and use of advanced bounding volume hierarchies around objects for more efficient broad phase collision detection.
- **Experimental Studies**
  - All of the experiments conducted in this dissertation allowed only translational movement of the virtual objects. The decision to eliminate rotation resulted from the difficulty in handling object rotations, as discussed in Section 2.4.2. An idea for future work to improve the handling of object rotations is discussed below under the haptics subheading. Given an improved ability to handle rotations, additional experiments could be conducted, including this factor in the studies.
  - The experiments conducted in this dissertation compared haptic and non-haptic versions of HIDRA. In both versions of the simulation, users manipulated objects

through the dual-PHANToM interface, which allows the grasping of virtual objects. During the experiments, it became apparent that users had a more difficult time handling objects in the non-haptic version of the simulation because they could not feel their grasp. An alternative to the dual-PHANToM setup would be to use only one PHANToM and a point-and-select method of moving objects, in which a selected object simply follows the motion of the PHANToM. By using a single PHANToM and the point-and-select technique all object handling difficulties would be eliminated. There were two reasons why experiments did not test this method during the research conducted for this dissertation. The first is that HIDRA was designed with the dual-PHANToM interface for the explicit reason of incorporating object grasping and an increased sense of realism to the simulation. Many non-haptic environments currently use interfaces such as tracking gloves that require a user to grab virtual objects without haptic feedback. The limitation in a technique where a virtual object follows the PHANToM position occurs when a collision response is generated between two objects in the simulation. If haptic feedback is not active, the user can push the PHANToM past the point of allowable motion for the object selected and the object will no longer be able to move, despite the user's efforts. The second reason a single PHANToM point-and-select method was not implemented is a limitation in the version of GHOST implemented in the simulation. To implement the point-and-select method, the simulation must have the capability to directly control the force signal to the PHANToM. This is necessary to respond to forces generated through collisions between virtual objects, which is modeled separately from the GHOST

software. The capability to apply forces directly to the PHANToM is not provided in HIDRA's version of GHOST.

- For Experiment 3 of the weight sensation study conducted in Section 5.2, the weight of a real object was compared to the weight of a virtual object. The results of this experiment were inconclusive, however, since the experimental design did not take into account the time-order effect of comparing the weights of two objects. An additional experiment could be conducted, with the time-order effect accounted for, to determine the true relation between the perception of weight in the virtual environment and that in the real world.
- Clearly, there are a countless number of experiments that can be conducted using the HIDRA simulation. The design, implementation, and analysis of such experiments should be considered as areas for future work.

- **Haptics**
  - The major limitation of the PHANToM haptic interface, as well as many other haptic devices, is the fact that manipulations of virtual objects are modeled as point-force interactions. This characteristic makes the control of rotation very difficult. One method that may alleviate this problem is using a point cloud to represent the user's finger pad. Instead of modeling the haptic interface with one point, several points could cover the virtual surface of the finger pad. The forces generated by interactions of all points in this representation with virtual objects could then be averaged to compute an overall force applied to the finger at the PHANToM interface. This technique may provide the user with a slight improvement in the ability to handle objects and control rotation. As an extension,

the entire fingertip could be modeled using a cloud of points, and one force could be calculated from all point-object interactions. The advantage that this would have is that the users virtual fingertips would no longer be able to slide between two virtual objects too close for the human finger to fit. However, implementation of these techniques would require direct control over the forces delivered to the PHANToMs, a capability not allowed by the version of GHOST used in the simulation.

- Another limitation in using the dual-PHANToM setup to model grasping of virtual objects was mentioned during the user survey conducted in Section 6.5. One participant noted that thin objects were more difficult to grab due to interference between the thimble devices of the PHANToMs. As the two PHANToM arms approached one another while the user attempted to grab a relatively thin object, the thimble interfaces would sometimes touch each other, disturbing the grasping process. There are two factors that cause this difficulty: the physical dimensions of the haptic interface and the flexibility in the PHANToM arm. Due to these characteristics of the haptic interfaces, a virtual object such as a piece of paper or a thin plate would be nearly impossible to grasp.

Within the suggestions for future work are a wide variety of areas open for study and research. Clearly, the motivating example given in Section 1.2.3 of an oil filter replacement is far beyond the capabilities of HIDRA or any haptic simulation. However, through many years of research, one may eventually be able to evaluate the replacement of an oil filter in a virtual environment with full-body haptic interaction.

APPENDIX A

PARTICIPANT INSTRUCTIONS FOR EXPERIMENTAL STUDIES

To start all experiments, each participant was given a set of typewritten instructions. For completeness of work, the instructions given for each experiment are provided in this appendix.

- **Appendix A.1:** Weight Comparison of Two Cubes (real or virtual, as the same set of instructions was used for both experiments)
- **Appendix A.2:** Weight Comparison of a Real and Virtual Cube
- **Appendix A.3:** Motion Tolerance Study
- **Appendix A.4:** Peg-in-hole Placement Experiment
- **Appendix A.5:** Assembling a Toy Train Track
- **Appendix A.6:** Replacing an Automotive Brake Pad
- **Appendix A.7:** HIDRA Experimental Participant Questionnaire (filled out by participants upon completion of experiments using HIDRA)

A.1 WEIGHT COMPARISON OF TWO CUBES (REAL OR VIRTUAL) – INSTRUCTIONS

During this experiment, you will be presented with a pair of cubes and will be asked to compare the weight of the cubes. The first cube (Cube #1) will remain the same throughout the experiment. The second cube (Cube #2) will be interchanged after each comparison. Cube #1 will be considered the base cube, against which you will compare all of the other cubes presented. Possible responses include *less than*, *equal to*, or *greater than*. In other words, for each pair of cubes you will state whether the weight of Cube #2 is *less than*, *equal to*, or *greater than* the weight of Cube #1.

After a pair of cubes is placed in front of you, you are asked to compare the weights of each cube. Please follow the rules listed below:

- You must use the same hand to pick up all cubes.
- You must use your forefinger and thumb to pick up the cube on its sides.
- You must always pick up Cube #1 (i.e., the base cube) first.
- You may only pick up one cube at a time, but may pick up each cube more than once.

When you have made your decision on the weight comparison, place all cubes on the table and verbally state your response (*less than*, *equal to*, or *greater than*) so that it can be recorded. Remember, your response should fill in the blank of the following sentence:

The weight of Cube #2 is _____ *the weight of Cube#1.*

If you have any questions, please ask.

A.2 WEIGHT COMPARISON OF A REAL AND VIRTUAL CUBE – INSTRUCTIONS

During this experiment, you will be presented with a pair of cubes and will be asked to compare the weight of the cubes. The first cube (Cube #1) is a physical cube (i.e., in the real, non-virtual environment). The second cube (Cube #2) is a model of a cube of the same size in the virtual environment. Cube #1 will be considered the base cube, against which you will compare the weight of the cube in the virtual environment. Possible responses include *less than*, *equal to*, or *greater than*. In other words, you will state whether the weight of Cube #2 is *less than*, *equal to*, or *greater than* the weight of Cube #1.

When instructed, you are asked to compare the weights of the real cube and the virtual cube. Please follow the rules listed below:

- One hand will be used to pick up the real cube, while your other hand will be used to pick up the virtual cube.
- You must use your forefinger and thumb to pick up each cube on its sides.
- You must always pick up Cube #1 (i.e., the real cube) first.
- You may only pick up one cube at a time, but may pick up each cube more than once.

When you have made your decision on the weight comparison, replace all cubes to their resting position and verbally state your response (*less than*, *equal to*, or *greater than*) so that it can be recorded. Remember, your response should fill in the blank of the following sentence:

The weight of Cube #2 is _____ *the weight of Cube#1.*

After your response is recorded, the weight of Cube #2 will be modified, and the procedure will start again.

If you have any questions, please ask.

A.3 MOTION TOLERANCE STUDY – INSTRUCTIONS

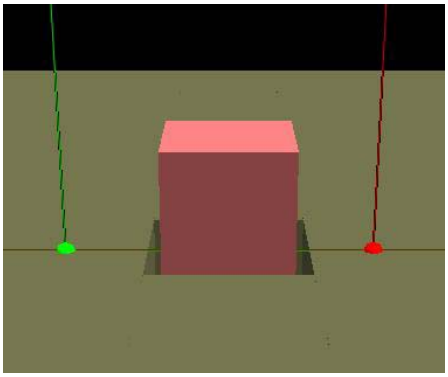
During this experiment, you will be presented with a virtual environment containing a square peg that rests in a square hole in the floor (see figure below). The peg will be constrained to motion in the plane of the floor (i.e., you will not be able to lift the peg up or remove it from the hole). Your task will be to move the peg from side-to-side and front-to-back and compare the allowable range of motion between the two directions. In other words, you are to determine whether the peg's range of motion is greater from side-to-side or from front-to-back.

This procedure will be completed several times, using different variations of the virtual environment. When assessing the range of motion, you must always grasp the peg on its sides (i.e., on the right and left sides). You are not to move the peg by grasping it on the front and back or the top.

Remember, there are only two possible responses:

1. The peg's allowable range of motion is greater from side-to-side than it is from front-to-back.
2. The peg's allowable range of motion is greater from front-to-back than it is from side-to-side.

If you have any questions, please ask.



Experimental Setup

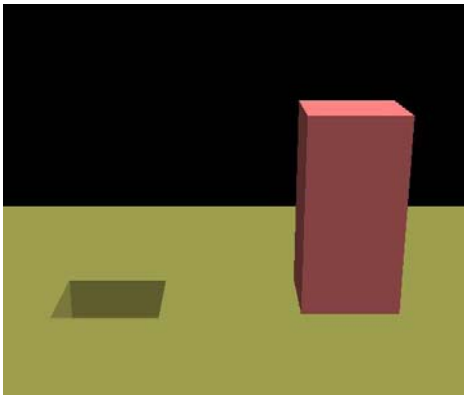
A.4 PEG-IN-HOLE PLACEMENT EXPERIMENT – INSTRUCTIONS

During this experiment, you will be presented with a square peg resting on the floor of the virtual environment. You will be asked to insert the peg into a square hole in the floor. The images provided below show the peg in its initial resting position and inserted in the hole. The procedure described below will be completed several times, using different variations of the virtual environment.

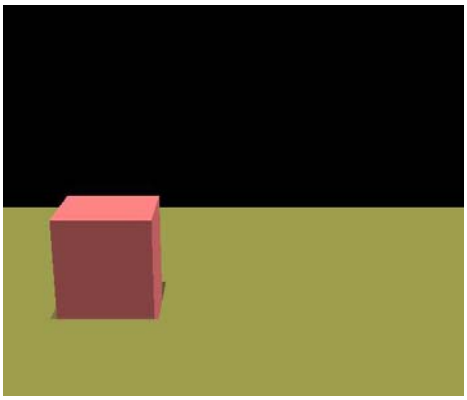
Procedure

1. When instructed, grab the peg from its resting position and insert the peg in the hole until it is fully seated (i.e., touches the bottom of the floor).
2. After the peg is fully inserted, remove the peg and return it to the original resting position.
3. Repeat this task (Steps 1 and 2) five times.

If you have any questions, please ask.



Peg in Initial Resting Position



Peg Inserted in Hole

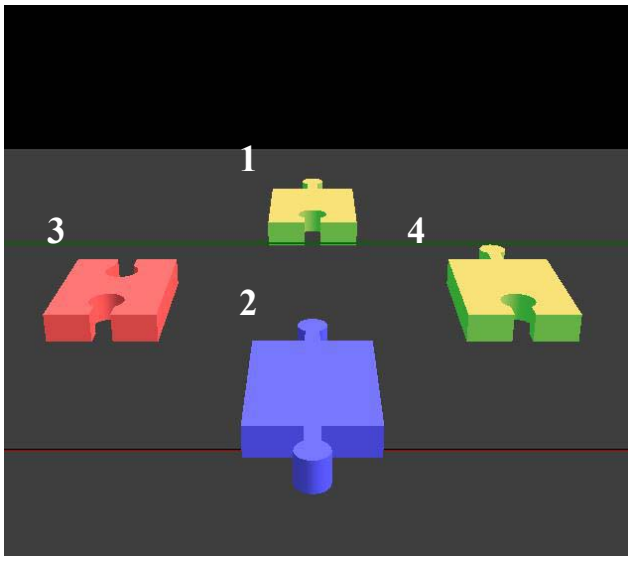
A.5 ASSEMBLING A TOY TRAIN TRACK – INSTRUCTIONS

During this experiment, you will be presented with 4 sections of a toy train track. You will be asked to assemble the individual pieces so that they form a continuous track section. This procedure will be completed several times, using both physical train track pieces (i.e., in the real, non-virtual environment) and virtual train track pieces in a computer simulation. The initial and final configurations of the pieces for the virtual environment are shown in the figures below. This configuration will be the same for all tasks, real and virtual.

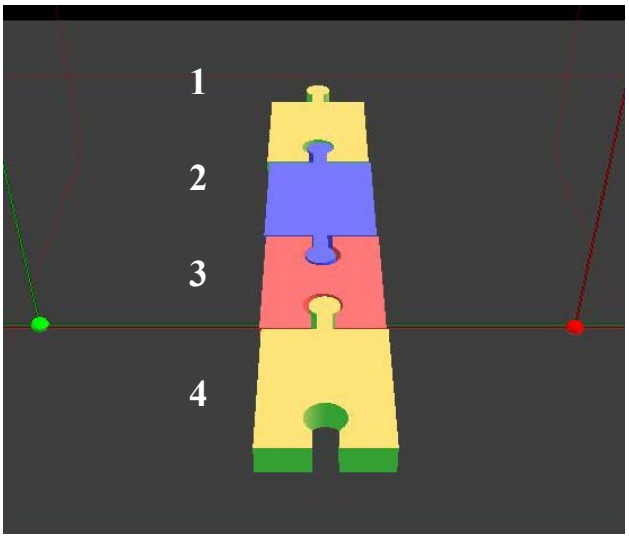
Procedure

1. The first section of the track (labeled 1) is located at the back of the workspace. This piece is already in the correct position for the final configuration, and will not be moved during the procedure. In fact, this piece will be anchored to the workspace so that it cannot be moved.
2. The second section of the track (labeled 2) is located toward the front of the workspace. You are to pick this piece up and assemble it to the first section by mating the female and male connections of each piece.
3. The task will continue with the third section of the track (labeled 3) located to the left in the workspace. Assemble this piece to the second section as described before.
4. The final section (labeled 4) is located on the right side of the workspace. Complete the track by connecting this piece to the third section.

A demonstration will be given. If you have any questions, please ask.



Initial Configuration



Final Configuration

A.6 REPLACING AN AUTOMOTIVE BRAKE PAD – INSTRUCTIONS

During this experiment, you will be presented with a simplified virtual representation of an automotive brake assembly. Your task is to replace a worn out brake pad with the new brake pad provided. Two views of the brake pad assembly are shown on the next page. The first figure shows the initial configuration of components for the procedure (objects labeled 1-5). A description of each of the objects follows: 1 – worn brake pad, 2 – new brake pad, 3 – brake caliper, 4 – piston, and 5 – rotor. The procedure described below will be completed several times, using different variations of the virtual environment.

Procedure

1. First, remove the old brake pad by grasping the tab located on the front edge (closest to observer) of the brake pad and pulling it toward you to remove it from the caliper. After the old brake pad is disassembled from the caliper, place it to the side next to the new brake pad.

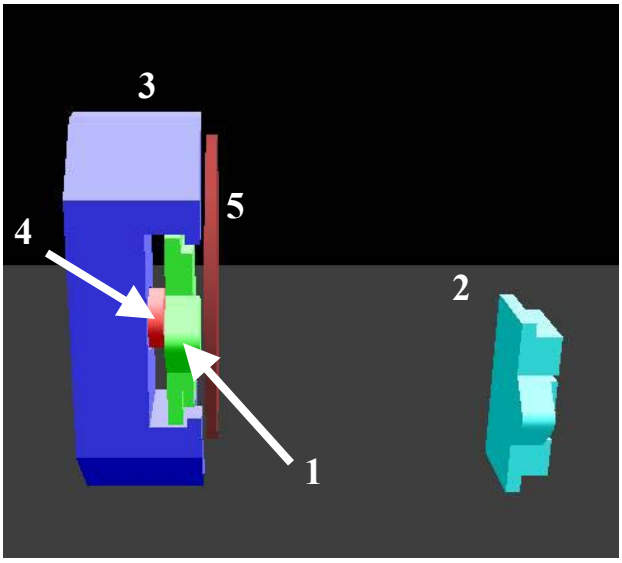
2. To allow room for the new brake pad, push the piston into the caliper until the surfaces of each part are flush. [Note: In a typical brake assembly, the piston is used to push the brake pad toward the rotor to initiate braking. When the old brake pad is worn and needs replacement, the piston is pushed out from the caliper much farther than its original position. This piston must be pushed back to allow room for the new brake pad.]

3. To complete the replacement, grasp the new brake pad by the tab (described in Step 1) and slide it into the caliper until it is fully inserted.

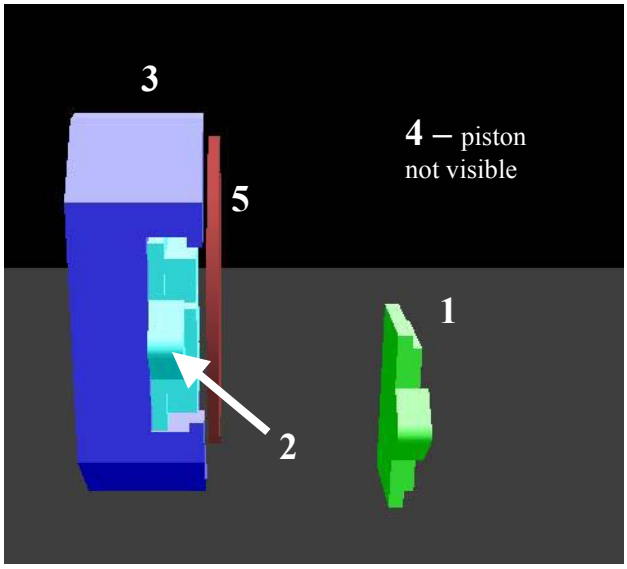
Your task is completed when the front edge of the new brake pad is flush with the front edge of the caliper.

The final configuration is shown in the second figure. Note that the caliper and the rotor do not move during the task.

A demonstration will be given. If you have any questions, please ask.



Initial Configuration



Final Configuration

A.7 HIDRA EXPERIMENTAL PARTICIPANT QUESTIONNAIRE

INSTRUCTIONS: Please answer each of the following questions to the best of your ability. For questions below, please mark your answer on the scale of 1-4 given.

1. How would you rate your knowledge and experience with computers?

Novice	($_1$)	($_2$)	($_3$)	($_4$)	Expert

2. How would you rate your knowledge and experience with virtual reality environments?

Novice	($_1$)	($_2$)	($_3$)	($_4$)	Expert

3. Have you ever used haptic technology or haptic devices? Yes ( ) No ( )

   If so, please describe: _____

4. How would you rate your knowledge with haptic technology and experience in using haptic devices?

Novice	($_1$)	($_2$)	($_3$)	($_4$)	Expert

5. When compared to your experiences manipulating objects in real life, how *similar* were manipulations of virtual objects through haptic feedback?

