A Haptic Stencil for Manufacturing Applications

by

Kirti Ramesh Mansukhani

B. Eng., Mechanical Engineering
Bayero University Kano, Nigeria 1999

Submitted to the Department of Mechanical Engineering in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2004

Author.....
Department of Mechanical Engineering
January 20th, 2004

Certified by.....
Sanjay E. Sarma
Associate Professor
Thesis Supervisor

Accepted by.....
Ain A. Sonin
Chairman, Department Committee on Graduate Students

A Haptic Stencil for Manufacturing Applications

by

Kirti Ramesh Mansukhani

Submitted to the Department of Mechanical Engineering on January 20th, 2004 in Partial Fulfillment of the Requirements for the Degree of Master of Science in Mechanical Engineering.

Abstract

The haptic stencil consists of a 5 DOF haptic device and an anti-collision algorithm that acts as a geometric stencil and can be used for a variety of applications ranging from training to rapid prototyping and manufacturing. Online manipulation of a three-axis desktop milling machine was established using this setup.

This work describes the algorithm design used to achieve the required performance and stencil-like behavior with specific reference to machining applications. Some of the primary aspects of this design include the collision detection, collision remediation and control methodologies employed. The parameters on which these methodologies depended and how they were developed are the focus of this work.

Collision detection is the core of any haptic interaction as it determines whether or not contact has been established between the virtual objects and therefore is essential in deciding the appropriate haptic feedback. In the case of the haptic stencil, the collision detection algorithm would have to identify whether or not contact occurs between the haptic probe-controlled tool object and the stationary part object.

Collision remediation provides the stencil-like behavior by enforcing geometric constraints on the regions/surfaces by preventing penetration by the tool object. The results from the collision detection and collision remediation modules are used to control the desktop milling machine which cuts out a copy of the part object used in the haptic simulation from a given stock according to the motions specified on the haptic probe by the operator.

Speed control is necessary in order to ensure that motions from the human operator are not lost due to the different communication speeds between the various modules of this setup. Speed control also helps in providing as 'real-time' a machining experience as possible for a given part and stock combination.

Thesis Supervisor: Sanjay E. Sarma

Title: Professor of Mechanical Engineering

This thesis is dedicated to my parents.

Without you, I wouldn't have been,

Without your love and care, I wouldn't have been who I am,

Without your faith and support, I wouldn't have come this far.

Thank you for being there in every way you can.

I love you always.

Acknowledgements

I wish to thank my parents, sisters and Jiju for always being there for me – I am lucky to have you all.

I give great thanks to my friends at MIT who have made being here memorable and meaningful. I wish to give special thanks to Suraj Deshmukh and Sriganesh Lokanathan for their support and encouragement.

To Professor Sanjay Sarma for taking the time out to discuss this work and provide insight and encouragement. Thank you for giving me this opportunity. I would also like to acknowledge the contributions of Adachi Yoshitaka and Suzuki Motor Corporation (Japan) for developing the haptic device, Stephen Ho for his work on full body collision detection and Dave Downs at The Motion Group for his help with implementing their controller.

To my cousin, Kusum Ramchandani and the entire Ramchandani family for welcoming me into your lives and making me feel at home.

To my nieces and nephew, Samaa, Sarina and Dinesh – your toothless grins and innocent smiles have brought many a smile to my face.

And finally to the dream close to my heart that has made my life complete and without whom this work would never have been.

Table of Contents

Table of Figures

Chapter 1: INTRODUCTION

The term haptic originates from the Greek word *haptesthai* and means "*relating to or based on the sense of touch*". Although the word has been a part of the English language since 1890, its development as a field is only recent.

A haptic interface is a force-reflecting device that allows a user to touch, feel, manipulate, create, and/or alter simulated objects in a virtual environment. In this manner, haptics provides an additional mode of communication between a user and a virtual environment.

There are two main areas of application and research being developed by the haptics community - Virtual Reality/Tele-robotic systems and Force Feedback/Tactile Display systems. Virtual reality and tele-robotic researchers seek to develop technologies that will allow the simulation or mirroring of virtual or remote forces. While virtual reality and similar systems seek to convey vector force information, the researchers in the field of force feedback and tactile displays seek to develop a method of conveying more subtle sensory information between humans and machines. Being able to convey information via the sense of touch allows for many applications ranging from virtual reality gaming gear to tele-surgery and tele-manufacturing. This work presents one such application – the haptic stencil.

Section 1.1 details the milestones in the development of haptics as a field. Section 1.2 describes the objective and scope of this work, while section 1.3 provides an outline for the rest of this paper.

1.1 Haptics – A History

Credit for envisioning simulated haptic interaction goes to the well-known pioneer of Computer Graphics, Ivan Sutherland, who in his 1965 article "The Ultimate Display", envisioned feeling forces at the surfaces of graphically displayed objects and beyond, using mechanical force-feedback devices. At the conclusion of his article, he states,

> "The ultimate display would, of course, be a room within which the computer can control the existence of matter. A chair displayed in such a room would be good enough to sit in."
>
> [Sutherland 1965]

It must be noted however, that specialized devices to provide virtual haptic feedback can be traced back to research in the early 1950s aimed at developing and improving tele-robotic systems. In such systems, the operator controls a "master" arm that transmits his / her commands to a remote slave. Sheridan [Sheridan 1992b] defines the master-slave tele-operator system as having two subsystems: the master device which is typically a multi DOF mechanical device positioned directly by the human operator, and the slave device which is usually equipped with an end effector (a hand for grasping, or a specialized tool to perform some specialized task).