Not Similar	($_1$)	($_2$)	($_3$)	($_4$)	Very Similar

6. When compared to your experiences manipulating objects in real life, how *easy* were manipulations of virtual objects through haptic feedback?

Not Easy	($_1$)	($_2$)	($_3$)	($_4$)	Very Easy

7. How realistic do you feel the interactions between virtual objects were (i.e., Virtual Object #1 interacting with Virtual Object #2)?

Not Realistic	( ) 1	( ) 2	( ) 3	( ) 4	Very Realistic

8. Have you ever used 3D shutter glasses? Yes ( ) No ( )

   If so, please describe: _____

9. How much of an improvement did the 3D shutter glasses provide over 2D monitor viewing for interacting with the virtual environment?

No Improvement	( ) 1	( ) 2	( ) 3	( ) 4	Much Improvement

10. How much of an improvement did haptic feedback provide over no haptic feedback for interacting with the virtual environment?

No Improvement	( ) 1	( ) 2	( ) 3	( ) 4	Much Improvement

In the space below, please provide any additional comments that you would like to make about your experiences. Comments may include, but are not limited to, ease of use of the haptic interfaces or the simulation, realistic quality of the simulation, usefulness of haptic feedback and 3D shutter glasses, etc.

Thank you for your comments, your time, and your participation in this study!

APPENDIX B

DATA LISTING FOR EXPERIMENTAL STUDIES

The tables contained in this appendix contain the raw data collected during each experiment. A listing of each table with a brief description is provided below.

- **Appendix B.1:** Data for Weight Sensation Experiments
- **Appendix B.2:** Data for Motion Tolerance Experiment
- **Appendix B.3:** Data for Peg-in-Hole Insertion Experiment
- **Appendix B.4:** Data for Train Track Assembly Experiment
- **Appendix B.5:** Data for Brake Pad Replacement Experiment
- **Appendix B.6:** Responses to Experimental Participant Questionnaire

B.1 DATA FOR WEIGHT SENSATION EXPERIMENTS

The raw data for the weight sensation experiments is provided in Tables B.1 (Experiment 1), B.2 (Experiment 2), B.3 (Experiment 2, Extended), and B.4 (Experiment 3). A description of the column headers is provided below.

- *Set* – One set corresponds to the sequence through all cube weight variations. Each experimental participant completes one set.
- *Cube Order* – The order in which the cubes are presented to the participant.
- *Cube Weight* – Weight of the test cube in grams.
- *Virtual Gravity* – The value of the virtual gravity constant in the simulation (Experiment 3 only).
- *Correct Response* – The correct response for the weight judgment.
- *Subject Response* – The actual response of the participant.
- *Response Time* – The time required by the participant to make judgment in seconds.
- *Hand* – D: dominant hand used for virtual environment, ND: non-dominant hand used for virtual environment (Experiment 3 only).

Table B.1: Raw Data for Weight Sensation Study – Experiment 1

Set	Cube Order	Cube Weight	Correct Response	Subject Response	Response Time
1	1	112	Greater Than	Greater Than	9.815
1	2	104	Greater Than	Greater Than	4.216
1	3	100	Equal	Greater Than	3.525
1	4	116	Greater Than	Greater Than	3.696
1	5	80	Less Than	Equal	9.473
1	6	120	Greater Than	Greater Than	4.847
1	7	88	Less Than	Less Than	5.998
1	8	92	Less Than	Less Than	8.632
1	9	84	Less Than	Equal	16.203
1	10	96	Less Than	Less Than	26.989
1	11	108	Greater Than	Greater Than	14.711
2	1	92	Less Than	Equal	12.678
2	2	120	Greater Than	Greater Than	8.342
2	3	96	Less Than	Less Than	10.244
2	4	84	Less Than	Less Than	5.327
2	5	112	Greater Than	Equal	18.917
2	6	80	Less Than	Less Than	2.544
2	7	88	Less Than	Less Than	12.698
2	8	108	Greater Than	Less Than	15.792
2	9	100	Equal	Greater Than	17.165
2	10	116	Greater Than	Greater Than	8.362
2	11	104	Greater Than	Equal	12.828
3	1	108	Greater Than	Equal	10.926
3	2	88	Less Than	Less Than	9.433
3	3	116	Greater Than	Greater Than	12.528
3	4	92	Less Than	Less Than	16.233
3	5	104	Greater Than	Greater Than	11.307
3	6	120	Greater Than	Greater Than	13.169
3	7	112	Greater Than	Equal	10.896
3	8	96	Less Than	Equal	10.265
3	9	100	Equal	Less Than	10.315
3	10	84	Less Than	Less Than	10.745
3	11	80	Less Than	Less Than	8.752
4	1	88	Less Than	Less Than	14.484
4	2	96	Less Than	Greater Than	12.859
4	3	80	Less Than	Less Than	8.359
4	4	104	Greater Than	Greater Than	6.922
4	5	112	Greater Than	Greater Than	8.797
4	6	108	Greater Than	Greater Than	12.36
4	7	84	Less Than	Less Than	7.594
4	8	92	Less Than	Greater Than	14.25
4	9	100	Equal	Equal	27.484
4	10	116	Greater Than	Greater Than	21.391
4	11	120	Greater Than	Greater Than	18.656
5	1	80	Less Than	Less Than	5.016
5	2	112	Greater Than	Greater Than	5.438
5	3	100	Equal	Equal	7.516
5	4	104	Greater Than	Equal	10.922
5	5	88	Less Than	Equal	12.765
5	6	92	Less Than	Equal	17.75
5	7	116	Greater Than	Greater Than	8.359
5	8	96	Less Than	Less Than	11.563
5	9	84	Less Than	Less Than	15.484
5	10	108	Greater Than	Greater Than	5.547
5	11	120	Greater Than	Greater Than	14.734

Table B.1 (continued)

Set	Cube Order	Cube Weight	Correct Response	Subject Response	Response Time
6	1	120	Greater Than	Greater Than	11.344
6	2	88	Less Than	Equal	18.813
6	3	108	Greater Than	Greater Than	11.031
6	4	96	Less Than	Equal	17.953
6	5	92	Less Than	Equal	44.359
6	6	84	Less Than	Less Than	30.266
6	7	80	Less Than	Less Than	11.594
6	8	100	Equal	Greater Than	19.828
6	9	116	Greater Than	Greater Than	3.703
6	10	112	Greater Than	Greater Than	23.765
6	11	104	Greater Than	Equal	19.515
7	1	108	Greater Than	Greater Than	9.172
7	2	84	Less Than	Equal	19.265
7	3	96	Less Than	Equal	17.422
7	4	88	Less Than	Less Than	11.359
7	5	80	Less Than	Less Than	6.485
7	6	100	Equal	Greater Than	14.812
7	7	120	Greater Than	Greater Than	8.735
7	8	104	Greater Than	Greater Than	10.218
7	9	92	Less Than	Less Than	5.813
7	10	112	Greater Than	Greater Than	14.032
7	11	116	Greater Than	Equal	15.922
8	1	108	Greater Than	Greater Than	15.5
8	2	116	Greater Than	Greater Than	2.594
8	3	80	Less Than	Less Than	3.765
8	4	96	Less Than	Less Than	10.031
8	5	100	Equal	Less Than	9.406
8	6	120	Greater Than	Greater Than	5.969
8	7	104	Greater Than	Greater Than	28.25
8	8	112	Greater Than	Greater Than	18.141
8	9	84	Less Than	Equal	4.658
8	10	88	Less Than	Equal	6.813
8	11	92	Less Than	Less Than	4.594
9	1	116	Greater Than	Greater Than	38.125
9	2	84	Less Than	Less Than	14.094
9	3	96	Less Than	Equal	22.86
9	4	112	Greater Than	Greater Than	5.875
9	5	100	Equal	Equal	22.453
9	6	120	Greater Than	Greater Than	12.125
9	7	88	Less Than	Less Than	8.875
9	8	108	Greater Than	Greater Than	17.703
9	9	92	Less Than	Less Than	5.281
9	10	104	Greater Than	Less Than	14.593
9	11	80	Less Than	Less Than	5.672
10	1	88	Less Than	Equal	37.203
10	2	108	Greater Than	Greater Than	21.734
10	3	80	Less Than	Less Than	7.828
10	4	112	Greater Than	Greater Than	10.047
10	5	116	Greater Than	Greater Than	15.547
10	6	120	Greater Than	Greater Than	4.843
10	7	104	Greater Than	Equal	5.047
10	8	100	Equal	Less Than	3.719
10	9	84	Less Than	Less Than	5.984
10	10	96	Less Than	Equal	7.641
10	11	92	Less Than	Less Than	8.094

Table B.1 (continued)

Set	Cube Order	Cube Weight	Correct Response	Subject Response	Response Time
11	1	88	Less Than	Equal	8.953
11	2	120	Greater Than	Greater Than	3.953
11	3	84	Less Than	Equal	8.375
11	4	80	Less Than	Less Than	5.953
11	5	112	Greater Than	Greater Than	6.078
11	6	108	Greater Than	Greater Than	6.218
11	7	116	Greater Than	Greater Than	4.954
11	8	104	Greater Than	Greater Than	15.125
11	9	100	Equal	Equal	9.343
11	10	92	Less Than	Less Than	24.109
11	11	96	Less Than	Equal	8.875
12	1	112	Greater Than	Greater Than	13.812
12	2	92	Less Than	Equal	15.703
12	3	120	Greater Than	Greater Than	5.484
12	4	96	Less Than	Less Than	4.828
12	5	80	Less Than	Less Than	4.859
12	6	108	Greater Than	Equal	11.187
12	7	100	Equal	Equal	6.828
12	8	84	Less Than	Less Than	6.25
12	9	116	Greater Than	Greater Than	8.14
12	10	88	Less Than	Less Than	8.937
12	11	104	Greater Than	Equal	10.719
13	1	96	Less Than	Less Than	8.656
13	2	80	Less Than	Less Than	18.016
13	3	88	Less Than	Equal	10.703
13	4	112	Greater Than	Greater Than	5.188
13	5	116	Greater Than	Greater Than	8.609
13	6	100	Equal	Equal	21.891
13	7	84	Less Than	Less Than	9.61
13	8	92	Less Than	Equal	19.969
13	9	120	Greater Than	Greater Than	5.031
13	10	108	Greater Than	Greater Than	11.406
13	11	104	Greater Than	Greater Than	11.454
14	1	100	Equal	Greater Than	41.25
14	2	84	Less Than	Less Than	12.937
14	3	96	Less Than	Equal	15.109
14	4	88	Less Than	Equal	17.485
14	5	92	Less Than	Less Than	10.297
14	6	108	Greater Than	Greater Than	7.375
14	7	104	Greater Than	Greater Than	6.782
14	8	116	Greater Than	Greater Than	10.078
14	9	120	Greater Than	Greater Than	6.875
14	10	80	Less Than	Less Than	8.781
14	11	112	Greater Than	Greater Than	21.234

Table B.2: Raw Data for Weight Sensation Study – Experiment 2

Set	Cube Order	Cube Weight	Correct Response	Subject Response	Response Time
1	1	100	Equal	Less Than	22.14
1	2	80	Less Than	Less Than	9.578
1	3	96	Less Than	Less Than	9.875
1	4	120	Greater Than	Equal	14.937
1	5	116	Greater Than	Greater Than	9.797
1	6	88	Less Than	Less Than	13.047
1	7	108	Greater Than	Less Than	20.796
1	8	104	Greater Than	Equal	15.484
1	9	84	Less Than	Greater Than	13.766
1	10	92	Less Than	Greater Than	13.047
1	11	112	Greater Than	Greater Than	11
2	1	112	Greater Than	Less Than	14.781
2	2	116	Greater Than	Greater Than	6.859
2	3	92	Less Than	Equal	10.406
2	4	84	Less Than	Less Than	7.781
2	5	80	Less Than	Less Than	4.031
2	6	100	Equal	Equal	23.062
2	7	120	Greater Than	Greater Than	9.281
2	8	108	Greater Than	Less Than	6.578
2	9	104	Greater Than	Greater Than	23.218
2	10	88	Less Than	Greater Than	9.657
2	11	96	Less Than	Less Than	4.532
3	1	88	Less Than	Greater Than	13.485
3	2	80	Less Than	Equal	26.703
3	3	104	Greater Than	Greater Than	14
3	4	96	Less Than	Greater Than	13.937
3	5	120	Greater Than	Greater Than	15.14
3	6	108	Greater Than	Greater Than	16.078
3	7	84	Less Than	Equal	16.547
3	8	100	Equal	Equal	35.203
3	9	112	Greater Than	Greater Than	14.797
3	10	92	Less Than	Equal	16.531
3	11	116	Greater Than	Greater Than	17.062
4	1	92	Less Than	Greater Than	14.969
4	2	108	Greater Than	Greater Than	23.859
4	3	84	Less Than	Less Than	33.735
4	4	100	Equal	Less Than	10.094
4	5	88	Less Than	Equal	28.203
4	6	116	Greater Than	Greater Than	28.188
4	7	112	Greater Than	Greater Than	11.687
4	8	120	Greater Than	Less Than	19.125
4	9	104	Greater Than	Less Than	31.547
4	10	80	Less Than	Equal	26.641
4	11	96	Less Than	Less Than	39.468
5	1	96	Less Than	Less Than	6.234
5	2	88	Less Than	Less Than	9.969
5	3	120	Greater Than	Greater Than	18.703
5	4	100	Equal	Less Than	12.171
5	5	92	Less Than	Equal	18.203
5	6	104	Greater Than	Equal	17.343
5	7	108	Greater Than	Equal	22.672
5	8	80	Less Than	Less Than	22.578
5	9	116	Greater Than	Equal	17.547
5	10	112	Greater Than	Less Than	12.625
5	11	84	Less Than	Less Than	14.609

Table B.2 (continued)

Set	Cube Order	Cube Weight	Correct Response	Subject Response	Response Time
6	1	116	Greater Than	Equal	31.562
6	2	120	Greater Than	Less Than	24.672
6	3	92	Less Than	Equal	33.219
6	4	108	Greater Than	Greater Than	17.421
6	5	104	Greater Than	Less Than	7.719
6	6	88	Less Than	Less Than	7.266
6	7	80	Less Than	Equal	14.984
6	8	84	Less Than	Greater Than	9.968
6	9	112	Greater Than	Equal	17.218
6	10	96	Less Than	Equal	10.329
6	11	100	Equal	Greater Than	11.515
7	1	84	Less Than	Less Than	15.907
7	2	88	Less Than	Equal	11.296
7	3	108	Greater Than	Greater Than	12.703
7	4	104	Greater Than	Greater Than	12.64
7	5	112	Greater Than	Equal	17.515
7	6	80	Less Than	Less Than	12.422
7	7	96	Less Than	Equal	9.032
7	8	120	Greater Than	Greater Than	5.687
7	9	116	Greater Than	Less Than	9.235
7	10	100	Equal	Greater Than	18.281
7	11	92	Less Than	Equal	15.094
8	1	100	Equal	Equal	53.437
8	2	92	Less Than	Less Than	10.704
8	3	112	Greater Than	Greater Than	7.813
8	4	116	Greater Than	Greater Than	13.985
8	5	80	Less Than	Less Than	5.625
8	6	96	Less Than	Equal	16.438
8	7	120	Greater Than	Greater Than	12.25
8	8	104	Greater Than	Greater Than	23.094
8	9	108	Greater Than	Greater Than	9.422
8	10	88	Less Than	Equal	9.89
8	11	84	Less Than	Less Than	15.766
9	1	120	Greater Than	Greater Than	21.297
9	2	84	Less Than	Equal	25.484
9	3	104	Greater Than	Equal	30.687
9	4	92	Less Than	Greater Than	16.672
9	5	108	Greater Than	Less Than	36.172
9	6	112	Greater Than	Greater Than	12.453
9	7	80	Less Than	Less Than	17.844
9	8	116	Greater Than	Equal	24.64
9	9	96	Less Than	Greater Than	20.141
9	10	100	Equal	Equal	23.312
9	11	88	Less Than	Greater Than	35.578
10	1	100	Equal	Equal	36.969
10	2	80	Less Than	Less Than	9.5
10	3	96	Less Than	Less Than	25.594
10	4	112	Greater Than	Less Than	8.25
10	5	92	Less Than	Equal	19.922
10	6	84	Less Than	Greater Than	10.843
10	7	104	Greater Than	Greater Than	10.531
10	8	120	Greater Than	Greater Than	8.563
10	9	88	Less Than	Less Than	29.39
10	10	116	Greater Than	Greater Than	8.297
10	11	108	Greater Than	Greater Than	8.437

Table B.2 (continued)

Set	Cube Order	Cube Weight	Correct Response	Subject Response	Response Time
11	1	96	Less Than	Less Than	39.735
11	2	104	Greater Than	Less Than	21.062
11	3	120	Greater Than	Greater Than	17.219
11	4	108	Greater Than	Greater Than	19.985
11	5	116	Greater Than	Greater Than	19.859
11	6	100	Equal	Less Than	14.922
11	7	112	Greater Than	Greater Than	18.609
11	8	88	Less Than	Less Than	5.86
11	9	92	Less Than	Equal	12.594
11	10	80	Less Than	Less Than	5.953
11	11	84	Less Than	Less Than	19.032
12	1	84	Less Than	Equal	31.031
12	2	92	Less Than	Greater Than	37.984
12	3	88	Less Than	Greater Than	19.421
12	4	96	Less Than	Greater Than	14.75
12	5	116	Greater Than	Less Than	34.625
12	6	108	Greater Than	Equal	22.985
12	7	104	Greater Than	Greater Than	25.344
12	8	100	Equal	Equal	20.093
12	9	80	Less Than	Greater Than	30.5
12	10	120	Greater Than	Equal	30.656
12	11	112	Greater Than	Equal	28.516
13	1	88	Less Than	Equal	33.938
13	2	112	Greater Than	Greater Than	7.547
13	3	100	Equal	Greater Than	16.563
13	4	116	Greater Than	Equal	23.938
13	5	108	Greater Than	Greater Than	24.813
13	6	96	Less Than	Greater Than	36.625
13	7	84	Less Than	Less Than	8.515
13	8	104	Greater Than	Greater Than	8.625
13	9	92	Less Than	Equal	34.063
13	10	80	Less Than	Equal	21.75
13	11	120	Greater Than	Greater Than	6.344
14	1	80	Less Than	Less Than	10.547
14	2	116	Greater Than	Greater Than	26.125
14	3	84	Less Than	Less Than	28.14
14	4	88	Less Than	Equal	46.671
14	5	112	Greater Than	Greater Than	15.125
14	6	108	Greater Than	Greater Than	12.375
14	7	120	Greater Than	Greater Than	5.344
14	8	100	Equal	Greater Than	8.937
14	9	92	Less Than	Equal	10.062
14	10	96	Less Than	Greater Than	23.031
14	11	104	Greater Than	Greater Than	33.141

Table B.3: Raw Data for Weight Sensation Study – Experiment 2, Extended

Set	Cube Order	Cube Weight	Correct Response	Subject Response	Response Time
1	1	130	Greater Than	Greater Than	37.204
1	2	55	Less Than	Less Than	9.406
1	3	60	Less Than	Less Than	6.375
1	4	125	Greater Than	Greater Than	30.766
1	5	145	Greater Than	Greater Than	16.266
1	6	140	Greater Than	Greater Than	9.297
1	7	65	Less Than	Less Than	18.047
1	8	70	Less Than	Equal	24.047
1	9	75	Less Than	Less Than	9.703
1	10	135	Greater Than	Greater Than	13.828
2	1	145	Greater Than	Greater Than	17.546
2	2	55	Less Than	Equal	13.109
2	3	130	Greater Than	Greater Than	11.328
2	4	140	Greater Than	Greater Than	15.765
2	5	70	Less Than	Equal	17.734
2	6	75	Less Than	Less Than	14.875
2	7	125	Greater Than	Equal	18.969
2	8	60	Less Than	Less Than	22.078
2	9	65	Less Than	Less Than	18.469
2	10	135	Greater Than	Equal	16.016
3	1	55	Less Than	Less Than	17.11
3	2	135	Greater Than	Greater Than	5.281
3	3	140	Greater Than	Greater Than	5.625
3	4	145	Greater Than	Greater Than	5.219
3	5	65	Less Than	Less Than	39.937
3	6	125	Greater Than	Greater Than	27.781
3	7	70	Less Than	Less Than	12.218
3	8	75	Less Than	Less Than	50.5
3	9	130	Greater Than	Greater Than	6.937
3	10	60	Less Than	Less Than	9.14
4	1	145	Greater Than	Greater Than	38.844
4	2	55	Less Than	Less Than	19.656
4	3	75	Less Than	Less Than	19.296
4	4	65	Less Than	Equal	9.844
4	5	60	Less Than	Less Than	26.703
4	6	125	Greater Than	Greater Than	11.453
4	7	135	Greater Than	Greater Than	20.313
4	8	140	Greater Than	Greater Than	6.437
4	9	130	Greater Than	Greater Than	14.844
4	10	70	Less Than	Equal	8.687
5	1	140	Greater Than	Equal	30.75
5	2	130	Greater Than	Equal	30.437
5	3	65	Less Than	Less Than	18.281
5	4	75	Less Than	Less Than	37.656
5	5	55	Less Than	Less Than	9.078
5	6	125	Greater Than	Equal	21.329
5	7	60	Less Than	Less Than	43.172
5	8	70	Less Than	Equal	33.563
5	9	135	Greater Than	Equal	29.093
5	10	145	Greater Than	Greater Than	17.968

Table B.3 (continued)

Set	Cube Order	Cube Weight	Correct Response	Subject Response	Response Time
6	1	65	Less Than	Less Than	12.687
6	2	145	Greater Than	Greater Than	8.453
6	3	135	Greater Than	Greater Than	9.609
6	4	70	Less Than	Less Than	10.328
6	5	75	Less Than	Equal	19.453
6	6	55	Less Than	Less Than	9.828
6	7	130	Greater Than	Greater Than	14.266
6	8	60	Less Than	Less Than	7.906
6	9	140	Greater Than	Greater Than	14.688
6	10	125	Greater Than	Greater Than	16.625
7	1	65	Less Than	Less Than	10.766
7	2	55	Less Than	Equal	23.75
7	3	135	Greater Than	Greater Than	30.5
7	4	75	Less Than	Greater Than	9.469
7	5	70	Less Than	Less Than	12.438
7	6	145	Greater Than	Greater Than	12.297
7	7	60	Less Than	Less Than	5.781
7	8	140	Greater Than	Greater Than	9.344
7	9	130	Greater Than	Greater Than	5.672
7	10	125	Greater Than	Greater Than	8.641
8	1	135	Greater Than	Greater Than	18.734
8	2	55	Less Than	Less Than	5.641
8	3	60	Less Than	Less Than	17.657
8	4	145	Greater Than	Greater Than	7.11
8	5	140	Greater Than	Greater Than	24.25
8	6	70	Less Than	Less Than	22.64
8	7	130	Greater Than	Greater Than	7.516
8	8	65	Less Than	Less Than	14.312
8	9	125	Greater Than	Equal	16.078
8	10	75	Less Than	Less Than	12.453
9	1	55	Less Than	Less Than	7.094
9	2	70	Less Than	Less Than	6.234
9	3	65	Less Than	Less Than	15.031
9	4	130	Greater Than	Less Than	9.641
9	5	60	Less Than	Less Than	7.235
9	6	135	Greater Than	Greater Than	13.672
9	7	125	Greater Than	Less Than	8.875
9	8	75	Less Than	Less Than	6.125
9	9	140	Greater Than	Greater Than	10.891
9	10	145	Greater Than	Greater Than	10.844
10	1	125	Greater Than	Greater Than	8.156
10	2	130	Greater Than	Greater Than	6.344
10	3	75	Less Than	Greater Than	17.172
10	4	70	Less Than	Greater Than	6.563
10	5	65	Less Than	Less Than	6.234
10	6	60	Less Than	Less Than	6.156
10	7	55	Less Than	Equal	8.359
10	8	135	Greater Than	Greater Than	7.625
10	9	140	Greater Than	Greater Than	6.172
10	10	145	Greater Than	Greater Than	7.187

Table B.3 (continued)

Set	Cube Order	Cube Weight	Correct Response	Subject Response	Response Time
11	1	60	Less Than	Less Than	6.593
11	2	145	Greater Than	Greater Than	6.375
11	3	55	Less Than	Equal	5.046
11	4	65	Less Than	Less Than	11.579
11	5	130	Greater Than	Greater Than	5.75
11	6	140	Greater Than	Greater Than	4.609
11	7	135	Greater Than	Greater Than	9.922
11	8	75	Less Than	Less Than	8.907
11	9	125	Greater Than	Greater Than	4.031
11	10	70	Less Than	Equal	11.813
12	1	60	Less Than	Less Than	5.734
12	2	130	Greater Than	Less Than	5.735
12	3	70	Less Than	Less Than	7.719
12	4	55	Less Than	Less Than	6.312
12	5	125	Greater Than	Greater Than	7.375
12	6	65	Less Than	Less Than	5.156
12	7	75	Less Than	Less Than	11.641
12	8	145	Greater Than	Greater Than	16.797
12	9	140	Greater Than	Greater Than	3.594
12	10	135	Greater Than	Greater Than	7.813
13	1	70	Less Than	Less Than	20.781
13	2	145	Greater Than	Greater Than	6.172
13	3	60	Less Than	Less Than	15.421
13	4	140	Greater Than	Greater Than	6.625
13	5	125	Greater Than	Greater Than	7.531
13	6	130	Greater Than	Greater Than	20.813
13	7	75	Less Than	Less Than	15.578
13	8	65	Less Than	Less Than	5.985
13	9	135	Greater Than	Greater Than	9.734
13	10	55	Less Than	Less Than	7.297
14	1	145	Greater Than	Greater Than	5.5
14	2	60	Less Than	Less Than	8.922
14	3	140	Greater Than	Greater Than	6.094
14	4	130	Greater Than	Greater Than	10.984
14	5	125	Greater Than	Greater Than	14.344
14	6	75	Less Than	Less Than	5.453
14	7	70	Less Than	Less Than	9.547
14	8	135	Greater Than	Greater Than	6.797
14	9	65	Less Than	Less Than	5.859
14	10	55	Less Than	Less Than	4.453

Table B.4: Raw Data for Weight Sensation Study – Experiment 3

Set	Cube Order	Virtual Gravity	Correct Response	Subject Response	Response Time	Hand
1	1	7.3	Less Than	Greater Than	21.821	D
1	2	9.3	Less Than	Greater Than	7.731	D
1	3	11.3	Greater Than	Greater Than	21.281	D
1	4	6.8	Less Than	Greater Than	6.679	D
1	5	10.8	Greater Than	Greater Than	8.613	D
1	6	11.8	Greater Than	Greater Than	6.048	D
1	7	9.8	Equal	Greater Than	13.009	D
1	8	10.3	Greater Than	Greater Than	11.496	D
1	9	7.8	Less Than	Greater Than	18.526	D
1	10	8.8	Less Than	Greater Than	18.156	D
1	11	12.3	Greater Than	Greater Than	10.605	D
1	12	6.3	Less Than	Greater Than	19.978	D
1	13	13.3	Greater Than	Greater Than	10.465	D
1	14	12.8	Greater Than	Greater Than	15.352	D
1	15	8.3	Less Than	Equal	18.697	D
2	1	11.8	Greater Than	Greater Than	20.749	D
2	2	6.8	Less Than	Less Than	40.298	D
2	3	7.3	Less Than	Less Than	15.732	D
2	4	10.3	Greater Than	Greater Than	53.887	D
2	5	6.3	Less Than	Less Than	18.457	D
2	6	9.3	Less Than	Greater Than	10.685	D
2	7	12.8	Greater Than	Greater Than	11.266	D
2	8	8.8	Less Than	Equal	13.86	D
2	9	7.8	Less Than	Equal	18.126	D
2	10	9.8	Equal	Greater Than	34.389	D
2	11	11.3	Greater Than	Greater Than	7.982	D
2	12	10.8	Greater Than	Equal	19.277	D
2	13	12.3	Greater Than	Less Than	7.111	D
2	14	13.3	Greater Than	Greater Than	29.192	D
2	15	8.3	Less Than	Greater Than	9.784	D
3	1	8.3	Less Than	Greater Than	14.641	D
3	2	11.8	Greater Than	Greater Than	13.619	D
3	3	9.3	Less Than	Greater Than	17.385	D
3	4	6.3	Less Than	Equal	26.087	D
3	5	12.3	Greater Than	Greater Than	22.072	D
3	6	10.8	Greater Than	Equal	20.84	D
3	7	9.8	Equal	Equal	18.777	D
3	8	12.8	Greater Than	Equal	31.535	D
3	9	11.3	Greater Than	Less Than	19.418	D
3	10	13.3	Greater Than	Greater Than	22.322	D
3	11	7.8	Less Than	Less Than	18.277	D
3	12	7.3	Less Than	Less Than	45.035	D
3	13	6.8	Less Than	Equal	23.824	D
3	14	10.3	Greater Than	Less Than	27.66	D
3	15	8.8	Less Than	Greater Than	18.927	D
4	1	7.3	Less Than	Greater Than	65.422	D
4	2	8.8	Less Than	Greater Than	19.688	D
4	3	7.8	Less Than	Less Than	41.078	D
4	4	11.3	Greater Than	Greater Than	29.328	D
4	5	10.8	Greater Than	Greater Than	25.375	D
4	6	6.3	Less Than	Less Than	21.828	D
4	7	9.8	Equal	Greater Than	33.828	D
4	8	6.8	Less Than	Equal	45.391	D
4	9	8.3	Less Than	Greater Than	6.515	D
4	10	12.3	Greater Than	Greater Than	32.531	D
4	11	12.8	Greater Than	Greater Than	44.188	D
4	12	11.8	Greater Than	Greater Than	26.094	D
4	13	13.3	Greater Than	Greater Than	9.125	D
4	14	9.3	Less Than	Equal	48.297	D
4	15	10.3	Greater Than	Greater Than	12.968	D
5	1	12.8	Greater Than	Greater Than	13.234	D
5	2	9.3	Less Than	Greater Than	30	D
5	3	12.3	Greater Than	Greater Than	14.031	D
5	4	10.8	Greater Than	Greater Than	29.907	D
5	5	7.3	Less Than	Equal	13.094	D
5	6	10.3	Greater Than	Greater Than	19.063	D
5	7	6.8	Less Than	Equal	15.11	D
5	8	11.3	Greater Than	Greater Than	13.734	D
5	9	13.3	Greater Than	Greater Than	17.016	D
5	10	11.8	Greater Than	Greater Than	45.219	D
5	11	9.8	Equal	Equal	24.36	D
5	12	6.3	Less Than	Less Than	8.421	D
5	13	8.3	Less Than	Less Than	37.984	D

Table B.4 (continued)

Set	Cube Order	Virtual Gravity	Correct Response	Subject Response	Response Time	Hand
6	1	12.8	Greater Than	Greater Than	31.25	D
6	2	12.3	Greater Than	Greater Than	70.407	D
6	3	8.8	Less Than	Equal	13.297	D
6	4	6.8	Less Than	Equal	40.422	D
6	5	8.3	Less Than	Equal	47.25	D
6	6	10.3	Greater Than	Greater Than	13.391	D
6	7	7.3	Less Than	Less Than	36.343	D
6	8	11.3	Greater Than	Equal	23.969	D
6	9	9.3	Less Than	Greater Than	12.062	D
6	10	10.8	Greater Than	Equal	14.766	D
6	11	9.8	Equal	Less Than	13.954	D
6	12	13.3	Greater Than	Greater Than	6.766	D
6	13	6.3	Less Than	Less Than	4.594	D
6	14	11.8	Greater Than	Less Than	11.812	D
6	15	7.8	Less Than	Less Than	6.531	D
7	1	11.8	Greater Than	Greater Than	10.172	D
7	2	7.3	Less Than	Greater Than	20.719	D
7	3	6.8	Less Than	Equal	19.093	D
7	4	8.8	Less Than	Greater Than	13.906	D
7	5	12.8	Greater Than	Greater Than	5.218	D
7	6	12.3	Greater Than	Greater Than	9.422	D
7	7	6.3	Less Than	Greater Than	17.641	D
7	8	11.3	Greater Than	Greater Than	9.641	D
7	9	7.8	Less Than	Greater Than	17.532	D
7	10	10.8	Greater Than	Greater Than	10.484	D
7	11	8.3	Less Than	Greater Than	18.031	D
7	12	9.8	Equal	Equal	13.922	D
7	13	9.3	Less Than	Equal	14.641	D
7	14	10.3	Greater Than	Greater Than	7.813	D
7	15	13.3	Greater Than	Greater Than	5.375	D
8	1	6.3	Less Than	Equal	26.609	ND
8	2	8.3	Less Than	Greater Than	6.313	ND
8	3	7.3	Less Than	Less Than	28.984	ND
8	4	12.8	Greater Than	Greater Than	4.656	ND
8	5	11.3	Greater Than	Greater Than	13.984	ND
8	6	10.3	Greater Than	Equal	9.515	ND
8	7	6.8	Less Than	Less Than	11.594	ND
8	8	9.8	Equal	Equal	10.594	ND
8	9	8.8	Less Than	Equal	16.609	ND
8	10	9.3	Less Than	Less Than	14.344	ND
8	11	10.8	Greater Than	Greater Than	7.766	ND
8	12	13.3	Greater Than	Greater Than	15.453	ND
8	13	7.8	Less Than	Less Than	4.563	ND
8	14	12.3	Greater Than	Greater Than	27.828	ND
8	15	11.8	Greater Than	Equal	9.969	ND
9	1	11.8	Greater Than	Greater Than	30.407	ND
9	2	8.3	Less Than	Greater Than	26.922	ND
9	3	9.8	Equal	Greater Than	34.265	ND
9	4	7.8	Less Than	Less Than	8.985	ND
9	5	10.8	Greater Than	Greater Than	8.969	ND
9	6	12.3	Greater Than	Greater Than	10.89	ND
9	7	11.3	Greater Than	Equal	36.079	ND
9	8	6.3	Less Than	Less Than	10.453	ND
9	9	9.3	Less Than	Less Than	28.797	ND
9	10	7.3	Less Than	Less Than	21.61	ND
9	11	10.3	Greater Than	Greater Than	12.735	ND
9	12	13.3	Greater Than	Greater Than	10.485	ND
9	13	8.8	Less Than	Less Than	10.344	ND
9	14	12.8	Greater Than	Greater Than	9.937	ND
9	15	6.8	Less Than	Less Than	17.75	ND
10	1	7.8	Less Than	Greater Than	37.203	ND
10	2	11.8	Greater Than	Equal	45.297	ND
10	3	8.8	Less Than	Less Than	14.797	ND
10	4	7.3	Less Than	Less Than	16.75	ND
10	5	10.8	Greater Than	Less Than	31.844	ND
10	6	10.3	Greater Than	Equal	24.89	ND
10	7	8.3	Less Than	Less Than	18.36	ND
10	8	9.8	Equal	Greater Than	22.656	ND
10	9	12.8	Greater Than	Greater Than	10.516	ND
10	10	12.3	Greater Than	Greater Than	12.938	ND
10	11	11.3	Greater Than	Greater Than	15.484	ND
10	12	6.8	Less Than	Equal	28.797	ND
10	13	6.3	Less Than	Equal	39.157	ND

Table B.4 (continued)

Set	Cube Order	Virtual Gravity	Correct Response	Subject Response	Response Time	Hand
11	1	7.3	Less Than	Less Than	33.546	ND
11	2	10.3	Greater Than	Greater Than	6.078	ND
11	3	6.3	Less Than	Equal	26.047	ND
11	4	12.8	Greater Than	Greater Than	5.859	ND
11	5	10.8	Greater Than	Equal	16.39	ND
11	6	6.8	Less Than	Less Than	5.36	ND
11	7	13.3	Greater Than	Greater Than	15.328	ND
11	8	8.3	Less Than	Equal	13.922	ND
11	9	9.8	Equal	Less Than	13.328	ND
11	10	12.3	Greater Than	Greater Than	14.156	ND
11	11	8.8	Less Than	Equal	7.266	ND
11	12	9.3	Less Than	Greater Than	14.156	ND
11	13	11.3	Greater Than	Greater Than	9.969	ND
11	14	7.8	Less Than	Equal	17.218	ND
11	15	11.8	Greater Than	Equal	24.312	ND
12	1	6.8	Less Than	Greater Than	13.265	ND
12	2	6.3	Less Than	Greater Than	19.922	ND
12	3	8.8	Less Than	Greater Than	40.375	ND
12	4	10.8	Greater Than	Equal	20.734	ND
12	5	11.3	Greater Than	Greater Than	38.312	ND
12	6	12.8	Greater Than	Greater Than	41.219	ND
12	7	7.3	Less Than	Less Than	8.266	ND
12	8	9.8	Equal	Equal	17.766	ND
12	9	8.3	Less Than	Equal	12.594	ND
12	10	13.3	Greater Than	Greater Than	13	ND
12	11	7.8	Less Than	Equal	14.297	ND
12	12	10.3	Greater Than	Equal	25.813	ND
12	13	11.8	Greater Than	Greater Than	13.5	ND
12	14	9.3	Less Than	Equal	21.437	ND
12	15	12.3	Greater Than	Greater Than	15.828	ND
13	1	11.3	Greater Than	Greater Than	14.531	ND
13	2	7.3	Less Than	Equal	25.187	ND
13	3	12.3	Greater Than	Greater Than	7.734	ND
13	4	13.3	Greater Than	Greater Than	12.719	ND
13	5	12.8	Greater Than	Equal	31.985	ND
13	6	8.3	Less Than	Greater Than	10.438	ND
13	7	6.8	Less Than	Less Than	5.453	ND
13	8	9.8	Equal	Greater Than	5.094	ND
13	9	10.3	Greater Than	Greater Than	9.313	ND
13	10	9.3	Less Than	Greater Than	15.438	ND
13	11	6.3	Less Than	Equal	19.172	ND
13	12	8.8	Less Than	Greater Than	25.218	ND
13	13	7.8	Less Than	Greater Than	10.171	ND
13	14	11.8	Greater Than	Greater Than	12.718	ND
13	15	10.8	Greater Than	Greater Than	15.235	ND
14	1	12.3	Greater Than	Greater Than	11.968	ND
14	2	9.3	Less Than	Equal	31.047	ND
14	3	6.3	Less Than	Less Than	10.844	ND
14	4	7.8	Less Than	Equal	27.266	ND
14	5	11.3	Greater Than	Equal	41.015	ND
14	6	12.8	Greater Than	Greater Than	9.328	ND
14	7	13.3	Greater Than	Greater Than	18.782	ND
14	8	9.8	Equal	Less Than	7.984	ND
14	9	6.8	Less Than	Less Than	4.422	ND
14	10	8.8	Less Than	Less Than	14.89	ND
14	11	7.3	Less Than	Less Than	8.765	ND
14	12	8.3	Less Than	Equal	11.047	ND
14	13	10.8	Greater Than	Equal	38.047	ND
14	14	10.3	Greater Than	Less Than	12.282	ND
14	15	11.8	Greater Than	Greater Than	14.094	ND

B.2 DATA FOR MOTION TOLERANCE EXPERIMENT

The raw data for the motion tolerance experiment is provided in Table B.5. A description of the column headers is provided below.