These systems are particularly suited to applications which involved harmful / unfriendly environments (such as nuclear, outer-space, or underwater sites). Initially, due to the purely mechanical design of the linkages between the master and the slave, the slave was not truly "remote". However, as newer electrical servomechanisms were developed, forces could be applied to the user's hand grasping the master based on feedback signals received from sensors in the slave, giving the user the impression of manipulating the remote environment directly.

Haptics in its own right as a field started attracting a lot of interest after Sutherland's vision. Frederick Brooks, Jr. and his colleagues at North Carolina Chapel Hill were so

inspired by Sutherland's vision that in 1967, they project GROPE - real-time simulation of three-dimensional molecular docking forces was started. After a number of versions of the GROPE and more than 20 years, faster computing hardware allowed Brooks and his colleagues to reach their original goal. [Burdea 1996]

During the same period, another group of researchers at the Cornell Aeronautical Laboratory, and later at General Electric constructed an exoskeleton master. The device accepted input from the user from both arms and legs. This master was placed inside a larger exoskeleton slave used to amplify the power of the user. This device, called "Hardiman" consisted of 30 hydraulically powered servo joints[1]. Control issues at normal operating speeds and the potential hazards of leaks in the hydraulics were the primary limitations of the Hardiman. Eventually, researchers at the University of Utah Center for Engineering refined hydraulic exoskeleton technology to produce the Utah-NOSC Arm[2].

Jones and Thousand, who in 1966 patented one of the first dexterous master manipulators, developed a much simpler and safer design using pneumatic bladders. As opposed to the previous Argonne Arm or Hardiman, which provided feedback to the user's wrist, a dexterous master measures user hand commands (usually through sensing gloves) and provides feedback to independent fingers. A pressure proportional to the measured position error (between the user and robot hands when object was grasped) was used to inflate pneumatic bladder actuators placed in the palm of the master. As a result, the user felt as if he was grasping the object directly. Twenty-five years later a similar concept was used by researchers at Advanced Robotics Research Ltd., in the

---

[1] B. Makinson, "Research and Development prototype for machine augmentation of human strength and endurance: Hardiman I project ", Technical Report S-71-1056, General Electric Co., Schenectady, NY, May 1971

[2] S. Jacobsen, E. Iversen, C. Davis, D. Poter, and T. McLain, "Design of a multiple degree-of-freedom, force-reflective hand master/slave with a high mobility wrist", Proceedings of ANS/IEEE/SMC 3rd Topical Meeting on Robotics and Remote Systems, IEEE, NY, March 1989

United Kingdom for their Teletact I and Teletact II gloves[3, 4]. A data glove with tactile feedback is used for outputs to the user, whereas a second data glove is used for inputs to the computer. The input glove is equipped with 20 pressure sensors, and the output glove with 20 air pads, controlled by 20 pneumatic pumps. The next generation, Teletact-II, has been equipped with 30 air pads and was commercialized.[5]

A more complex dexterous master with feedback to each finger phalange was subsequently patented by Zarudiansky[6]. His design uses a rigid external shell and an inner glove worn by the user. Combinations of actuators housed in the external shell connected to several rings that are attached to the user's inner glove provide force feedback to the fingers as well as to the palm and wrist. [Burdea 1996]

During the mid and late 1980s, different exoskeletons (master arms for tele-operation) were being developed at the Jet Propulsion Laboratories (JPL), the EXOS Company, National Aeronautics and Space Administration (NASA) and the University of Utah. The systems used a kind of master-slave combination, and forces were applied by motors at the joints. Unfortunately, these devices are usually very heavy and could therefore only be used for specialized applications. The advance brought by the Salisbury/JPL arm[7,8] was the introduction of computer-based Cartesian control, allowing the master to be more compact and to be able to tele-operate slaves with different kinematic configurations.

---

[3] R. Stone, "Advanced Human-System Interfaces for Tele-robotics using Virtual Reality and Telepresence technologies", Proceedings of the 5th Intl. Conf. on Advanced Robotics (ICAR), IEEE, pp 168-173, 1991

[4] R. Stone, "Virtual Reality Tutorial", MICAD Conference, Micado, Paris, France, 1992

[5] http://hwr.nici.kun.nl/~miami/taxonomy/node30.html

[6] A. Zarudiiansky, "Remote Handling Devices", U. S. Patent 4, 392, 138, November 24, 1981

[7] A. Bejczy, K. Salisbury, "Kinematic Coupling between operator and remote manipulator", Advances in Computer Technology, Vol. 1, ASME, NY, pp. 197-211, 1980

[8] B. Hannaford, L. Wood, B. Guggisberg, D. McAffee, H. Zak, "Performance evaluation of a 6-axis generalized force-reflecting teleoperator", JPL Publication 89-18, California Inst. of Technology, Pasadena, CA 1989