- *Set* – One set corresponds to the sequence through all hole size variations for all four combinations of *2D or 3D?* and *Hapics?*.
- *Presentation Order* – The order in which the different holes are presented to the participant.
- *2D or 3D?* – 2D monitor visualization or 3D visualization with shutter glasses.
- *Hapics?* – Whether haptic feedback is active or not.
- *Tol. Diff. (x-z)* – The difference in allowable range of motion of the peg along the x-axis and the z-axis in millimeters (See Section 5.3.2).
- *Correct Response* – The correct response for the tolerance judgment. Either motion along the x-axis is greater (LeftToRightGreater) or motion along the z-axis is greater (FrontToBackGreater).
- *Subject Response* – The actual response of the participant.
- *Response Time* – The time required by the participant to make judgment in seconds.
- *Env. Sequence* – The environment sequence is defined as the order in which a given combination of *2D or 3D?* and *Haptics?* was presented to the user.

Table B.5: Raw Data for Motion Tolerance Experiment

Set	Presentation Order	2D or 3D?	Haptics?	Tol. Diff. (x - z)	Correct Response	Subject Response	Response Time	Env. Sequence
1	1	2D	N	3	LeftToRightGreater	LeftToRightGreater	20.031	1
1	2	2D	N	-3	FrontToBackGreater	LeftToRightGreater	27.375	1
1	3	2D	N	1	LeftToRightGreater	LeftToRightGreater	8.906	1
1	4	2D	N	5	LeftToRightGreater	LeftToRightGreater	4.735	1
1	5	2D	N	4	LeftToRightGreater	LeftToRightGreater	10.703	1
1	6	2D	N	2	LeftToRightGreater	LeftToRightGreater	5.313	1
1	7	2D	N	-5	FrontToBackGreater	FrontToBackGreater	5.438	1
1	8	2D	N	-4	FrontToBackGreater	FrontToBackGreater	19.438	1
1	9	2D	N	-2	FrontToBackGreater	LeftToRightGreater	30.109	1
1	10	2D	N	-1	FrontToBackGreater	LeftToRightGreater	15.703	1
2	1	2D	N	-1	FrontToBackGreater	LeftToRightGreater	8.031	2
2	2	2D	N	5	LeftToRightGreater	LeftToRightGreater	5.016	2
2	3	2D	N	-5	FrontToBackGreater	FrontToBackGreater	6.578	2
2	4	2D	N	3	LeftToRightGreater	LeftToRightGreater	4.938	2
2	5	2D	N	-4	FrontToBackGreater	FrontToBackGreater	7.063	2
2	6	2D	N	2	LeftToRightGreater	LeftToRightGreater	9.406	2
2	7	2D	N	1	LeftToRightGreater	LeftToRightGreater	3.343	2
2	8	2D	N	-2	FrontToBackGreater	FrontToBackGreater	21.718	2
2	9	2D	N	4	LeftToRightGreater	LeftToRightGreater	5.125	2
2	10	2D	N	-3	FrontToBackGreater	FrontToBackGreater	7.297	2
3	1	2D	N	-3	FrontToBackGreater	FrontToBackGreater	7.578	3
3	2	2D	N	1	LeftToRightGreater	LeftToRightGreater	4.89	3
3	3	2D	N	4	LeftToRightGreater	LeftToRightGreater	4.657	3
3	4	2D	N	-1	FrontToBackGreater	FrontToBackGreater	10.203	3
3	5	2D	N	-2	FrontToBackGreater	FrontToBackGreater	7.656	3
3	6	2D	N	2	LeftToRightGreater	LeftToRightGreater	6.359	3
3	7	2D	N	-5	FrontToBackGreater	FrontToBackGreater	8.5	3
3	8	2D	N	-4	FrontToBackGreater	FrontToBackGreater	8.032	3
3	9	2D	N	3	LeftToRightGreater	LeftToRightGreater	4.531	3
3	10	2D	N	5	LeftToRightGreater	LeftToRightGreater	7.469	3
4	1	2D	N	-4	FrontToBackGreater	FrontToBackGreater	4.516	4
4	2	2D	N	-3	FrontToBackGreater	FrontToBackGreater	6.64	4
4	3	2D	N	3	LeftToRightGreater	LeftToRightGreater	3.609	4
4	4	2D	N	1	LeftToRightGreater	FrontToBackGreater	9.125	4
4	5	2D	N	2	LeftToRightGreater	LeftToRightGreater	5.329	4
4	6	2D	N	5	LeftToRightGreater	LeftToRightGreater	3.328	4
4	7	2D	N	-2	FrontToBackGreater	FrontToBackGreater	13.219	4
4	8	2D	N	-5	FrontToBackGreater	FrontToBackGreater	4.328	4
4	9	2D	N	4	LeftToRightGreater	LeftToRightGreater	3.266	4
4	10	2D	N	-1	FrontToBackGreater	LeftToRightGreater	4.422	4
5	1	2D	N	2	LeftToRightGreater	LeftToRightGreater	11.656	1
5	2	2D	N	-5	FrontToBackGreater	FrontToBackGreater	16.562	1
5	3	2D	N	1	LeftToRightGreater	LeftToRightGreater	12.141	1
5	4	2D	N	3	LeftToRightGreater	LeftToRightGreater	14.531	1
5	5	2D	N	-4	FrontToBackGreater	FrontToBackGreater	17.657	1
5	6	2D	N	-2	FrontToBackGreater	LeftToRightGreater	13.735	1
5	7	2D	N	-1	FrontToBackGreater	LeftToRightGreater	7.953	1
5	8	2D	N	5	LeftToRightGreater	LeftToRightGreater	5.734	1
5	9	2D	N	-3	FrontToBackGreater	FrontToBackGreater	4.437	1
5	10	2D	N	4	LeftToRightGreater	LeftToRightGreater	7.406	1

Table B.5 (continued)

Set	Presentation Order	2D or 3D?	Haptics?	Tol. Diff. (x - z)	Correct Response	Subject Response	Response Time	Env. Sequence
6	1	2D	N	-4	FrontToBackGreater	FrontToBackGreater	19.157	2
6	2	2D	N	-2	FrontToBackGreater	LeftToRightGreater	6.469	2
6	3	2D	N	3	LeftToRightGreater	LeftToRightGreater	17.469	2
6	4	2D	N	5	LeftToRightGreater	LeftToRightGreater	9.14	2
6	5	2D	N	4	LeftToRightGreater	LeftToRightGreater	13.5	2
6	6	2D	N	2	LeftToRightGreater	LeftToRightGreater	9.484	2
6	7	2D	N	-1	FrontToBackGreater	LeftToRightGreater	7.344	2
6	8	2D	N	1	LeftToRightGreater	LeftToRightGreater	13.032	2
6	9	2D	N	-5	FrontToBackGreater	FrontToBackGreater	6.609	2
6	10	2D	N	-3	FrontToBackGreater	FrontToBackGreater	5.688	2
7	1	2D	N	1	LeftToRightGreater	LeftToRightGreater	5.485	3
7	2	2D	N	-5	FrontToBackGreater	FrontToBackGreater	13.094	3
7	3	2D	N	2	LeftToRightGreater	LeftToRightGreater	4.953	3
7	4	2D	N	4	LeftToRightGreater	LeftToRightGreater	8	3
7	5	2D	N	-2	FrontToBackGreater	FrontToBackGreater	6.891	3
7	6	2D	N	-1	FrontToBackGreater	LeftToRightGreater	10.078	3
7	7	2D	N	-3	FrontToBackGreater	LeftToRightGreater	11.766	3
7	8	2D	N	-4	FrontToBackGreater	FrontToBackGreater	14.875	3
7	9	2D	N	5	LeftToRightGreater	LeftToRightGreater	5.953	3
7	10	2D	N	3	LeftToRightGreater	LeftToRightGreater	5.344	3
8	1	2D	N	-2	FrontToBackGreater	FrontToBackGreater	8.907	4
8	2	2D	N	-4	FrontToBackGreater	FrontToBackGreater	12.812	4
8	3	2D	N	-5	FrontToBackGreater	FrontToBackGreater	7.406	4
8	4	2D	N	1	LeftToRightGreater	LeftToRightGreater	14.187	4
8	5	2D	N	2	LeftToRightGreater	LeftToRightGreater	7.516	4
8	6	2D	N	-1	FrontToBackGreater	FrontToBackGreater	17.703	4
8	7	2D	N	4	LeftToRightGreater	LeftToRightGreater	10.375	4
8	8	2D	N	3	LeftToRightGreater	LeftToRightGreater	8.109	4
8	9	2D	N	5	LeftToRightGreater	LeftToRightGreater	5.516	4
8	10	2D	N	-3	FrontToBackGreater	FrontToBackGreater	11.156	4
1	1	2D	Y	2	LeftToRightGreater	LeftToRightGreater	11.954	2
1	2	2D	Y	-2	FrontToBackGreater	FrontToBackGreater	9.344	2
1	3	2D	Y	-4	FrontToBackGreater	FrontToBackGreater	5.625	2
1	4	2D	Y	1	LeftToRightGreater	LeftToRightGreater	4.578	2
1	5	2D	Y	5	LeftToRightGreater	LeftToRightGreater	3.61	2
1	6	2D	Y	-5	FrontToBackGreater	FrontToBackGreater	7.703	2
1	7	2D	Y	4	LeftToRightGreater	LeftToRightGreater	3.625	2
1	8	2D	Y	3	LeftToRightGreater	LeftToRightGreater	4.86	2
1	9	2D	Y	-1	FrontToBackGreater	LeftToRightGreater	6.625	2
1	10	2D	Y	-3	FrontToBackGreater	FrontToBackGreater	7.828	2
2	1	2D	Y	-2	FrontToBackGreater	LeftToRightGreater	9.094	1
2	2	2D	Y	-3	FrontToBackGreater	FrontToBackGreater	28.594	1
2	3	2D	Y	3	LeftToRightGreater	LeftToRightGreater	4.079	1
2	4	2D	Y	2	LeftToRightGreater	LeftToRightGreater	5.562	1
2	5	2D	Y	1	LeftToRightGreater	LeftToRightGreater	4.375	1
2	6	2D	Y	5	LeftToRightGreater	LeftToRightGreater	2.61	1
2	7	2D	Y	-5	FrontToBackGreater	FrontToBackGreater	3.594	1
2	8	2D	Y	-4	FrontToBackGreater	FrontToBackGreater	4.188	1
2	9	2D	Y	4	LeftToRightGreater	LeftToRightGreater	5.828	1
2	10	2D	Y	-1	FrontToBackGreater	LeftToRightGreater	8.609	1

Table B.5 (continued)

Set	Presentation Order	2D or 3D?	Haptics?	Tol. Diff. (x - z)	Correct Response	Subject Response	Response Time	Env. Sequence
3	1	2D	Y	4	LeftToRightGreater	LeftToRightGreater	10.953	4
3	2	2D	Y	5	LeftToRightGreater	LeftToRightGreater	3.875	4
3	3	2D	Y	-1	FrontToBackGreater	LeftToRightGreater	4.828	4
3	4	2D	Y	2	LeftToRightGreater	LeftToRightGreater	4.109	4
3	5	2D	Y	-5	FrontToBackGreater	FrontToBackGreater	3.297	4
3	6	2D	Y	-4	FrontToBackGreater	FrontToBackGreater	5.719	4
3	7	2D	Y	-3	FrontToBackGreater	FrontToBackGreater	4.14	4
3	8	2D	Y	1	LeftToRightGreater	LeftToRightGreater	4.125	4
3	9	2D	Y	-2	FrontToBackGreater	FrontToBackGreater	6.813	4
3	10	2D	Y	3	LeftToRightGreater	LeftToRightGreater	3.375	4
4	1	2D	Y	-1	FrontToBackGreater	LeftToRightGreater	10.969	3
4	2	2D	Y	-5	FrontToBackGreater	FrontToBackGreater	6.422	3
4	3	2D	Y	-4	FrontToBackGreater	FrontToBackGreater	6.781	3
4	4	2D	Y	-3	FrontToBackGreater	FrontToBackGreater	8.719	3
4	5	2D	Y	2	LeftToRightGreater	LeftToRightGreater	4.421	3
4	6	2D	Y	1	LeftToRightGreater	LeftToRightGreater	6.469	3
4	7	2D	Y	3	LeftToRightGreater	LeftToRightGreater	4.078	3
4	8	2D	Y	5	LeftToRightGreater	LeftToRightGreater	2.844	3
4	9	2D	Y	-2	FrontToBackGreater	FrontToBackGreater	4.485	3
4	10	2D	Y	4	LeftToRightGreater	LeftToRightGreater	3.312	3
5	1	2D	Y	4	LeftToRightGreater	LeftToRightGreater	7.187	2
5	2	2D	Y	3	LeftToRightGreater	LeftToRightGreater	9.188	2
5	3	2D	Y	-3	FrontToBackGreater	FrontToBackGreater	6.469	2
5	4	2D	Y	-1	FrontToBackGreater	LeftToRightGreater	14.156	2
5	5	2D	Y	-4	FrontToBackGreater	FrontToBackGreater	8.469	2
5	6	2D	Y	-2	FrontToBackGreater	FrontToBackGreater	11.89	2
5	7	2D	Y	5	LeftToRightGreater	LeftToRightGreater	5.547	2
5	8	2D	Y	2	LeftToRightGreater	LeftToRightGreater	3	2
5	9	2D	Y	1	LeftToRightGreater	LeftToRightGreater	7.031	2
5	10	2D	Y	-5	FrontToBackGreater	FrontToBackGreater	7.688	2
6	1	2D	Y	4	LeftToRightGreater	LeftToRightGreater	58.437	1
6	2	2D	Y	5	LeftToRightGreater	LeftToRightGreater	10.547	1
6	3	2D	Y	3	LeftToRightGreater	LeftToRightGreater	25.797	1
6	4	2D	Y	1	LeftToRightGreater	LeftToRightGreater	27.625	1
6	5	2D	Y	-3	FrontToBackGreater	FrontToBackGreater	6.766	1
6	6	2D	Y	2	LeftToRightGreater	LeftToRightGreater	6.313	1
6	7	2D	Y	-5	FrontToBackGreater	FrontToBackGreater	3.813	1
6	8	2D	Y	-2	FrontToBackGreater	FrontToBackGreater	4.344	1
6	9	2D	Y	-1	FrontToBackGreater	FrontToBackGreater	10.906	1
6	10	2D	Y	-4	FrontToBackGreater	FrontToBackGreater	9.016	1
7	1	2D	Y	-5	FrontToBackGreater	FrontToBackGreater	5.36	4
7	2	2D	Y	3	LeftToRightGreater	LeftToRightGreater	2.781	4
7	3	2D	Y	-1	FrontToBackGreater	FrontToBackGreater	5.703	4
7	4	2D	Y	-4	FrontToBackGreater	FrontToBackGreater	4.985	4
7	5	2D	Y	-2	FrontToBackGreater	FrontToBackGreater	12.453	4
7	6	2D	Y	5	LeftToRightGreater	LeftToRightGreater	3.016	4
7	7	2D	Y	1	LeftToRightGreater	LeftToRightGreater	5.906	4
7	8	2D	Y	4	LeftToRightGreater	LeftToRightGreater	3.719	4
7	9	2D	Y	-3	FrontToBackGreater	FrontToBackGreater	5.171	4
7	10	2D	Y	2	LeftToRightGreater	LeftToRightGreater	4.891	4

Table B.5 (continued)

Set	Presentation Order	2D or 3D?	Haptics?	Tol. Diff. (x - z)	Correct Response	Subject Response	Response Time	Env. Sequence
8	1	2D	Y	-3	FrontToBackGreater	FrontToBackGreater	7.937	3
8	2	2D	Y	2	LeftToRightGreater	LeftToRightGreater	3.469	3
8	3	2D	Y	-5	FrontToBackGreater	FrontToBackGreater	6.938	3
8	4	2D	Y	-4	FrontToBackGreater	FrontToBackGreater	14.938	3
8	5	2D	Y	-2	FrontToBackGreater	LeftToRightGreater	11.25	3
8	6	2D	Y	3	LeftToRightGreater	LeftToRightGreater	9.328	3
8	7	2D	Y	-1	FrontToBackGreater	LeftToRightGreater	6.015	3
8	8	2D	Y	5	LeftToRightGreater	LeftToRightGreater	5.172	3
8	9	2D	Y	1	LeftToRightGreater	LeftToRightGreater	4.343	3
8	10	2D	Y	4	LeftToRightGreater	LeftToRightGreater	4.906	3
1	1	3D	N	-5	FrontToBackGreater	FrontToBackGreater	15.407	3
1	2	3D	N	-3	FrontToBackGreater	FrontToBackGreater	10.094	3
1	3	3D	N	5	LeftToRightGreater	LeftToRightGreater	11.203	3
1	4	3D	N	1	LeftToRightGreater	LeftToRightGreater	12.078	3
1	5	3D	N	-2	FrontToBackGreater	FrontToBackGreater	11.735	3
1	6	3D	N	2	LeftToRightGreater	LeftToRightGreater	7.328	3
1	7	3D	N	4	LeftToRightGreater	LeftToRightGreater	4.093	3
1	8	3D	N	-1	FrontToBackGreater	LeftToRightGreater	15.703	3
1	9	3D	N	3	LeftToRightGreater	LeftToRightGreater	9.859	3
1	10	3D	N	-4	FrontToBackGreater	FrontToBackGreater	4.469	3
2	1	3D	N	4	LeftToRightGreater	LeftToRightGreater	4.641	4
2	2	3D	N	3	LeftToRightGreater	LeftToRightGreater	4.141	4
2	3	3D	N	-1	FrontToBackGreater	FrontToBackGreater	16.922	4
2	4	3D	N	-2	FrontToBackGreater	FrontToBackGreater	5.047	4
2	5	3D	N	-4	FrontToBackGreater	FrontToBackGreater	3.719	4
2	6	3D	N	2	LeftToRightGreater	LeftToRightGreater	5.297	4
2	7	3D	N	-3	FrontToBackGreater	FrontToBackGreater	3.907	4
2	8	3D	N	1	LeftToRightGreater	LeftToRightGreater	4.735	4
2	9	3D	N	5	LeftToRightGreater	LeftToRightGreater	3.984	4
2	10	3D	N	-5	FrontToBackGreater	FrontToBackGreater	5.921	4
3	1	3D	N	-5	FrontToBackGreater	FrontToBackGreater	18.031	1
3	2	3D	N	-4	FrontToBackGreater	FrontToBackGreater	16.656	1
3	3	3D	N	4	LeftToRightGreater	LeftToRightGreater	7.969	1
3	4	3D	N	1	LeftToRightGreater	LeftToRightGreater	9.704	1
3	5	3D	N	2	LeftToRightGreater	LeftToRightGreater	8.031	1
3	6	3D	N	5	LeftToRightGreater	LeftToRightGreater	5.937	1
3	7	3D	N	-2	FrontToBackGreater	FrontToBackGreater	15.969	1
3	8	3D	N	-1	FrontToBackGreater	FrontToBackGreater	20.406	1
3	9	3D	N	3	LeftToRightGreater	LeftToRightGreater	9.25	1
3	10	3D	N	-3	FrontToBackGreater	FrontToBackGreater	10.593	1
4	1	3D	N	2	LeftToRightGreater	LeftToRightGreater	6.5	2
4	2	3D	N	-3	FrontToBackGreater	FrontToBackGreater	4.907	2
4	3	3D	N	-2	FrontToBackGreater	FrontToBackGreater	5.812	2
4	4	3D	N	-1	FrontToBackGreater	LeftToRightGreater	6.875	2
4	5	3D	N	4	LeftToRightGreater	LeftToRightGreater	8.015	2
4	6	3D	N	3	LeftToRightGreater	FrontToBackGreater	5.407	2
4	7	3D	N	-5	FrontToBackGreater	FrontToBackGreater	4	2
4	8	3D	N	-4	FrontToBackGreater	FrontToBackGreater	6.593	2
4	9	3D	N	1	LeftToRightGreater	LeftToRightGreater	19.125	2
4	10	3D	N	5	LeftToRightGreater	LeftToRightGreater	6.484	2

Table B.5 (continued)

Set	Presentation Order	2D or 3D?	Haptics?	Tol. Diff. (x - z)	Correct Response	Subject Response	Response Time	Env. Sequence
5	1	3D	N	-1	FrontToBackGreater	FrontToBackGreater	11.234	3
5	2	3D	N	-2	FrontToBackGreater	FrontToBackGreater	15.86	3
5	3	3D	N	1	LeftToRightGreater	LeftToRightGreater	12.515	3
5	4	3D	N	2	LeftToRightGreater	LeftToRightGreater	5.094	3
5	5	3D	N	3	LeftToRightGreater	LeftToRightGreater	8.782	3
5	6	3D	N	-3	FrontToBackGreater	FrontToBackGreater	8.968	3
5	7	3D	N	-4	FrontToBackGreater	FrontToBackGreater	12.031	3
5	8	3D	N	4	LeftToRightGreater	LeftToRightGreater	6.422	3
5	9	3D	N	-5	FrontToBackGreater	FrontToBackGreater	6.563	3
5	10	3D	N	5	LeftToRightGreater	LeftToRightGreater	6.969	3
6	1	3D	N	-3	FrontToBackGreater	FrontToBackGreater	21.875	4
6	2	3D	N	3	LeftToRightGreater	LeftToRightGreater	2.797	4
6	3	3D	N	-2	FrontToBackGreater	FrontToBackGreater	5.016	4
6	4	3D	N	5	LeftToRightGreater	LeftToRightGreater	3.218	4
6	5	3D	N	-1	FrontToBackGreater	FrontToBackGreater	7.578	4
6	6	3D	N	2	LeftToRightGreater	LeftToRightGreater	3.906	4
6	7	3D	N	-5	FrontToBackGreater	FrontToBackGreater	5.422	4
6	8	3D	N	-4	FrontToBackGreater	FrontToBackGreater	9.609	4
6	9	3D	N	4	LeftToRightGreater	LeftToRightGreater	6.687	4
6	10	3D	N	1	LeftToRightGreater	LeftToRightGreater	10.266	4
7	1	3D	N	-3	FrontToBackGreater	LeftToRightGreater	8.156	1
7	2	3D	N	3	LeftToRightGreater	LeftToRightGreater	8.203	1
7	3	3D	N	-2	FrontToBackGreater	FrontToBackGreater	7.219	1
7	4	3D	N	5	LeftToRightGreater	LeftToRightGreater	4.125	1
7	5	3D	N	-1	FrontToBackGreater	LeftToRightGreater	10.969	1
7	6	3D	N	2	LeftToRightGreater	LeftToRightGreater	11.188	1
7	7	3D	N	-5	FrontToBackGreater	FrontToBackGreater	5.407	1
7	8	3D	N	-4	FrontToBackGreater	FrontToBackGreater	7.469	1
7	9	3D	N	4	LeftToRightGreater	LeftToRightGreater	15.422	1
7	10	3D	N	1	LeftToRightGreater	LeftToRightGreater	9.094	1
8	1	3D	N	5	LeftToRightGreater	LeftToRightGreater	8.313	2
8	2	3D	N	-5	FrontToBackGreater	FrontToBackGreater	7.844	2
8	3	3D	N	2	LeftToRightGreater	FrontToBackGreater	21.469	2
8	4	3D	N	-2	FrontToBackGreater	LeftToRightGreater	3.125	2
8	5	3D	N	-3	FrontToBackGreater	FrontToBackGreater	6.703	2
8	6	3D	N	4	LeftToRightGreater	LeftToRightGreater	9.36	2
8	7	3D	N	3	LeftToRightGreater	LeftToRightGreater	6.828	2
8	8	3D	N	-1	FrontToBackGreater	FrontToBackGreater	8.906	2
8	9	3D	N	1	LeftToRightGreater	LeftToRightGreater	4.063	2
8	10	3D	N	-4	FrontToBackGreater	FrontToBackGreater	7.625	2
1	1	3D	Y	-2	FrontToBackGreater	FrontToBackGreater	17.266	4
1	2	3D	Y	4	LeftToRightGreater	LeftToRightGreater	2.391	4
1	3	3D	Y	-5	FrontToBackGreater	FrontToBackGreater	4.422	4
1	4	3D	Y	-4	FrontToBackGreater	FrontToBackGreater	4.125	4
1	5	3D	Y	5	LeftToRightGreater	LeftToRightGreater	6.766	4
1	6	3D	Y	3	LeftToRightGreater	LeftToRightGreater	3.969	4
1	7	3D	Y	-3	FrontToBackGreater	FrontToBackGreater	6.36	4
1	8	3D	Y	2	LeftToRightGreater	LeftToRightGreater	3.953	4
1	9	3D	Y	1	LeftToRightGreater	LeftToRightGreater	10.578	4
1	10	3D	Y	-1	FrontToBackGreater	FrontToBackGreater	12.594	4

Table B.5 (continued)

Set	Presentation Order	2D or 3D?	Haptics?	Tol. Diff. (x - z)	Correct Response	Subject Response	Response Time	Env. Sequence
2	1	3D	Y	5	LeftToRightGreater	LeftToRightGreater	2.703	3
2	2	3D	Y	-2	FrontToBackGreater	FrontToBackGreater	4.125	3
2	3	3D	Y	3	LeftToRightGreater	LeftToRightGreater	2.328	3
2	4	3D	Y	-5	FrontToBackGreater	FrontToBackGreater	6.891	3
2	5	3D	Y	1	LeftToRightGreater	LeftToRightGreater	3.703	3
2	6	3D	Y	-1	FrontToBackGreater	LeftToRightGreater	4.844	3
2	7	3D	Y	-4	FrontToBackGreater	FrontToBackGreater	20.954	3
2	8	3D	Y	-3	FrontToBackGreater	FrontToBackGreater	10.906	3
2	9	3D	Y	4	LeftToRightGreater	LeftToRightGreater	2.671	3
2	10	3D	Y	2	LeftToRightGreater	LeftToRightGreater	3.5	3
3	1	3D	Y	-2	FrontToBackGreater	FrontToBackGreater	8.219	2
3	2	3D	Y	4	LeftToRightGreater	LeftToRightGreater	5	2
3	3	3D	Y	5	LeftToRightGreater	LeftToRightGreater	5.875	2
3	4	3D	Y	1	LeftToRightGreater	LeftToRightGreater	16.907	2
3	5	3D	Y	2	LeftToRightGreater	LeftToRightGreater	5.656	2
3	6	3D	Y	-1	FrontToBackGreater	FrontToBackGreater	22.14	2
3	7	3D	Y	-5	FrontToBackGreater	FrontToBackGreater	7.516	2
3	8	3D	Y	3	LeftToRightGreater	LeftToRightGreater	5.281	2
3	9	3D	Y	-3	FrontToBackGreater	FrontToBackGreater	7.109	2
3	10	3D	Y	-4	FrontToBackGreater	FrontToBackGreater	6.157	2
4	1	3D	Y	-2	FrontToBackGreater	FrontToBackGreater	26.547	1
4	2	3D	Y	-5	FrontToBackGreater	FrontToBackGreater	4.375	1
4	3	3D	Y	4	LeftToRightGreater	LeftToRightGreater	2.719	1
4	4	3D	Y	3	LeftToRightGreater	LeftToRightGreater	7.375	1
4	5	3D	Y	-1	FrontToBackGreater	FrontToBackGreater	9.141	1
4	6	3D	Y	-3	FrontToBackGreater	FrontToBackGreater	8.422	1
4	7	3D	Y	5	LeftToRightGreater	LeftToRightGreater	3.234	1
4	8	3D	Y	1	LeftToRightGreater	LeftToRightGreater	14.063	1
4	9	3D	Y	2	LeftToRightGreater	LeftToRightGreater	5.672	1
4	10	3D	Y	-4	FrontToBackGreater	FrontToBackGreater	5.672	1
5	1	3D	Y	-1	FrontToBackGreater	FrontToBackGreater	19.797	4
5	2	3D	Y	-4	FrontToBackGreater	FrontToBackGreater	6.61	4
5	3	3D	Y	1	LeftToRightGreater	LeftToRightGreater	9.703	4
5	4	3D	Y	-2	FrontToBackGreater	FrontToBackGreater	11.437	4
5	5	3D	Y	2	LeftToRightGreater	LeftToRightGreater	8.641	4
5	6	3D	Y	4	LeftToRightGreater	LeftToRightGreater	5.985	4
5	7	3D	Y	3	LeftToRightGreater	LeftToRightGreater	9.297	4
5	8	3D	Y	-5	FrontToBackGreater	FrontToBackGreater	7.438	4
5	9	3D	Y	-3	FrontToBackGreater	FrontToBackGreater	10.609	4
5	10	3D	Y	5	LeftToRightGreater	LeftToRightGreater	8.781	4
6	1	3D	Y	1	LeftToRightGreater	LeftToRightGreater	5.094	3
6	2	3D	Y	-3	FrontToBackGreater	FrontToBackGreater	3.469	3
6	3	3D	Y	-1	FrontToBackGreater	FrontToBackGreater	7.078	3
6	4	3D	Y	2	LeftToRightGreater	LeftToRightGreater	3.282	3
6	5	3D	Y	-5	FrontToBackGreater	FrontToBackGreater	3.031	3
6	6	3D	Y	4	LeftToRightGreater	LeftToRightGreater	3.953	3
6	7	3D	Y	-2	FrontToBackGreater	FrontToBackGreater	6.188	3
6	8	3D	Y	5	LeftToRightGreater	LeftToRightGreater	3.219	3
6	9	3D	Y	3	LeftToRightGreater	LeftToRightGreater	3	3
6	10	3D	Y	-4	FrontToBackGreater	FrontToBackGreater	3.204	3

Table B.5 (continued)

Set	Presentation Order	2D or 3D?	Haptics?	Tol. Diff. (x - z)	Correct Response	Subject Response	Response Time	Env. Sequence
7	1	3D	Y	1	LeftToRightGreater	LeftToRightGreater	4.25	2
7	2	3D	Y	-3	FrontToBackGreater	FrontToBackGreater	5.453	2
7	3	3D	Y	-1	FrontToBackGreater	FrontToBackGreater	3.891	2
7	4	3D	Y	2	LeftToRightGreater	LeftToRightGreater	3.672	2
7	5	3D	Y	-5	FrontToBackGreater	FrontToBackGreater	4.656	2
7	6	3D	Y	4	LeftToRightGreater	LeftToRightGreater	3	2
7	7	3D	Y	-2	FrontToBackGreater	FrontToBackGreater	8.062	2
7	8	3D	Y	5	LeftToRightGreater	LeftToRightGreater	3.641	2
7	9	3D	Y	3	LeftToRightGreater	LeftToRightGreater	1.343	2
7	10	3D	Y	-4	FrontToBackGreater	FrontToBackGreater	5.859	2
8	1	3D	Y	-4	FrontToBackGreater	FrontToBackGreater	7.047	1
8	2	3D	Y	1	LeftToRightGreater	LeftToRightGreater	6	1
8	3	3D	Y	3	LeftToRightGreater	LeftToRightGreater	7.625	1
8	4	3D	Y	-2	FrontToBackGreater	FrontToBackGreater	10.875	1
8	5	3D	Y	-5	FrontToBackGreater	FrontToBackGreater	5.593	1
8	6	3D	Y	4	LeftToRightGreater	LeftToRightGreater	3.953	1
8	7	3D	Y	-3	FrontToBackGreater	FrontToBackGreater	12.813	1
8	8	3D	Y	5	LeftToRightGreater	LeftToRightGreater	3	1
8	9	3D	Y	2	LeftToRightGreater	LeftToRightGreater	3.329	1
8	10	3D	Y	-1	FrontToBackGreater	LeftToRightGreater	21.578	1

B.3 DATA FOR PEG-IN-HOLE INSERTION EXPERIMENT

The raw data for the peg-in-hole insertion experiment is provided in Table B.6. A description of the column headers is provided below.

- *Set* – One set corresponds to completion of the task for all four combinations of *2D or 3D?* and *Hapics?*.
- *2D or 3D?* – 2D monitor visualization or 3D visualization with shutter glasses.
- *Hapics?* – Whether haptic feedback is active or not.
- *t_insert_#* – Time (in seconds) required to insert the peg into the hole on the #th iteration.
- *t_remove_#* – Time (in seconds) required to remove the peg from the hole on the #th iteration.
- *Env. Sequence* – Environment Sequence

Table B.6: Raw Data for Peg-in-Hole Insertion Experiment

Set	2D or 3D?	Haptics?	t_insert_1	t_remove_1	t_insert_2	t_remove_2	t_insert_3	t_remove_3	t_insert_4	t_remove_4	t_insert_5	t_remove_5	Env. Sequence
1	2D	N	7.844	6.156	10.469	6.484	4.766	5.062	6.391	10.938	4.625	4.421	1
2	2D	N	11.75	5.125	10.953	6.719	5.75	4.75	8.578	3.75	9.047	3.906	2
3	2D	N	6.235	6.093	7.516	4.5	6.766	5.468	6.938	4.687	11.078	3.672	3
4	2D	N	5.328	2.329	5.89	2.344	3.391	2.093	2.86	3.765	2.094	3.5	4
5	2D	N	5.938	14.047	8.546	3.75	5.813	3.469	5.578	5.031	6.266	6.968	1
6	2D	N	5.437	2.688	3.468	2.485	5.812	11.375	5.75	5.297	3.141	2.734	2
7	2D	N	3.172	11.609	7.782	4.343	5.938	4.406	4.188	6.234	3.437	4.219	3
8	2D	N	4.125	3.235	9.375	4.39	6.907	3.906	11.484	3.688	5.359	3.906	4
1	2D	Y	11.218	5.703	4.907	4.172	6.531	4.078	8.641	3.343	3.407	3.312	2
2	2D	Y	6.406	8.922	5.062	5.954	6.312	2.703	5.438	4.812	4.36	5.359	1
3	2D	Y	9.063	4.578	5.453	4.781	6	4.641	4.703	5.109	5.813	3.547	4
4	2D	Y	6.235	1.765	6.079	1.859	3.766	2.031	5.547	3.875	4.234	4.688	3
5	2D	Y	5.063	3.781	5.563	3.015	4.36	3.687	3.594	2.969	5.437	2.5	2
6	2D	Y	8.782	4.812	3.672	3.391	3.531	3.672	4.265	4.219	10.828	1.641	1
7	2D	Y	4.672	3.906	6	5.047	6.86	3.75	5.328	3.234	3.281	2.922	4
8	2D	Y	5.188	4.219	5.343	4.157	5.984	4.172	4.797	3.906	4.125	3.031	3
1	3D	N	13.204	8.109	6.094	2.89	15.891	10.187	6.765	3.016	12	5.109	3
2	3D	N	6.766	4.797	8.375	9.359	11.828	4.938	9.203	4.625	12.078	6.484	4
3	3D	N	11.86	7.75	9.875	3.39	13.594	4.375	12.688	4.656	7.266	4.937	1
4	3D	N	7.312	3.516	1.781	4.813	5.812	3.875	2.297	2.828	2.063	2.312	2
5	3D	N	16.562	3.984	5.782	3.25	7.562	2.719	4.812	3.407	5.359	3.953	3
6	3D	N	3.141	1.64	6.563	1.469	5.343	1.563	3.234	1.641	6.547	1.64	4
7	3D	N	4.562	7.516	6.328	6.375	5.469	4.812	5.891	6.359	11.703	4.11	1
8	3D	N	7.625	5.797	4.093	4.422	4.235	8.265	4.016	2.875	5.766	3.468	2
1	3D	Y	5.047	6.157	6.171	3.266	4.266	3.031	7.281	2.422	4.797	3.078	4
2	3D	Y	7.594	2.578	13.391	2.844	5.671	2.813	5.625	3.328	4.328	3.125	3
3	3D	Y	7.531	4.922	5.266	4.344	4.593	4.172	4.531	4.172	4.5	4.516	2
4	3D	Y	4.015	2.782	2.828	5.953	3.547	2.437	2.969	1.75	2.25	1.859	1
5	3D	Y	3.469	4.672	5.016	3.687	5.11	3.578	4.687	2.86	4.468	3.407	4
6	3D	Y	2.672	1.469	3.297	1.718	2.516	1.844	2.828	2.25	3.125	2.234	3
7	3D	Y	4.391	5.171	6.438	4.047	4.625	3.875	4.14	5.735	7.094	3.765	2
8	3D	Y	8.359	7.032	5.562	3.438	5.234	3.656	5.891	3.391	3.625	3.75	1

B.4 DATA FOR TRAIN TRACK ASSEMBLY EXPERIMENT

The raw data for the train track assembly experiment is provided in Table B.7. A description of the column headers is provided below.