Most of the above masters were developed originally for tele-robotic applications and not to serve as I/O devices for VR, a field that appeared in the late 1970s. Researchers then started to develop special-purpose tactile/force feedback hardware. One of the first prototypes to provide tactile feedback from a graphics simulation was the "Sandpaper" system developed at the Massachusetts Institute of Technology Media Laboratory [Minsky 1990]. The prototype consisted of a two-degree-of-freedom joystick with large electrical actuators in an enclosure placed by the computer. The high bandwidth of the feedback loop (500 to 1000 Hz) allowed for both force and tactile feedback in a single simple and easy to use device. It was thus possible to move a cursor over various samples of virtual sandpaper and feel their surface texture. Inertia and damping were also modeled in a two-dimensional simulation. In 1992, dial with force feedback (Force Dial) was used by computer scientists and chemists for simple molecular modeling tasks[9]. The force is controlled by a motor. The main advantage of this device is its low price and its robustness but due to its simplicity it will not be very useful for a multimodal system.

Certain applications required the masters to be portable and light in order to allow the user natural freedom of motion rather than being constrained to a desktop system to provide tactile and force feedback. An example of such a device is the Rutgers Master, developed at the Rutgers University CAIP Center in 1992. The Rutgers Master used four pneumatic micro-actuators placed in the palm to give users the feel of the hardness of virtual objects being manipulated [Burdea 1996].

The first commercial systems designed for virtual I/O became available at the end of 1993 through the introduction of the Touch Master and SAFIRE Master. These were followed by the recent introduction of low-cost masters such as the PHANToM Arm

---

[9] M. Good, "Participatory Design of A Portable Torque-Feedback Device", in P. Bauersfeld, J. Bennett, and G. Lynch, editors, *Proc. of the Conf. on Human Factors in Computing Systems, CHI'92*, pages 439--446. ACM/SIGCHI, 1992.

[Massie and Salisbury 1994] and the Impulse Engine [Immersion Corporation]. With these new devices, developers have the tools to complement the visual and sound feedback created by earlier I/O devices. [Burdea 1996]

A concise listing of key developments in device development in haptics can be found in Minsky 1995.

Applications as well as devices are being researched in the field of tactile and force feedback. Haptics for manufacturing has been an active area of research at the Rapid Autonomous Manufacturing Laboratory (RAMLAB) at MIT. Mahadevan Balasubramaniam [Balasubramaniam 2000] worked on tool path generation software using a 5 DOF haptic device. Stephen Ho [Ho 1999] and Sriram Krishnan [Sriram 2001] worked on full-body collision detection methodologies for haptic applications. Edmund Golaski [Golaski 2001] did some early work in the direct haptic control of a machine tool. Ho and Sriram's work are discussed further in Chapter 2 – Literature review, and Golaski's work is introduced in Chapter 4 – Collision remediation.

Mahadevan [Balasubramaniam *et al* 2002] developed an intuitive man-machine interface for generating 5-axis tool paths. The system is based on a 5 degree-of-freedom force feedback haptic system, which is used to interface a human with an impenetrable 3D part. In the process of feeling the object, the user 'teaches' a milling machine to machine a virtual 3D object. The tool path generation has two phases: recording of access directions at the surface of the object and the post-processing phase. During the recording phase, three functions are carried out simultaneously: first, a fast collision detection algorithm, using hierarchical object representation, to drive the haptic system; second, visual feedback to show the regions that have been accessed by the tool; and third, a system to capture the access directions of the tool as the user touches the object. The post-processing phase involves the use of information generated in the recording phase to

generate 5-axis tool paths. First, the access directions at the surface of the part are interpolated; and second, any residual collisions are detected and eliminated.

1.2 Problem Statement

The aim of this project was to implement a 3-dimensional geometric stencil for manufacturing purposes using haptic feedback. Online manipulation of a three-axis desktop milling machine tool was established using a haptic device (developed by Suzuki Japan) via a computer. The haptic stencil functions in a manner analogous to a conventional stencil used for drawing purposes. The differences lie in the conceptualization and dimensional space. A conventional stencil is by definition two-dimensional and enforces constraints on the user by specifying physical boundaries. The haptic stencil on the other hand is three dimensional in the sense that it specifies constraints for arbitrarily shaped objects (rather than regions on planes/surfaces) and enforces constraints by specifying "virtual" geometric boundaries. How these boundaries are defined and constraints imposed are the core of this research work.