- *Set* – One set corresponds to completion of the task for all four combinations of *2D or 3D?* and *Hapics?*.
- *Environment* – Task performed in real or virtual environment.
- *2D or 3D?* – 2D monitor visualization or 3D visualization with shutter glasses.
- *Hapics?* – Whether haptic feedback is active or not.
- *t_pick_#* – Time (in seconds) required to grasp the #th section of the train track for placement.
- *t_place_#* – Time (in seconds) required to assemble the #th section of the train track.
- *Env. Sequence* – Environment Sequence

Table B.7: Raw Data for Train Track Assembly Experiment

Set	Environment	2D or 3D?	Haptics?	t_pick_1	t_place_1	t_pick_2	t_place_2	t_pick_3	t_place_3	Env. Sequence
1	Real	n/a	n/a	1.15	1.891	0.906	0.953	0.766	0.484	1
2	Real	n/a	n/a	0.875	1.672	0.906	1	0.656	0.782	3
3	Real	n/a	n/a	0.891	1.359	0.75	1.031	0.766	1.484	5
4	Real	n/a	n/a	0.781	1.89	1	1.204	0.921	0.782	2
5	Real	n/a	n/a	1.032	1.484	1.516	1.125	0.812	1.438	4
6	Real	n/a	n/a	1.25	2.266	1.953	2.203	1.125	1.313	1
7	Real	n/a	n/a	0.735	0.531	0.547	0.672	0.562	0.5	3
8	Real	n/a	n/a	0.906	0.781	0.75	1.531	0.578	0.922	5
9	Real	n/a	n/a	1.047	0.813	0.609	0.688	0.515	0.563	2
10	Real	n/a	n/a	0.766	1.39	0.672	0.813	0.484	0.766	4
11	Real	n/a	n/a	1.172	0.765	0.922	0.922	0.61	0.828	1
12	Real	n/a	n/a	0.75	1.266	0.984	1.172	0.578	0.703	3
13	Real	n/a	n/a	1.266	0.734	0.547	0.594	0.515	1.375	5
14	Real	n/a	n/a	2.109	1.109	0.969	1.063	0.562	1.188	2
15	Real	n/a	n/a	1.453	0.953	0.672	1.203	0.532	0.828	4
1	Virtual	2D	N	4.503	62.937	6.079	23.625	4.203	45.125	2
2	Virtual	2D	N	6.484	146.578	4.953	8.61	3.171	10.688	1
3	Virtual	2D	N	1.734	97.531	9.985	42.25	7.547	11.937	3
4	Virtual	2D	N	3.094	88.8205	10.6635	15.563	8.297	25.812	4
5	Virtual	2D	N	4.219	32.406	8.969	11.234	4.641	42.656	5
6	Virtual	2D	N	3.406	123.078	6.938	32.578	10.562	34.375	2
7	Virtual	2D	N	9.687	45.703	11.25	18.86	11.296	17.079	1
8	Virtual	2D	N	2.641	7.219	4.953	56.328	3.64	24.922	3
9	Virtual	2D	N	3.422	99.984	4.141	29.797	4.641	41.531	4
10	Virtual	2D	N	3.578	83.266	6.093	54.75	4.454	16.688	5
11	Virtual	2D	N	3.062	21.891	3.39	31.36	4.406	8.313	2
12	Virtual	2D	N	2.843	180.141	72.922	90.843	73.907	59.515	1
13	Virtual	2D	N	3.469	19.312	7.594	18.922	6.047	27.109	3
14	Virtual	2D	N	4.469	71.281	6.734	15.11	2.547	47.406	4
15	Virtual	2D	N	6.468	47.188	4.766	86.921	2.329	135.265	5
1	Virtual	2D	Y	3.194	18.531	7.859	5.36	5.812	5.516	3
2	Virtual	2D	Y	2.266	51.578	4.531	20.672	6.031	13.828	2
3	Virtual	2D	Y	3	13.656	9.219	9.703	7.735	21.437	4
4	Virtual	2D	Y	2.609	51.266	4.719	14.515	4.688	5.64	5
5	Virtual	2D	Y	6.657	13.968	8.868	42.763	6.807	9.812	1
6	Virtual	2D	Y	6.422	11.984	5.953	10.75	6.563	14.187	3
7	Virtual	2D	Y	2.328	14.172	6.438	7.969	8.26	16.531	2
8	Virtual	2D	Y	1.64	18.828	3.469	6.938	5.093	9.391	4
9	Virtual	2D	Y	1.984	67.985	3.64	7.797	5.766	30.703	5
10	Virtual	2D	Y	2.047	24.672	3.859	22.156	10.219	5.391	1
11	Virtual	2D	Y	3.219	60.343	2.875	9.375	2.782	13.234	3
12	Virtual	2D	Y	6.907	39.343	3.375	22.266	4.438	9.906	2
13	Virtual	2D	Y	4.14	15.203	3.094	12	4.047	10.656	4
14	Virtual	2D	Y	4.547	29.516	6.969	5.578	3.64	16.797	5
15	Virtual	2D	Y	7.297	19.265	3.875	21.625	4.188	12.25	1

Table B.7 (continued)

Set	Environment	2D or 3D?	Haptics?	t_pick1	t_place1	t_pick2	t_place2	t_pick3	t_place3	Env. Sequence
1	Virtual	3D	N	3.841	46.156	15.688	23.844	1.734	47.672	4
2	Virtual	3D	N	4.813	16.468	4.469	18.188	4.109	23.922	5
3	Virtual	3D	N	6.188	125.984	9.594	29.156	8.469	22.641	2
4	Virtual	3D	N	7.578	165.875	4.407	37.437	5.86	32.171	1
5	Virtual	3D	N	2.453	25	8.297	21.797	6.641	66.797	3
6	Virtual	3D	N	4.765	23	10.703	11.453	3.969	29.891	4
7	Virtual	3D	N	2.859	19.844	3.328	7.297	5.203	6.532	5
8	Virtual	3D	N	4.656	16.781	3.703	16.297	1.703	51.485	2
9	Virtual	3D	N	2.828	126.109	5.485	9.453	3.781	25.266	1
10	Virtual	3D	N	1.734	71.688	9.5	13.343	7.375	80.922	3
11	Virtual	3D	N	3.813	44.656	8.703	29.5	3.391	54.843	4
12	Virtual	3D	N	6.312	98.172	5.047	18.266	1.578	33.515	5
13	Virtual	3D	N	5.469	32.515	5.157	15.593	7.657	45.828	2
14	Virtual	3D	N	6.61	191.922	12.062	12.532	7.703	46.703	1
15	Virtual	3D	N	6.5	32.875	4.422	35.062	3.063	78.953	3
1	Virtual	3D	Y	2.923	25.969	9.75	3.359	2.469	5.156	5
2	Virtual	3D	Y	4.875	54.187	4.844	48.047	14.469	21.14	4
3	Virtual	3D	Y	2.687	113.75	7.375	11.109	5.579	13.687	1
4	Virtual	3D	Y	3.641	37.578	3.64	12.594	10.734	3.782	3
5	Virtual	3D	Y	4.562	100.344	6	16.25	6.844	28.531	2
6	Virtual	3D	Y	3.609	39.203	3.329	8.265	4.844	19.281	5
7	Virtual	3D	Y	3.782	9.234	3.047	3.594	5.578	8.453	4
8	Virtual	3D	Y	3.141	15.015	2.719	10.938	2.953	5.968	1
9	Virtual	3D	Y	2.329	13.515	6.047	13.016	3.765	17.141	3
10	Virtual	3D	Y	3.125	11.093	7.25	20.032	7.468	30.813	2
11	Virtual	3D	Y	4.672	31.375	2.141	7.812	4.031	8.094	5
12	Virtual	3D	Y	3.093	63.125	6.703	51.36	6.844	13.703	4
13	Virtual	3D	Y	6.359	24.906	6.235	17.078	2.391	9.484	1
14	Virtual	3D	Y	3.875	52.047	9.344	16.625	8.312	32.688	3
15	Virtual	3D	Y	8.531	22.188	11.625	18.625	5.578	18.719	2

B.5 DATA FOR BRAKE PAD REPLACEMENT EXPERIMENT

The raw data for the brake pad replacement experiment is provided in Table B.8. A description of the column headers is provided below.

- *Set* – One set corresponds to completion of the task for all four combinations of *2D or 3D?* and *Hapics?*.
- *2D or 3D?* – 2D monitor visualization or 3D visualization with shutter glasses.
- *Hapics?* – Whether haptic feedback is active or not.
- *t_pick_#* and *t_place_#* – Time (in seconds) required to grasp and place the #th component in the replacement procedure as follows.
- 1st component: Time to grasp and remove old brake pad.
- 2nd component: Time required to first touch and push in the piston.
- 3rd component: Time to grasp and assemble new brake pad.
- *Env. Sequence* – Environment Sequence

Table B.8: Raw Data for Brake Pad Replacement Experiment

Set	2D or 3D?	Haptics?	t_pick_1	t_place_1	t_pick_2	t_place_2	t_pick_3	t_place_3	Env. Sequence
1	2D	N	4.797	22.281	3.016	3.547	8.093	8.953	1
2	2D	N	3.891	13.219	2.562	2.734	3.61	13.078	2
3	2D	N	5.25	13.859	3.36	3.656	5.547	14.25	3
4	2D	N	4.093	9.5	2.782	3.375	6.359	11.078	4
5	2D	N	6.015	12.438	4.594	3.5	11.187	23.797	1
6	2D	N	7.375	19.172	5.891	4.797	7.921	19.704	2
7	2D	N	3.312	7.782	2.875	4.593	4.578	14.219	3
8	2D	N	4.672	23.672	3.437	8.844	2.375	9.812	4
9	2D	N	7.14	17.031	5.813	2.484	2.75	26.719	1
10	2D	N	6.469	14.156	7.89	3.235	13.594	8	2
11	2D	N	4.297	10.047	3.625	4.719	4.046	6.204	3
12	2D	N	2.281	18.094	7.109	3.407	6.75	32.281	4
1	2D	Y	5.726	9.656	4.688	6.547	13.89	10	2
2	2D	Y	4.063	8.047	3.484	3.359	3.75	9.282	1
3	2D	Y	4.062	9.125	2.563	4.125	4.672	9.703	4
4	2D	Y	5.266	5.687	2.813	3.937	5.36	8.125	3
5	2D	Y	4.656	11	3.516	4.297	5.906	12.609	2
6	2D	Y	9.75	7.703	4.657	5.375	6.25	28.156	1
7	2D	Y	2.39	10.656	1.735	3.297	2.875	5.453	4
8	2D	Y	3.734	11.094	1.765	3.922	3.453	7.704	3
9	2D	Y	7.609	10.141	3.969	3.531	2.781	12.297	2
10	2D	Y	5.563	8.406	4.609	3.344	8.578	6.063	1
11	2D	Y	2.906	8.984	2.079	4.546	2.813	8.359	4
12	2D	Y	4.515	8.469	5.016	3.719	3.875	23.406	3
1	3D	N	6.384	12.593	14.969	4.047	9.187	15.579	3
2	3D	N	5.906	11.015	4.875	2.563	3.25	12.281	4
3	3D	N	5.985	7.937	0.938	2.984	10.656	23.016	1
4	3D	N	6.219	12.187	1.844	3.156	5.047	15.625	2
5	3D	N	4.125	13.281	4.063	3.359	4.547	27.594	3
6	3D	N	6.453	11	3.914	4.93	3.406	14.047	4
7	3D	N	8.188	11.812	1.141	3.422	2.484	14	1
8	3D	N	4.265	11.829	3.25	2.906	4.656	24.406	2
9	3D	N	3.563	9.468	5.219	2.719	3.187	7.875	3
10	3D	N	6.563	21.031	5.391	4.344	8.484	15.344	4
11	3D	N	7.641	16.329	2.812	3.609	4.079	16.328	1
12	3D	N	2.719	15.219	12.078	3.437	5.75	24.375	2
1	3D	Y	4.725	10.218	4.063	3.703	7.812	11.344	4
2	3D	Y	5.297	8.062	3.016	3.219	3.437	13.875	3
3	3D	Y	5.718	17.454	2.718	2.782	9.515	16.063	2
4	3D	Y	4.766	9.156	5.703	3.984	3.235	7.859	1
5	3D	Y	3.203	7.25	3.078	3.453	4.391	8.234	4
6	3D	Y	10.609	6.078	4.391	3.781	5.844	8.078	3
7	3D	Y	5.219	6.625	3.031	3.156	2.594	5.391	2
8	3D	Y	6.531	11.672	2.765	3	1.844	6.344	1
9	3D	Y	3.594	7.375	3.171	3.266	2.688	8.156	4
10	3D	Y	5.953	28.078	2.344	3.313	9.156	6.562	3
11	3D	Y	5.156	8.344	3.265	3.125	6.235	6.687	2
12	3D	Y	4.125	11.843	11.594	9.438	3.562	15.75	1

B.6 RESPONSES TO EXPERIMENTAL PARTICIPANT QUESTIONNAIRE

The results of the experimental subject survey are provided in Table B.9. The first column in the table, *Subject #*, is a generic number corresponding to each subject who completed the survey. Each of the remaining columns correspond the each subject's response to the ten questions listed below. For the questions with a numerical response, the participants were asked for an integer answer on a scale of 1-4. See Appendix A.7 for details of the questionnaire.

Survey Questions:

1. How would you rate your knowledge and experience with computers?
2. How would you rate your knowledge and experience with virtual reality environments?
3. Have you ever used haptic technology or haptic devices?
4. How would you rate your knowledge with haptic technology and experience in using haptic devices?
5. When compared to your experiences manipulating objects in real life, how similar were manipulations of virtual objects through haptic feedback?
6. When compared to your experiences manipulating objects in real life, how easy were manipulations of virtual objects through haptic feedback?
7. How realistic do you feel the interactions between virtual objects were (i.e., Virtual Object #1 interacting with Virtual Object #2)?
8. Have you ever used 3D shutter glasses?
9. How much of an improvement did the 3D shutter glasses provide over 2D monitor viewing for interacting with the virtual environment?

10. How much of an improvement did haptic feedback provide over no haptic feedback for interacting with the virtual environment?

Table B.9: Response Listing for Experimental Subject Survey

Subject #	Question									
	1	2	3	4	5	6	7	8	9	10
1	2	1	Y	1	2	3	2	Y	1	3
2	3	1	N	1	2	3	4	N	1	4
3	2	1	Y	2	3	3	2	N	3	4
4	4	2	N	1	2	3	3	N	3	4
5	2	1	N	1	1	1	2	N	2	4
6	3	1	N	1	3	3	4	N	4	4
7	3	3	Y	3	2	2	3	Y	2	4
8	4	4	Y	3	2	1	2	Y	2	3
9	4	3	Y	2	2	2	2	N	2	4
10	2	1	N	1	2	2	3	N	2	3
11	4	2	Y	2	4	3	4	Y	4	4
12	3	2	N	1	2	2	3	N	2	4
13	3	1	N	1	3	2	4	N	3	4
14	2	1	N	1	4	3	3	N	2	4
15	3	1	N	1	3	3	3	N	2	4
16	3	1	N	1	4	4	4	N	1	4
17	3	1	Y	1	3	3	3	Y	4	4

APPENDIX C

DISCUSSION OF STATISTICAL DESIGN AND ANALYSIS TECHNIQUES

Several statistical methods and techniques were utilized during the design and analysis of the experiments conducted in Chapters 5 and 6. In this appendix, discussion will include a description of the statistical techniques used for analysis (Appendix C.1), the latin square experimental design technique (Appendix C.2), and the validation process for assuring the appropriateness of each model for the experimental data collected (Appendix C.3). The references for the information contained in this appendix can be found in (Devore 1995; Hayter 1996; Neter *et al.* 1996) and the MINITAB software help documentation.

- **Appendix C.1:** Statistical Techniques Utilized During Analysis of Experimental Data
- **Appendix C.2:** Discussion on Repeated Measures and Latin Square Design
- **Appendix C.3:** Validating the Appropriateness of Statistical Models Used

C.1 STATISTICAL TECHNIQUES UTILIZED DURING ANALYSIS OF EXPERIMENTAL DATA

A description of each of the statistical analysis techniques utilized is provided below.

Analysis of Variance

The majority of statistical tests performed during the analysis of experimental data used analysis of variance, or ANOVA. The ANOVA technique determines the amount of variability in the data that is caused by each of the treatments, or control variables. From this evaluation, the statistical significance of changes in treatment levels on the response variable can be determined. For clarity, the steps involved in computing the ANOVA table are described. For simplicity, a one-way ANOVA will be discussed, in which there is only one control variable. Analysis of variance with multiple factors is essentially an extension of this process, but the general procedure remains the same.

Step 1

Calculate degrees of freedom for each treatment (otherwise known as factor or control variable), an error term, and the total data set. For each treatment, the degrees of freedom is equal to the number of treatment levels minus 1. For instance, when the availability of haptic feedback is a factor in the ANOVA, the degrees of treatment for haptics as a factor is equal to 2 levels (haptic and non-haptic) – 1. The equation for degrees of freedom for each treatment is given below

$$DF_{Tr} = n_{levels} - 1 \tag{C.1}$$

In addition, the total degrees of freedom for the data and the degrees of freedom for the error term are calculated using the equations below.

$$DF_{total} = n_{total} - 1 \tag{C.2}$$

$$DF_{error} = DF_{total} - \sum_{i} DF_{Tr_i} \tag{C.3}$$

The total degrees of freedom is equal to the total number of observations in the data set minus 1. The degrees of freedom for the error term is equal to the total degrees of freedom minus the degrees of freedom for each treatment.

Step 2

The next step is to calculate the sum of squares for treatments, the sum of squares for error, and the total sum of squares. The sum of squares for treatments is a measure of the variability between the levels of a treatment. For instance, how much does the response variable change when a given treatment level is changed. The equation for the sum of squares for treatments given two control variables follows.

$$SSTr = \sum_{i=1}^{k} n_i (\bar{x}_{i\cdot} - \bar{x}_{\cdot\cdot})^2 \tag{C.4}$$

In this equation, k stands for the number of treatment levels for a given treatment, n_i is the number of observations at this treatment level, $\bar{x}_{i\cdot}$ is the population mean for the i^{th} level of the treatment, and $\bar{x}_{\cdot\cdot}$ is the mean for the entire population of data.

The total sum of squares is a measure of the total variability in the data set and is calculated with the equation below.

$$SST = \sum_{i=1}^{k} \sum_{j=1}^{n_i} (x_{ij} - \bar{x}_{\cdot\cdot})^2 \tag{C.5}$$

Here, x_{ij} stands for the j^{th} observation and the i^{th} factor level.

Finally, the sum of squares for error can be calculated as the difference between *SSTr* and *SST*.

$$SSE = SST - SSTr \tag{C.6}$$

This term accounts for the variability of the data within each factor level.

Step 3

The mean squares for treatments and mean square error are calculated simply by dividing the sum of squares term by the degrees of freedom. The mean squares for treatments is shown below.

$$MSTr = \frac{SSTr}{DF_{Tr}} \tag{C.7}$$

The equation for mean square error follows.

$$MSE = \frac{SSE}{DF_{error}} \tag{C.8}$$

Step 4

Using the mean square for treatments and the mean square error, an F-statistic is calculated for the treatment as follows.

$$F = \frac{MSTr}{MSE} \tag{C.9}$$

This F-statistic is then used to determine the probability that a random variable X is a part of an F distribution with degrees of freedom k-1 and n_T-k. The equation for the probability term, known as the p-value, is shown below.

$$p - value = P(X \geq F) \tag{C.10}$$

This p-value is used to determine whether the mean response values for all factor levels are equal or different. For a small p-value, the mean response values for each factor level are declared different. In this dissertation, the critical p-value used is 0.05, for which smaller p-values are declared significant. For p-values less than 0.05, there is at least a 95% probability that the levels of treatment affect the response variable.

Binary Logistic Regression

In this dissertation, binary logistic regression is utilized to determine the relationship between a binary response variable and any variety of predictor variables. For logistic regression, the response variable is modeled as a Bernoulli random variable. Also, the relation between the response variable and the various predictor variables is modeled as a sigmoidal, or S-shaped function. Due to complexity, the details of the logistic regression equations will not be shown here. The details and equations for computing the logistic regression can be found in (Neter *et al.* 1996) and in the MINTAB software documentation.

During the MINITAB analysis shown in this dissertation, the logit function was used for transforming the logistic response function, which is non-linear, into a linear equation. The result of the logistic regression analysis is a p-value and an odds ratio. The p-value is interpreted in the same manner as with the ANOVA, such that a p-value less than the critical value of 0.05 is evidence that the factor corresponding to that p-value is significant in affecting the response variable. The odds ratio provides an estimation of the increase in the response variable for a unit step increase of the predictor variable. For an odds ratio of 1, changes in the predictor variable have no effect on the response variable.

For odds ratios greater than 1, an increase in the predictor variable increases the probability that the response variable will be 1. Just the opposite occurs for an odds ratio less than 1, in that an increase in the predictor variable, reduces the probability of the response variable being 1.

2-sample t-test

The 2-sample t-test compares data from two populations of a normally distributed response variable. For the analysis conducted in this dissertation, the t-test is used to determine whether the means of two populations are statistically different. To do this, the following t-statistic is computed.

$$t = \frac{\bar{x} - \bar{y}}{\sigma\sqrt{\frac{1}{n} + \frac{1}{m}}} \tag{C.11}$$

In this equation, $\bar{x}$ and $\bar{y}$ represent the mean of each population, and σ is the estimated standard deviation of each population, which is assumed to be equal. Also, *n* and *m* are the number of observations for each population. Using this test statistic, a two-sided p-value is calculated as shown below.

$$p - value = 2 \times P(X > |t|) \tag{C.12}$$

In this equation, X is a random variable that has a t-distribution with $n + m - 2$ degrees of freedom. As with other analysis techniques, the two populations are considered to have statistically different means if the p-value is less the 0.05.

2-proportion hypothesis test

The 2-proportion hypothesis test is similar in the 2-sample t-test in function, with the exception that the response variable is binary rather than normally distributed. To

conduct this analysis, the idea of a population proportion must first be explained. The population proportion, $\hat{p}$, is defined as the probability that the response value is 1 for any given binary data set. To determine the effect of the predictor variable on the binary response variable, the z-statistic is first calculated as shown below.

$$z = \frac{\hat{p}_A - \hat{p}_B}{\sqrt{\hat{p}(1-\hat{p})(\frac{1}{n}+\frac{1}{m})}} \tag{C.13}$$

In this equation, $\hat{p}_A$ is the population proportion for the first set of data and $\hat{p}_B$ is the population proportion for the second set. Also, *n* and *m* are the sample sizes of the two populations. Lastly, $\hat{p}$ is defined below.

$$\hat{p} = \frac{x+y}{n+m} \tag{C.14}$$

Here, *x* and *y* are the number of affirmative responses, or values of 1, in each population. The z-statistic defined above is then used to compute the following p-value.

$$p - value = 2 \times \Phi(-|z|) \tag{C.15}$$

In this last equation, $\Phi()$ refers to the normal distribution. Again, there is considered to be a significant difference in the population proportions for the two data sets if the p-value is less than 0.05.

C.2 DISCUSSION ON REPEATED MEASURES AND LATIN SQUARE DESIGN

During four of the experiments discussed in this dissertation, the motion tolerance study, peg-in-hole task, train track assembly, and brake pad replacement, several variations of HIDRA were used. These variations included the haptic and non-haptic versions of the simulation and 2D monitor viewing versus 3D visualization with shutter glasses. During these experiments, all participants completed the procedures involved in each of the four combinations of HIDRA. This type of experimental design is referred to as repeated measures. The advantage of a repeated measures design is that all sources of error between experimental subjects are excluded from the experimental error. However, with this advantage comes a potential disadvantage known as the order effect. This refers to the fact that the performance of experimental subjects may improve or degrade as the procedure is repeated in each of the four versions of HIDRA due to factors such as learning or fatigue.

In order to prevent the effect of learning or fatigue from affecting the main treatments (availability of haptic feedback and type of visualization), the latin square design was implemented. Latin square design randomizes the order in which the various environments are presented to each participant, such that the overall performance in each environment is not affected by fatigue or learning.

The environment ordering for each experimental participant is shown in Tables C1-C.4. In these tables, each row corresponds the environment ordering presented to each subject. Also, the letters within each cell represents a particular environment as follows.

- A = 2D, Non-haptic
- B = 2D, Haptic

- C = 3D, Non-haptic
- D = 3D, Haptic
- E = Real World (Train Track Assembly only).

Table C.1: Environment Ordering for Each Participant in Motion Tolerance Study Based on Latin Square Design Method

Subject	Order			
	1	2	3	4
1	A	B	C	D
2	B	A	D	C
3	C	D	A	B
4	D	C	B	A
5	A	B	C	D
6	B	A	D	C
7	C	D	A	B
8	D	C	B	A

Table C.2: Environment Ordering for Each Participant in Peg-in-Hole Experiment Based on Latin Square Design Method

Subject	Order			
	1	2	3	4
1	A	B	C	D
2	B	A	D	C
3	C	D	A	B
4	D	C	B	A
5	A	B	C	D
6	B	A	D	C
7	C	D	A	B
8	D	C	B	A

Table C.3: Environment Ordering for Each Participant in Train Track Assembly Experiment Based on Latin Square Design Method

Subject	Order				
	1	2	3	4	5
1	E	A	B	C	D
2	A	B	E	D	C
3	D	C	A	B	E
4	C	E	D	A	B
5	B	D	C	E	A
6	E	A	B	C	D
7	A	B	E	D	C
8	D	C	A	B	E
9	C	E	D	A	B
10	B	D	C	E	A
11	E	A	B	C	D
12	A	B	E	D	C
13	D	C	A	B	E
14	C	E	D	A	B
15	B	D	C	E	A

Table C.4: Environment Ordering for Each Participant in Brake Pad Replacement Experiment Based on Latin Square Design Method

Subject	Order			
	1	2	3	4
1	A	B	C	D
2	B	A	D	C
3	C	D	A	B
4	D	C	B	A
5	A	B	C	D
6	B	A	D	C
7	C	D	A	B
8	D	C	B	A
9	A	B	C	D
10	B	A	D	C
11	C	D	A	B
12	D	C	B	A

C.3 VALIDATING THE APPROPRIATENESS OF STATISTICAL MODELS USED

After analysis of the experimental results, care must be taken to ensure that each model provided an appropriate fit for the data analyzed. For analysis using the 2-proportion hypothesis test, the only major assumption is that the response variables analyzed are binary, which was satisfied, so no significant analysis of the appropriateness of the model is necessary. Similarly, for the 2-sample t-test, the only major assumption in using the model is that the response variable is normally distributed. The only time this statistical analysis technique was used during this dissertation, it followed analysis of variance on the same data. As will be discussed below, analyzing the appropriateness of the ANOVA model ensured that the response variables were normally distributed, and, therefore, satisfy the requirements for the 2-sample t-test.

Ensuring the integrity of the binary logistic regression model required a slightly different approach. Firstly, all data analyzed using this model was binary, which is clearly a prerequisite. To ensure that the model provided an accurate fit to the actual response data, expected observations from the model were compared to actual data. For all binary logistic regression analyses performed, these values compared closely, signifying a satisfactory fit of the data to the model.

The statistical method utilized most during this dissertation and requiring the most attention to ensure integrity of the model for the data was analysis of variance. The most common technique used to analyze the appropriateness of the data in an ANOVA model, and that used in this dissertation, is analysis of the residuals. Here, the residual refers to the difference between the expected value at a given set of treatment levels and the actual data values.

The basic assumption in the ANOVA model is that the response variable being analyzed is normally distributed. The integrity of this model can be checked by an analysis of the residuals. Within MINITAB, several residual plots can be created at the completion of ANOVA calculations. These plots were utilized to determine appropriateness of the model for the data in all experiments. A description of the four residual plots used is below.

- In the *normal probability plot*, the residual is plotted against its expected value under normality. If the plot lies nearly on a straight line, the residuals are normal, which is appropriate for the model. However, if the plot differs substantially from linearity, then the error distribution is not normal.
- The *histogram plot of the residuals* gives a graphical representation of the distribution of the residuals. For appropriateness, the residual distribution should be as close to normal as possible.
- When plotting the *residual versus the fitted values* of the response variable, the residuals should display a random distribution around zero, and remain constant for all fitted values (constant variance).
- A *sequence plot* shows the residuals versus the order of the data. In the experiments conducted for this dissertation, this plot is less meaningful since multiple participants were used for data collection. Any change in variability seen in this plot is mostly attributed to the performance difference between each participant.

To provide an example of the residual analysis procedure undertaken for each experiment, the ANOVA results for piece grasp and assembly time in the train track experiment are discussed.

During the train track experiment, the total track piece time, including grasp and assembly, was analyzed for its dependence on type of visualization and availability of haptic feedback. Initially, a two-way ANOVA was carried out comparing this piece time with these two predictor variables. In Figure C.1, the residual plots are shown for this initial ANOVA are shown. In the normal probability plot (upper-left), the residuals are highly non-linear. Also, a plot of the residuals versus the fitted values (upper-right) shows that the variance of error terms is not constant across treatments. Lastly, the histogram of the residuals (lower-left) clearly shows non-normality in the error distribution as there is an extended tail on the positive side of the plot. All of the plots discussed in Figure C.1 suggest that the timing data being analyzed is not normally distributed, and a transformation should be applied. For the type of non-linearity and non-constancy of variance seen in these plots, typical transformations include taking the square root or natural log of the response variable prior to subsequent analysis.

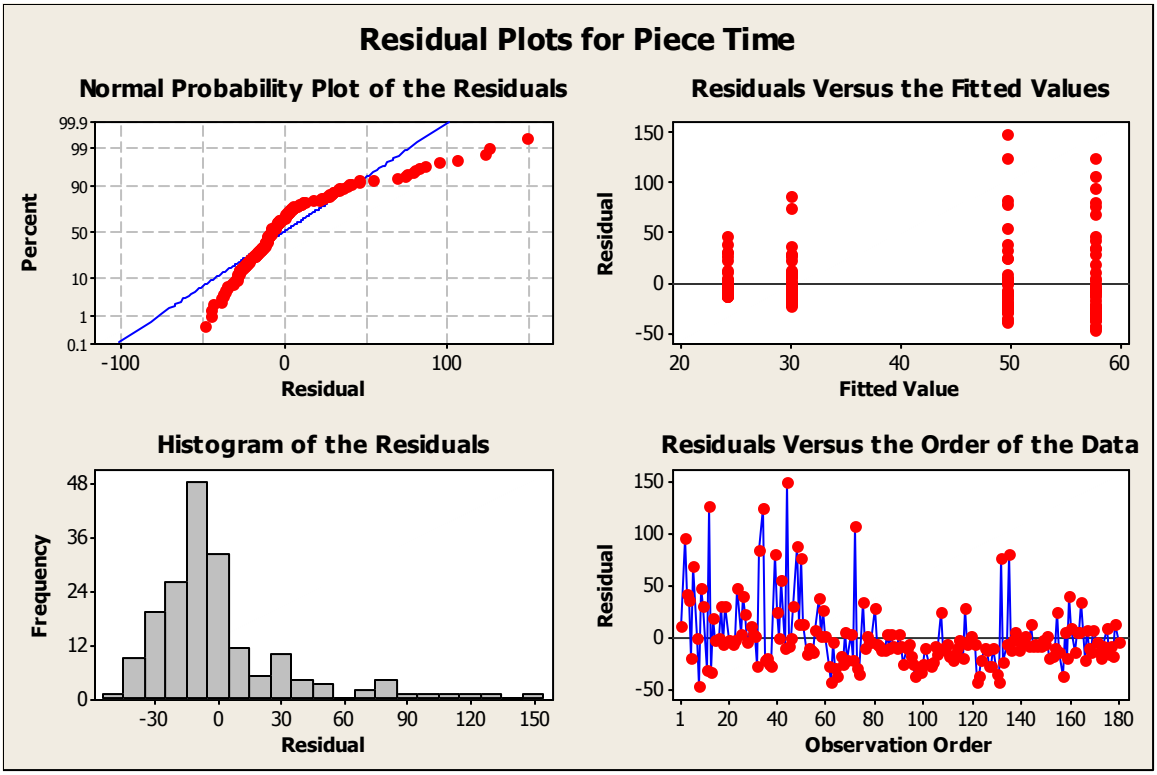


Figure C.1: Residual Analysis for ANOVA Results Comparing Piece Time to Type of Visualization and Availability of Haptic Feedback

As a next step, the same data was analyzed again, comparing the square root of piece time to the control variables. The residual plots for this ANOVA are shown in Figure C.2. As can be seen, the data in the normal probability plot follows a slightly more linear pattern. Also, the differences in the error variance are reduced in the plot of residuals versus the fitted values. Lastly, the histogram plot of the shows that the residuals are close to normal, but a small tail still exists.

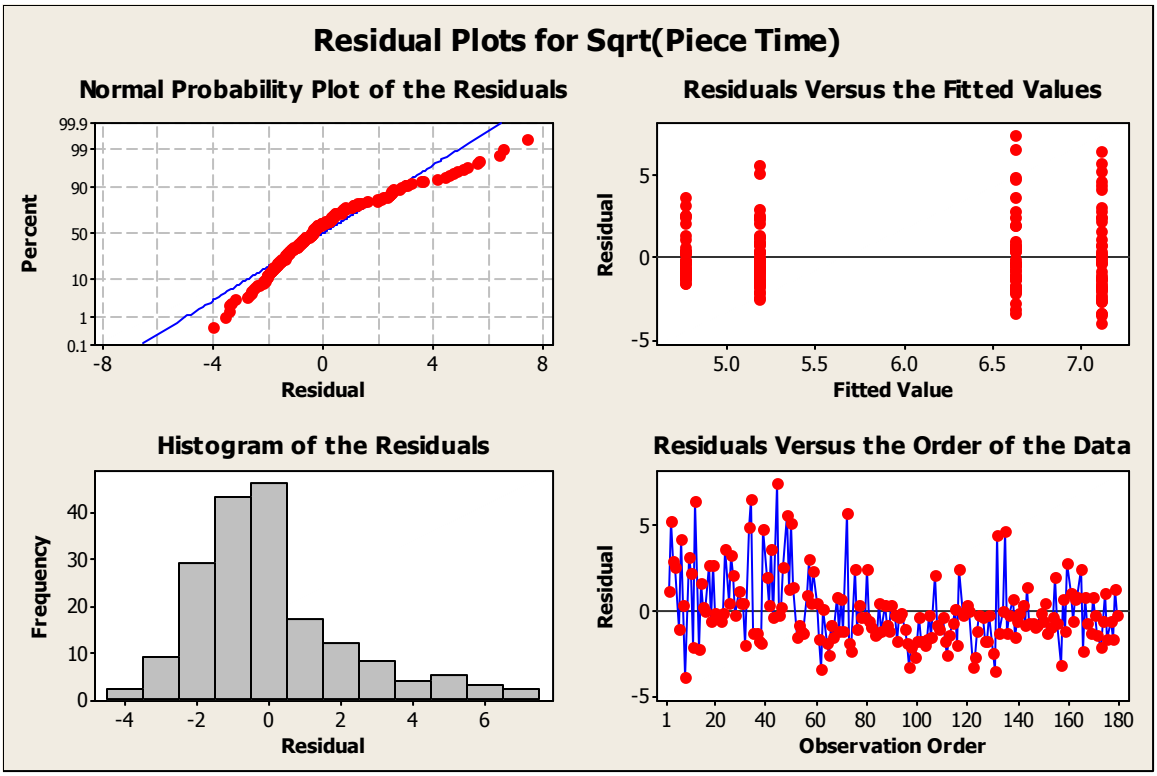


Figure C.2: Residual Analysis for ANOVA Results Comparing Sqrt(Piece Time) to Type of Visualization and Availability of Haptic Feedback

The data was analyzed one more time using ANOVA, comparing the natural log of piece time to the control variables. The residual plots for this analysis are shown in Figure C.3. In the normal probability plot for this analysis, the error terms are highly linear with their expected values. Also, the variance of the error terms is more constant across experimental treatments. Although not perfect, the histogram of the residuals represents a more normal distribution, as there is no noticeable tail in either direction, and the data peaks in the middle.