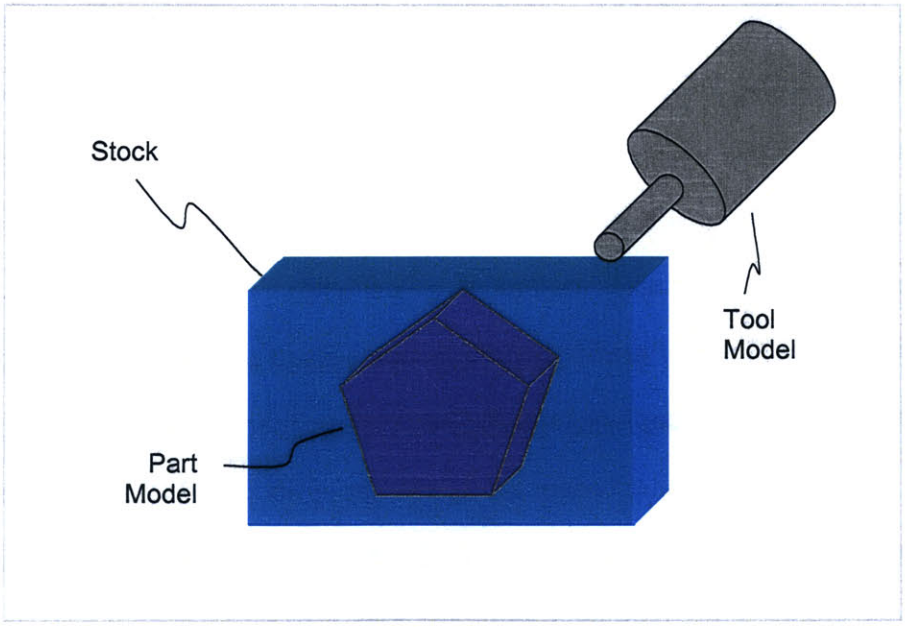

Figure 1-1: Stencil Operation

The operation of the haptic stencil is illustrated us Figure 1-1. The hexagonal *part model* shown would behave like a stencil embedded in the stock. The tool would therefore be able to cut through the stock (shown as the light material surrounding the dark hexagonal solid) but would not be able to machine through the part model due to the stencil-like behavior of the interface. Any time the real tool attempts to penetrate the part model a resistive force is sent through the haptic device to the operator and a non-penetrative position is specified for the tool. In this way, the 3D haptic stencil ensures correct machining.

1.3 Motivation

In general, haptic feedback applications are being developed in a wide variety of industries including medicine, entertainment, tele-robotics and the military. The interface discussed in this work is one such application, and has been specifically designed for manufacturing applications.

A primary motivation of this work is to investigate the applicability of haptic manufacturing. Presently, CNC machining has few limitations with regard to the manufacturing industry. However, as parts become less conventionally shaped and product designers in various industries/fields experiment with extremely intricate geometries and materials, the demand for "part sculpting" (rather than part machining) interfaces is gradually increasing. Sensable Technologies has already developed sculpting software (FreeForm) that is being used by CAD/CAM designers (especially in the toy industry) in conjunction with Sensable's Phantom [Sensable].

Although the setup presented in this paper addresses similar applications, it utilizes a significantly different – embedded 3D geometric stencil – approach. Furthermore, the haptic stencil implementation goes one step further in aiding the designer by establishing online manipulation of a milling machine thus rendering the setup highly appropriate for

rapid prototyping. The present implementation would also be suitable for training applications such as training students in machining or dental training.

1.4 Chapter Outline

The next chapter is dedicated to the terminology used in this work and a brief summary of conceptual models relevant to the implementation of the haptic stencil. Chapter 3 discusses collision detection – the core of any haptic simulation, and the specific algorithm employed for the haptic stencil. Chapters 4 and 5 are dedicated to the primary intellectual contributions of this work – Collision remediation and Machine tool control. The overall implementation and operation of the system is described in chapter 6. This work concludes with a brief discussion of performance of the setup, alternatives to certain methods used, modifications for other applications, critical challenges faced in the development of and suggestions for future work on the Haptic Stencil.

Chapter 2: LITERATURE REVIEW AND TERMINOLOGY

Due to the fact that haptics is still a nascent technology, research in this area has been primarily focused on defining and testing rather than developing. Key terms are still being coined, characteristic parameters defined and limitations being discovered. This chapter aims to summarize some of the terminology that has been standardized and conceptual models that have been developed and provide a deep and meaningful insight into the primary aspects of haptic technologies. Some of the applications being researched and/ or commercialized have also been discussed in the final section.

2.1 Terminology

2.1.1 Haptic Interface Design Terminology

- Haptic Display: A mechanical device configured to convey kinesthetic cues to a human operator. Haptic displays can be broadly classified into two categories, namely:
    - *Impedance Displays*: These *measure motion and display force.* As a result, they are usually designed to have a low inertia and be highly back-drivable. Examples include the Phantom family of haptic displays [Massie & Salisbury1994], the McGill University Pantograph, and the University of Washington pen-based force display.
    - *Admittance Displays*: These *measure force and display motion.* Admittance displays are therefore designed to have high inertia, and are typically non back-drivable manipulators fitted with force sensors and driven by position/velocity control loops. An example is Carnegie Mellon

University's WYSIWYF Display [Yokokohji et al 1996] which is based on the PUMA 560 industrial robot.

- *Haptic Interface*: This includes everything that comes between the human operator and the virtual environment. This includes the haptic device, control software, and analog-to-digital/digital-to-analog conversion. The haptic interface characterizes the exchange of energy between the operator and the virtual world and thus is important for both stability and performance analysis (Adams 1999).
- *Virtual Environment*: Computer generated model of some physically motivated scene. Complexity of virtual worlds could range from simple computer games to complex high-fidelity flight simulations. Despite the wide range of complexity and applications, it has been found that there are only two fundamental ways in which a virtual model/environment can interact with the haptic interface. The two approaches are briefly described below:
    - *Penalty-based Approach*: In this approach, the environment accepts velocities (or positions) and generates forces according to some physical model (e.g. stiff spring model). This kind of interaction is popularly termed an Impedance Interaction.
    - *Constraint-based Approach*: Here, the environment accepts forces and returns velocities (or positions). This kind of interaction is also termed Admittance Interaction.
- *Haptic Simulation*: This is the synthesis of human operator, haptic interface, and virtual environment that creates a kinesthetically immersive experience. (Adams 1999)

- *Causality structures*[10]*:* choice of haptic display type (impedance or admittance) and virtual environment type (impedance or admittance), of which there are four possibilities. The causality structure of the haptic stencil setup is discussed in the System Implementation chapter of this work (chapter 6).

Sheridan introduced the concept of Afferent and Efferent channels.

  - *Afferent channels* convey information (forces, positions) from the system to the user.
  - *Efferent channels* on the other hand, convey information from the user to the system.

- *Computer Haptics*[11]*:* generation and haptic rendering of virtual objects.
- *Collision Detection:* determines whether or not two geometric entities have a non-empty intersection.
- *B-Rep:* Boundary representation. A solid model representation where the geometric primitives on the boundary is used to represent the solid.
- *BSP:* Binary Space Partition. A scheme of dividing *n*-dimensional space in binary fashion using primitives in *n-1* dimensions.
- *Convexity:* A simple polygon is convex if, given any two points on its boundary or in its interior; all points on the line segment drawn on the line segment between them are contained in the polygon's boundary or interior.
- *Voronoi diagram:* The Voronoi diagram of a finite point set $S \subseteq R^d$ subdivides R^d into regions such that for each point a in S, there is a Voronoi cell V_a such that every point in V_a is at least as close to a as any other point in S.

---

[10] Adams 1999

[11] Basdogan & Srinivasan 1997

2.1.2 Modeling Techniques / Model Representations

Geometric modeling is one of the key factors that determine algorithm design. There are many types of model representations used in CAD/CAM and computer graphics. The taxonomy presented here has been adapted from Lin & Gottschalk 1998.

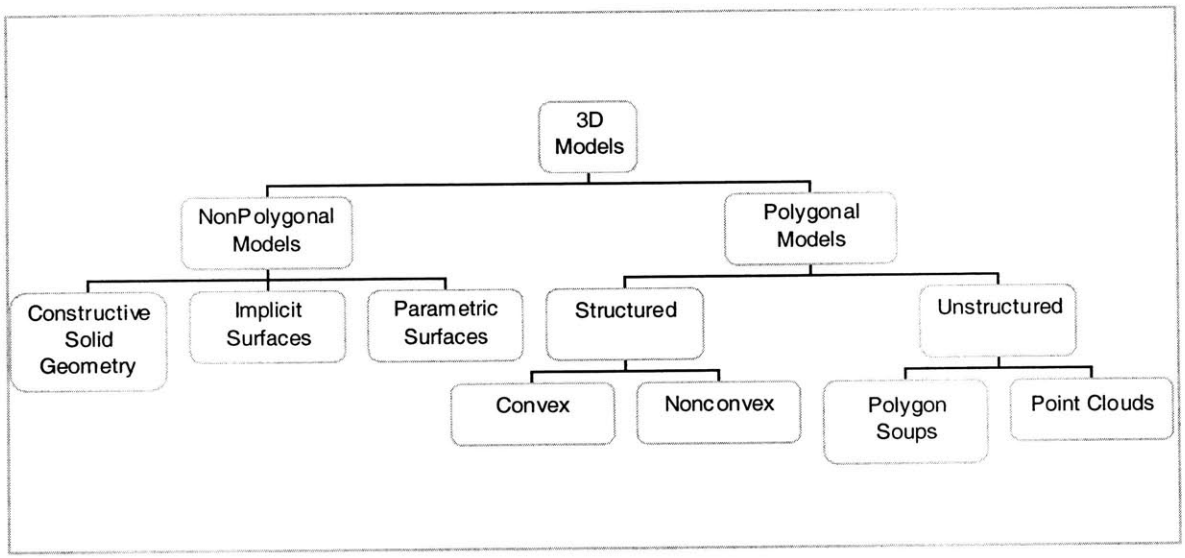

Figure 2-1: Geometric Models, Taxonomy

2.1.2.1 Polygonal models

Polygonal objects are the most commonly used models in computer graphics and modeling. They have a simple representation and are versatile.

The most general class of a polygonal model is a "*polygon soup*", which consists of a collection of polygons that are not geometrically connected and has no topology information available. Point clouds are similar to polygon soups with the only difference that the planar polygon is replaced by a point.

If the polygons form a closed manifold, then the model has a well-defined inside and outside – it is a proper solid. This is, therefore, a structured representation. If the object

defined by the closed manifold is *convex*, then this additional structure can be exploited in collision detection algorithms.

2.1.2.2 Non-polygonal models:

In Constructive Solid Geometry (CSG), simple primitives such as blocks, spheres, cylinders, cones are combined using regularized Boolean set operations to form complex objects. The object is stored as a tree with operators at the internal nodes and simple primitives at the leaves (Figure 2-2). Some nodes represent Boolean operators, whereas others perform translation, rotation, and scaling. The primary strength of CSG is that it enables an intuitive design process of building shapes by means of cutting (intersection and set difference) and joining (union) simple shapes to form more complex ones. On the other hand, however, simple operations such as rounding an edge or filleting a joint are difficult to describe with CSG.