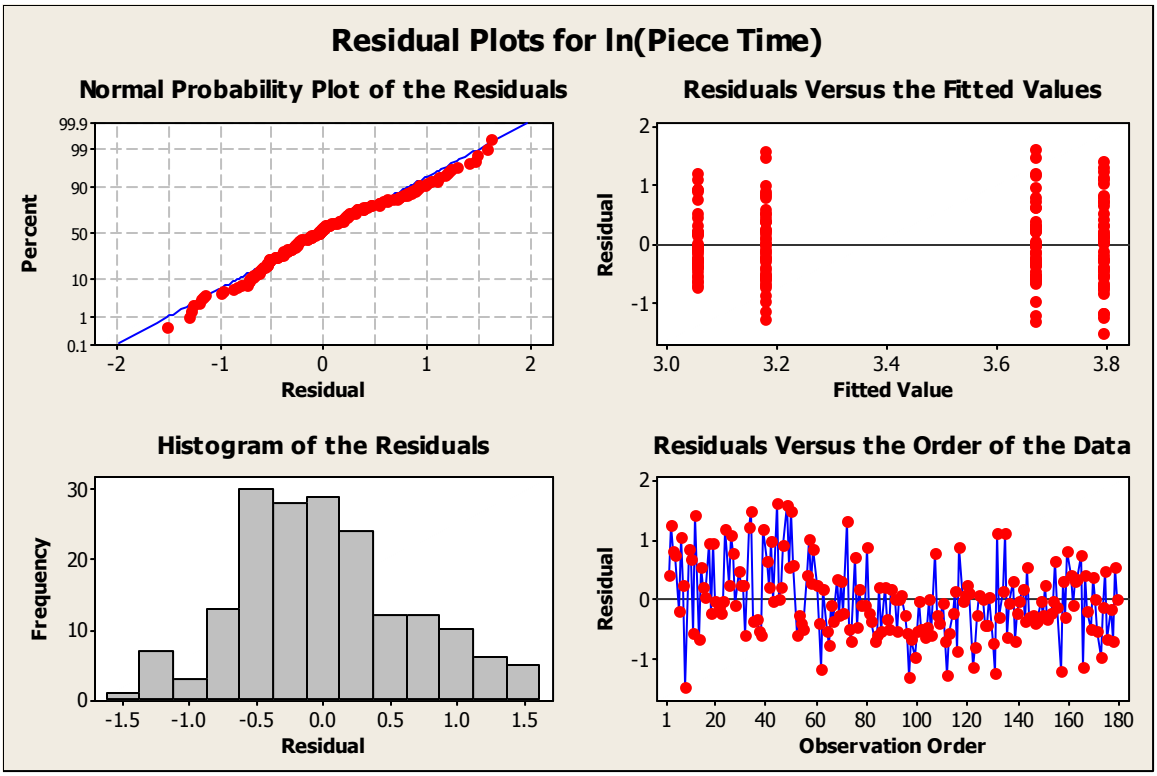


Figure C.3: Residual Analysis for ANOVA Results Comparing ln(Piece Time) to Type of Visualization and Availability of Haptic Feedback

The analysis of residuals for the ANOVA comparing piece time with the control variables in the train track experiment resulted in a transformation of the response variable prior to analysis. This transformation was necessary to insure the appropriateness of the ANOVA model for the data given, namely that the response variable is normally distributed. In this instance, analysis of the residuals has shown that a natural log transformation on piece time is best. In fact, the natural log turned out to be the most suitable transformation for all ANOVA models in this dissertation using time as a response variable. The reason for this is that the timing data collected can never be less than zero but can reach high levels. This results in the distribution of the timing data having an elongated tail along its right side, similar to the histogram of the residuals in Figure C.1. Since this phenomenon is similar across all experiments, it was expected that

the natural log transformation would be appropriate for all ANOVA calculations, although the validity of the analysis procedure was checked for each experiment separately.

REFERENCES

Adams, J. D. and D. E. Whitney (1999). *Application of Screw Theory to Constraint Analysis of Assemblies of Rigid Parts*. Proceedings of the 1999 IEEE International Symposium on Assembly and Task Planning, Porto, Portugal.

Antonishek, B., D. D. Egts and U. R. Obeysekare (1998). *Virtual Assembly Using Two-Handed Interaction Techniques on the Virtual Workbench*. Proceedings of the 1998 ASME Design Engineering Technical Conference, Atlanta, GA.

Askins, S. A. and W. J. Book (2003). *Digital Clay: User Interaction Model for Control of a Fluidically Actuated Haptics Device*. Sim2003 1st International Conference on Computational Methods in Fluid Power Technology, Melbourne, Australia.

Baraff, D. (1989). *Analytical Methods for Dynamic Simulation of Non-penetrating Rigid Bodies*. SIGGRAPH 89, Boston, MA.

Baraff, D. (1991). *Coping with Friction for Non-penetrating Rigid Body Simulation*. SIGGRAPH 91, Las Vegas, NV.

Baraff, D. (1994). *Fast Contact Force Computation for Nonpenetrating Rigid Bodies*. SIGGRAPH 94, Orlando, FL.

Baumann, R. and R. Clavel (1998). *Haptic interface for virtual reality based minimally invasive surgery simulation*. Proceedings of the 1998 IEEE International Conference on Robotics and Automation. Part 1, Leuven, Belgium.

Burdea, G. (2000). *Haptics Issues in Virtual Environments*. CGI 2000: The 18th Computer Graphics International 'Humans and Nature' Conference, Geneve, Switzerland.

Burdea, G., G. Patounakis, V. Popescu and R. E. Weiss (1998). *Virtual reality training for the diagnosis of prostate cancer*. 1998 IEEE Virtual Reality Annual International Symposium, Atlanta, GA.

Cameron, S. A. (1997). *Enhancing GJK: Computing Minimum and Penetration Distances betweeen Convex Polyhedra*. Int. Conf. Robotics and Automation, Albuquerque, NM.

Chang, B. and E. J. Colgate (1997). *Real-time Impulse-based Simulation of Rigid Body Systems for Haptic Display*. ASME International Mechanical Engineering Congress and Exhibition, Dallas, TX.

Cohen, J. D., M. C. Lin, D. Manocha and M. K. Ponamgi (1996). I-COLLIDE: An Interactve and Exact Collision Detection System for Large-Scale Environments. Chapel Hill, NC, University of North Carolina.

Colwell, C., H. Petrie, D. Kornbrot, A. Hardwick and S. Furner (1998). *Haptic virtual reality for blind computer users*. Proceedings of the 1998 3rd International ACM Conference on Assistive Technologies, Marina del Rey, CA.

Coutee, A. S., S. D. McDermott and B. Bras (2001). "A Haptic Assembly and Disassembly Simulation Environment and Associated Computation Load Optimization Techniques." *ASME Journal of Computing and Information Science* **1**(2): 113-122.

Dawson, S. L. and J. A. Kaufman (1998). "Imperative for medical simulation." *Proceedings of the IEEE* **86**(3): 479-483.

Devore, J. L. (1995). Probability and Statistics for Engineering and the Sciences. Belmont, Duxbury Press.

Dureck, L., N. J. Macias, D. M. Weinstein, C. R. Johnson and J. M. Hollerbach (1998). *SCIRun Haptic Display for Scientific Visualization*. PHANToM Users Group Meeting, Dedham, MA.

Ehmann, S. A. and M. C. Lin (2001). *Accelerated Proximity Queries Between Convex Polyhedra by Multi-Level Voronoi Marching*. Proceedings of the International Conference on Intelligent Robots and Systems.

Ehmann, S. A. and M. C. Lin (2001). *Accurate and Fast Proximity Queries Between Polyhedra Using Convex Surface Decomposition*. Proceedings of Eurographics: Computer Graphics Forum.

Fernando, T., P. Wimalaratne and K. Tan (1999). *Constraint-Based Virtual Environment for Supporting Assembly and Maintainability Tasks*. Proceedings of the 1999 ASME Design Engineering Technical Conference: Computers in Engineering, Las Vegas, NV.

Fernando, T., P. Wimalaratne, K. Tan and N. Murray (1999). "Interactive Product Simulation Environment for Assessing Assembly and Maintainability Tasks." *ASME Industrial Virtual Reality: Manufacturing and Design Tool for the Next Millenium* **5**: 179-189.

Ganter, M. A. and B. P. Isarankura (1993). "Dynamic Collision Detection Using Space Partitioning." *Journal of Mechanical Design* **115**(1): 150-155.

Gilbert, E. G., D. W. Johnson and S. A. Keerthi (1998). "A Fast Procedure for Computing the Distance Between Complex Objects in Three-Dimensional Space." *IEEE Transactions Robotics and Automation* **4**(2): 193-203.

Gillespie, R. B. and J. E. Colgate (1997). *A Survey of Multibody Dynamics for Virtual Environments*. 1997 ASME International Mechanical Engineering Congress and Exposition, Dallas, TX.

Gomes de Sa, A. and G. Zachmann (1999). "Virtual Reality as a Tool for Verification of Assembly and Maintenance Processes." *Computers & Graphics* **23**: 389-403.

Gottschalk, S., M. C. Lin and D. Manocha (1996). *OBBTree: A Hierarchical Structure for Rapid Interference Detection*. Proceedings of ACME Siggraph.

Gregory, A., M. C. Lin, S. Gottschalk and R. Taylor (1999). *A Framework for Fast and Accurate Collision Detection for Haptic Interaction*. IEEE Virtual Reality, Houston, TX.

Gupta, R., D. Whitney and D. Zeltzer (1997). "Prototyping and Design for Assembly Analysis using Multimodal Virtual Environments." *Computer Aided Design (Special issue on VR in CAD)* **29**(8): 585-597.

Gupta, R. and D. Zeltzer (1995). *Prototyping and Design for Assembly Analysis using Multimodal Virtual Environments*. Proceedings of ASME Computers in Engineering Conference and the Engineering Database Symposium, Boston, MA.

Gurocak, H., S. Jayaram, B. Parrish and U. Jayaram (2003). "Weight Sensation in Virtual Environments Using a Haptic Device with Air Jets." *ASME Journal of Computing and Information Science in Engineering* **3**(2): 130-135.

Gutierrez, T., J. I. Barbero, M. Aizpitarte, A. R. Carrillo and A. Eguidazu (1998). *Assembly Simulation Through Haptic Virtual Prototypes*. Third PHANToM Users Group Workshop, Dedham, MA.

Hahn, J. K. (1988). "Realistic Animation of Rigid Bodies." *Computer Graphics* **22**(4): 299-308.

Hardwick, A., S. Furner and J. Rush (1998). "Tactile display of virtual reality from the World Wide Web - a potential access method for blind people." *Displays* **18**(3): 153-161.

Hayter, A. J. (1996). Probability and Statistics for Engineers and Scientists. Boston, MA, PWS Publishing Company.

He, T. (1999). *Fast Collision Detection using QuOSPO Trees*. 1999 Symposium on Interactive 3D Graphics, Atlanta, GA.

Hellstrom, A. (1985). "The time-order error and its relatives: Mirrors of cognitive processes in comparing." *Psychological Bulletin* **97**: 35-61.

Hubbard, P. (1996). "Appoximating Polyhedra with Spheres for Time-Critical Collision Detection." *ACM Transactions on Graphics* **15**(3): 179-210.

Hudson, T. C., M. C. Lin, J. Cohen, S. Gottschalk and D. Manocha (1997). V-COLLIDE: Accelerated Collision Detection for VRML. Chapel Hill, NC, University of North Carolina.

Jayaram, S., U. Jayaram, Y. Wang, H. Tirumali, K. Lyons and P. Hart (1999). "VADE: A Virtual Assembly Design Environment." *IEEE Computer Graphics and Applications* **19**(6): 44-50.

Jayaram, S., Y. Wang, U. Jayaram, K. Lyons and P. Hart (1999). *Virtual Assembly Design Environment*. 1999 IEEE Virtual Reality International Symposium, Houston, TX.

Johnson, D. E. and E. Cohen (1998). *A Framework for Efficient Minimum Distance Computations*. IEEE International Conference on Robotics and Automation, Leuven, Belgium.

Johnson, D. E. and E. Cohen (1999). *Bound Coherence for Minimum Distance Computations*. 1999 International Conference on Robotics and Automation, Detroit, MI.

Johnson, T. C. and J. M. Vance (2001). *The Use of the Voxmap PointShell Method of Collision Detection in Virtual Assembly Methods Planning*. ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Pittsburgh, PA.

Joskowicz, L. and E. Sacks (1998). "Computer-Aided Mechanical Design Using Configuration Spaces."

Kim, H., H. Ko, K. Lee and C. Lee (1997). *A Collision Detection for Interactive Mechanical Assembly Simulation*. 1997 IEEE International Symposium on Assembly and Task Planning, Marina del Ray, CA.

Klosowski, J. T., M. Held, J. S. B. Mithcell, H. Sowizral and K. Zikan (1998). "Efficient Collision Detection Using Bounding Volume Hierarchies of k-DOPs." *IEEE Transactions on Visualization and Computer Grahpics* **4**(1): 21-36.

Krishnan, S., A. Pattekar, M. C. Lin and D. Manocha (1998). *Spherical Shell: A Higher Order Bounding Volume for Fast Proximity Queries*. Proceedings of WAFR.

Kuehne, R. and J. Oliver (1995). *A Virtual Environment for Interactive Assembly Planning and Evaluation*. Proceedings of ASME Design Automation Conference, Boston, MA.

Lin, M. C. and J. F. Canny (1991). *A Fast Algorithm for Incremental Distance Calculation*. IEEE International Conference on Robotics and Automation, Sacramento, CA.

Lin, M. C., J. F. Canny and D. Manocha (1993). Fast Collision Detection between Geometric Models. Chapel Hill, Depratment of Computer Science, University of North Carolina at Chapel Hill.

Lin, M. C. and S. Gottschalk (1998). *Collision Detection Between Geometric Models: A Survey*. IMA Conference on Mathematics of Surfaces.

Massie, T. H. (1996). Initial Haptic Explorations with the Phantom: Virtual Touch Through Point Interaction. Master of Science, Massachusetts Institute of Technology, Cambridge.

McDermott, S. D. (1999). Development of a Haptically Enabled Disassembly Simulation Environment. Master of Science in Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA.

McDermott, S. D. and B. Bras (1999). *Development of a Haptically Enabled Dis/Re-Assembly Simulation Environment*. Proceedings of the 1999 ASME Design Engineering Technical Conference, Las Vegas, NV.

McNeely, W. A., K. D. Puterbaugh and J. J. Troy (1999). *Six Degree-of-Freedom Haptic Rendering Using Voxel Sampling*. Proceedings of Siggraph, Los Angeles, CA.

Mirtich, B. (1995). *Hybrid Simulation: Combining Constraints and Impulses*. First Workshop on Simulation and Interaction in Virtual Environments, Berkeley, CA, University of California, Berkeley.

Mirtich, B. (1996). Impulse-based Dynamic Simulation of Rigid Body Systems. Ph.D., University of California at Berkeley, Berkeley, CA.

Mirtich, B. (1997). *V-Clip*: Fast and Robust Polyhedral Collision Detection. Cambridge, MA, Mitsubishi Electric Research Laboratory.

Mirtich, B. (1998). Rigid Body Contact: Collision Detection to Force Computation. Cambridge, MA, Mitsubishi Electric Research Laboratory.

Mirtich, B. and J. Canny (1995). *Impulse-based Dynamic Simulation*. The Algorithmic Foundations of Robotics, Boston, MA.

Mirtich, B. and J. Canny (1995). *Impulse-based Simulation of Rigid Bodies*. Symposium on Interactive 3D Graphics, New York, NY.

Moore, M. P. and J. Wilhelms (1988). "Collision Detection and Response for Computer Animation." *Computer Graphics* **22**(4): 289-298.

M'Sirdi, N. K. (1993). *Application of robust control for elastic joint robots*. 1993 ASME Winter Annual Meeting, New Orleans, LA.

Neves, N., J. P. Silva, P. Goncalves, J. Muchaxo, J. M. Silva and A. Camara (1997). "Cognitive spaces and metaphors: A Solution for Interaction with Spatial Data." *Computers & Geosciences* **23**(4): 483-388.

Neter, J., M. H. Kutner, C. J. Nachtsheim and W. Wasserman (1996). Applied Linear Statistical Models. Boston, MA, McGraw-Hill.

Pere, E., D. Gomez, N. Langrana and G. Burdea (1996). *Virtual Mechanical Assembly on a PC-based System*. ASEM Design Engineering Technical Conference and Computers in Engineering Conference, Irvine, CA.

Rosen, D., A. Nguyen and H. Wang (2003). *On the Geometry of Low Degree-of-Freedom Digital Clay Human-Computer Interface Devices*. ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Chicago, IL.

SensAble Technologies, I., Ed. (2000). GHOST SDK Programmer's Guide. Cambridge, MA, SensAble Technologies, Inc.

Sheffield, J., T. F. Chan, R. V. Dubey and R. L. Kress (1993). *Evaluation of position and orientation increment methods for teleoperation*. 1993 ASME Winter Annual Meeting, New Orleans, LA.

Siddique, Z. and D. W. Rosen (1997). "A Virtual Prototyping Approach to Product Disassembly Reasoning." *Computer-Aided Design* **29**(12): 847-860.

Springer, S. L. and R. Gadh (1997). *Haptic Feedback for Virtual Reality Computer Aided Design*. 1997 ASME International Mechanical Engineering Congress and Exposition: Concurrent Product Design and Environmentally Conscious Manufacturing, Dallas, TX.

Srinivasan, H., N. Shyamsundar and R. Gadh (1997). *A Virtual Disassembly Tool to Support Environmentally Concious Product Design*. 1997 IEEE International

Symposium on Electronics and the Environment, Life Cycle Envrionmental Stewardship for Electronic Products, San Fransisco, CA.

Srinivasan, M. A. and C. Basdogan (1997). "Haptics in Virtual Environments: Taxonomy, Research Status, and Challenges." *Computers & Graphics* **21**(4): 393-404.

Stewart, P., Y. Chen and P. Buttolo (1997). *Direct integration of haptic user interface in CAD systems*. 1997 ASME International Mechanical Engineering Congress and Exposition, Dallas, TX.

Thomas, F. and C. Torras (1988). *Constraint-Based Inference of Assembly Configurations*. IEEE International Conference on Robotics and Automation, Philadelphia, PA.

Volkov, S. A. and J. M. Vance (2001). *Effectiveness of Haptic Sensation for the Evaluation of Virtual Prototypes*. ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference, Pittsburgh, PA.

Wang, Y., U. Jayaram, S. Jayaram and K. Lyons (2001). *Physically Based Modeling in Virtual Assembly*. ASME Computers in Engineering Conference, Pittsburgh, PA.

VITA

Adam Coutee was born in Willoughby, Lake County, Ohio on February 12, 1976. He grew up in Ohio, Texas, and Florida where he attended Niceville High School, Niceville, FL. He earned a Bachelor of Mechanical Engineering from Georgia Institute of Technology (Atlanta, GA) in 1999, during which he worked as a cooperative education student for General Research Corporation in Shalimar, FL. He earned a Master of Science in Mechanical Engineering from Georgia Institute of Technology in 2002. His graduate work was funded by National Science Foundation Grant DMI-9908335 and a National Science Foundation Graduate Research Fellowship. Upon graduation, Adam has accepted a position as a Research Engineer with ExxonMobil Corporation in Houston, TX.