Implicit surfaces are defined using implicit functions. An implicit equation is any algebraic equation defined on a set of variables, of the form f ( ) = 0. Implicit equation defined objects are described in further detail in Chapter 3 since the tool in the haptic stencil is modeled using this approach.

Parametric surfaces are continuous surfaces defined by coordinates of points on them, which themselves are functions of a parameter. They are mappings from some subset of the plane to space, $f : R^2 \mapsto R^3$. Unlike implicit surfaces, parametric surfaces are not generally closed manifolds. As a result, they do not represent a complete solid model, but rather a description of a surface boundary. Common types of parametric surfaces include Bezier surfaces, B-spline surfaces and rational parametric surfaces like NURBS (Non-Uniform Rational B-Spline).

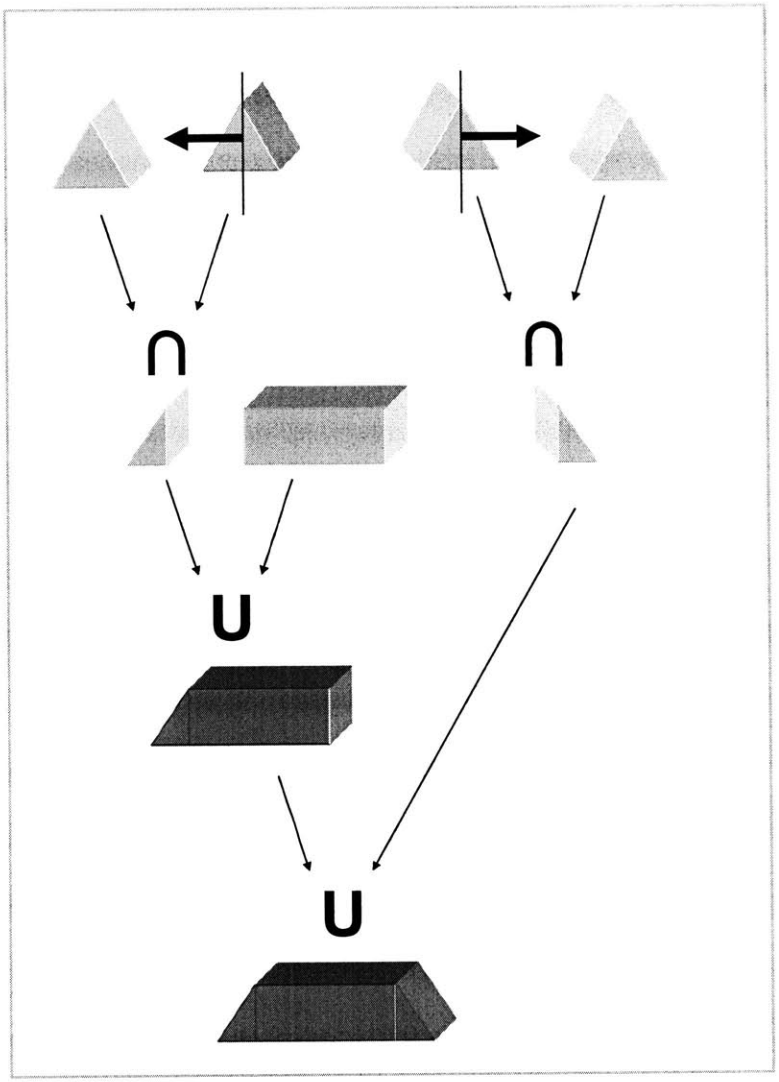

Figure 2-2: Constructive Solid Geometry model of a trapezoidal slab

2.2 History of Collision Detection Algorithms

"Whenever two objects attempt to interpenetrate each other, we call it a collision"

-Moore and Wilhelms [1988]

Collision detection for static objects has been known for a number of years. Cyrus and Beck[12] developed an algorithm that detects the interaction of two- and three-

[12] M. Cyrus, J. Beck, "Generalilzed two- and three-dimensional clipping", Computers and Graphics, Vol. 3, No. 1, pp. 23 – 28, 1978.

dimensional lines against arbitrary convex polyhedrons. It used the dot product between the normal to an edge and the vector from one of the vertices of that edge and the point of interest. Penetration has occurred if this dot product is negative.

Later, the Cyrus-Beck algorithm for static objects was extended to collision detection for computer animation (Moore and Wilhelms 1988). Here, the trajectory of the object of interest is known a priori and parameterized as a function of time. The objects are then checked for collision at fixed time intervals. For this algorithm, therefore, the size of the time interval determines the accuracy of collision detection. If the time interval is large, collision may go undetected. Also, in haptic simulations, the trajectory of at least one of the moving objects is determined by the user through a haptic device (e.g. force feedback enabled mouse or joystick) and is therefore random. As a result, algorithms that require a priori knowledge of object trajectories are not particularly suited to haptic applications.

Due to the computational complexity and the high bandwidth requirements of haptic feedback systems, haptic systems have been limited in their ability to simulate contact interactions. The computation of force feedback must be completed approximately 1000 times per second [Massie 96] in order for the haptic system to realistically re-create the feeling of contact situations. Early haptic demonstrations had been limited to computationally simple tasks such as simulating the contact interaction between a point and a plane. The problem with single entity interaction is that it lacks true solid representation, as can be seen from the following observation by Massie:

> "... one's hand can physically pass through the volume occupied by a virtual sphere, while only the finger tip is constrained to remain outside the virtual sphere."
>
> [Massie 1996]

Developments in haptic systems led to more useful contact simulation situations such as point to convex object contact, ray contact with a convex union [Basdogan 1997], and point to NURBS surfaces [Hollerbach 1997]. Although, the ray representation is an

improvement over the point, it is still not a true solid representation. Furthermore, decomposition of solids into convex components is difficult, and some objects cannot be decomposed into convex sub-units.

In order to meet the haptic servo requirements for more complex geometric models, a variety of efficiency improving approaches have been proposed over time. Two that have gained a lot of popularity and have proven to be computationally very efficient are the use of hierarchical representations for object models, and spatial partitioning. For the haptic stencil, both these approaches have proven very useful and will be discussed in further detail in Chapter 3. Other approaches that have also been proposed and used in literature include geometric reasoning, algebraic formulations, and optimization methods. Using hierarchical representations, Dobkin 1990 developed an algorithm for polytope-polytope overlap problem.

When positioning accuracy of the collision and interpenetration locations is desired, an exact collision detection algorithm is needed. The performance of such algorithms usually degrades with the complexity of the virtual scene. Lin and Canny [Lin 1993, Lin & Canny 1995] developed a fast algorithm that has almost constant performance versus the number of object vertices. The algorithm uses *local features* (such as vertices, edges or facets) of convex polyhedra and is extendable to concave ones (with the additional complexity however). Cohen 1995 (I-COLLIDE) extended Lin's local collision detection algorithm to multi-body collision detection for VR simulations. To have both speed and accuracy, the researchers took a two-step approach consisting of an approximate (gross) collision detection followed by an exact collision detection where needed. In I-COLLIDE, the simulation loop starts with a pruning of multi-body pairs where overlapping bounding boxes are detected. These are then subject to pair-wise exact collision detection. When collision is detected, then an analysis is performed involving object dynamics, interpenetration forces, and so on, and an appropriate collision response is generated.

However, the objects of the collision detection are limited to convex polyhedra (or concave ones that can be decomposed into convex sub-units). Mirtch improved upon the Lin-Canny approach with the V-Clip collision detection method, which enhances the robustness of the Lin-Canny approach and adds penetration depth information [Mirtich 1998].

Another exact collision detection method was proposed by Schlaroff & Pentland 1991 that uses implicit functions to describe the object models rather than polygonal representations.

As stated in Chapter 1, researchers at the Rapid Autonomous Manufacturing Laboratory at MIT have developed rapid interference detection and analysis methodologies for arbitrarily shaped objects. Stephen [Ho 1999] described an algorithm and representations for rapidly detecting and correcting collision between a manually pre-defined tool and an arbitrary work piece. He computed collision and correction information at the rate of 1000 times a second, making it possible to perform force control for haptics using the collision detection algorithm in a real-time loop. Ho's collision detection methodology has been employed for the haptic stencil setup.

Sriram [Krishnan 2001] extended Ho's work to rapid collision detection between arbitrary shapes or non-convex polyhedra. Based on real-world applications, the algorithm is restricted to shallow penetration between solids. The solids in the simulation are modeled as boundary representations. Bounding volume hierarchies are used to improve the efficiency of the algorithm. Proximity is determined by computing the Voronoi diagram in the solids, and the proposed method incorporates Voronoi specification both implicitly and explicitly.

Lin & Gottschalk 1998, Jimenez 2000, and Sriram 2001 provide a detailed survey of the development of collision detection and related algorithms.

2.3 Contemporary/ Related Work

Sensable Technologies has already developed sculpting software that is being used by CAD/CAM designers (especially in the toy industry) in conjunction with Sensable's Phantom haptic device/hardware. However, the Sensable platform does not provide direct control of the machine tool. [Sensable]

This work describes an embedded 3D geometric stencil. The haptic stencil enables online manipulation of a milling machine while preventing over cuts and gouges that a human is likely to make. In every sense, this is a stencil, except in 3D. Beyond rapid prototyping and artistic machining, this approach can be used to train dentists, for example.

The following subsections describe some of the contemporary research areas and industrial applications for haptic feedback technology.

2.3.1 Research

Research in the Haptics community can be primarily divided into Human haptics and Machine haptics. Human haptics is involved with understanding the mechanics (biomechanics) of touch and developing paradigms for simulating/ modeling touch. Touch, in this context encompasses texture, temperature, shape, geometry and stiffness. This area of research also covers Tactile Neurophysiology, Human Perception, and Motor action.[13] Machine haptics, on the other hand, deals with the development of the hardware required to simulate touch and the software development for simulated worlds.

For the haptic stencil, we are dealing with collisions between rigid bodies and are primarily interested in contact forces. Significant amount of research is focused at developing collision detection algorithms for non-rigid bodies and simulating properties

[13] MIT Touch Lab: touchlab.mit.edu

other than contact forces (e.g. vibrations, shock, elastic deformations, plastic deformations, texture).

A detailed listing of all the institutions and corporations involved with research in haptic feedback technologies can be found in Burdea 1996.

2.3.2 Industrial Applications

In general, the applications for haptic feedback being developed in industry today include medical devices/ simulations, human-machine interfaces and training. Human-machine interfaces are being developed for a variety of industries ranging from medical to military. Some of these have been briefly presented in this section.

Information regarding some of the commercial devices/ interfaces for haptic feedback applications has been obtained from company websites such as Immersion Corporation, Sensable technologies, Logitech[14] and Industrial Virtual Reality Inc.[15].

2.3.2.1 Medical

Immersion Medical designs, manufactures and markets computer-based medical simulators that allow healthcare personnel to practice minimally invasive procedures without placing patients at risk. These include the Vascular Access Simulator, the Endoscopy Simulator, and the Endovascular simulator. Immersion also provides enabling technologies to leading developers of virtual-reality medical and scientific simulations such as the Virtual Laparoscopic Interface, the Laparoscopic Impulse Engine® and the Surgical Workstation (in partnership with Medtronic).

---

[14] www.logitech.com

[15] www.ivri.com

Tissue palpitation, anesthesia, minimally invasive surgery and microsurgery training are some other areas in which incorporating haptic feedback is being investigated. Various types of rehabilitation devices incorporating haptic feedback are also being developed. Hogan and colleagues at MIT [Hogan 1993] developed the "MIT-MANUS" workstation for manual therapy and training. In the robot-aided therapy, a person sitting at a table puts the lower arm and wrist into a brace attached to the arm of the robot. A video screen prompts the person to perform an arm exercise such as connecting the dots. If movement does not occur, MIT-Manus moves the person's arm. If the person starts to move on his own, the robot provides adjustable levels of guidance and assistance to facilitate the person's arm movement.

2.3.2.2 Gaming and Consumer Electronics

In the VR and games industry, force feedback enabled joysticks and mice are being developed by companies like Logitech and Immersion Technologies.

In the gaming and consumer electronics markets, Immersion licenses its haptics technology to computer peripheral and software companies to add realistic touch sensations to the computing and gaming experience. Peripherals with Immersion technology include devices such as joysticks, game pads, steering wheels, and mice (in partnership with Logitech).

2.3.2.3 Robotics

Robotics and Automation is another area where human-machine interfaces with haptics are being developed for a variety of applications ranging from tele-operations in unfriendly environments to robot path planning and training.

In robot motion planning and control, collision detection helps to steer the robot away from its surrounding obstacles. The NASA rovers are a classic application.

2.3.2.4 Manufacturing & CAD/CAM

In manufacturing, CAD/CAM designers in the toy industry for example, use haptic interfaces developed by companies like Sensable Technologies Inc.

In the 3D Digitizing market, Immersion sells products based on leading technologies for accurate, affordable 3D digitizing. Immersion's MicroScribe™ products capture the physical properties of three-dimensional objects and accurately translate them into complete 3D models on-screen. In the 3D Interaction market, Immersion develops whole-hand, force-feedback 3D-interaction technologies that allow the user to "reach in" and physically interact with simulated computer content for the manufacturing design and the consumer markets. The 3D Interaction product offerings include: CyberForce®, "desktop" whole-hand and arm force-feedback; CyberGlove®, fully instrumented glove; and VirtualHand® Studio software tools.

Immersion is pioneering a revolutionary new approach to automotive design by embedding touch sensations in control devices such as knobs and dials. The result is a streamlined driver interface with minimal dashboard and steering column clutter, a solution both aesthetically pleasing and less visually distracting (in partnership with BMW).

In virtual prototyping, haptic feedback is used to refine designs without productions of physical prototypes in the initial design stages. Sensable technologies' Freeform & Phantom package is one such example widely used in the toy industry.

2.3.2.5 Research/ Academia

Immersion produces high-fidelity products suitable for researching the science of touch. Whether you're feeling the contact forces of a protein-DNA interface, remotely

operating a robot, or interacting with a microscopic surface, Immersion's Impulse Engine® 2000 delivers crisp, clean haptic sensations.

In MEMS, research shows opportunity for haptics-based interfaces in micro-assembly. In fact, an alternative application that was considered for the haptic stencil was pick-and-place type micro-assembly. Using this interface for micro-assembly operations would have the advantage of deterministic assembly and also enable the user to feel micro-scale forces [16]. However, with the industry driving towards large-scale micro-assembly methodologies (using fluidics or vibration, for example) to exploit the economies of scale these methodologies provide (usually at the cost of not being deterministic), applications requiring pick-and-place type assembly operations are limited.

Molecular docking and other molecular dynamic simulations is another field where haptics interfaces are being implemented for academic research purposes. [Sankaranarayanan 2003]

---

[16] Sitti 1999

Chapter 3: COLLISION DETECTION

A key factor for a convincing haptic interaction is the input to the haptic interface, which depends on both simulation modeling and control algorithms. This chapter discusses the simulation modeling approach used in this work, while chapters 4 and 5 discuss the control algorithm. Because today's haptic interfaces do not have imbedded intelligence, they rely entirely on real-time computer input. Without such input, the most sophisticated haptic feedback interface would be useless. Thus, physical modeling that generates the variables (forces, accelerations, vibrations, temperatures, etc.), is necessary for haptic interface control. Collision detection is the first step in the physical modeling of the virtual world.

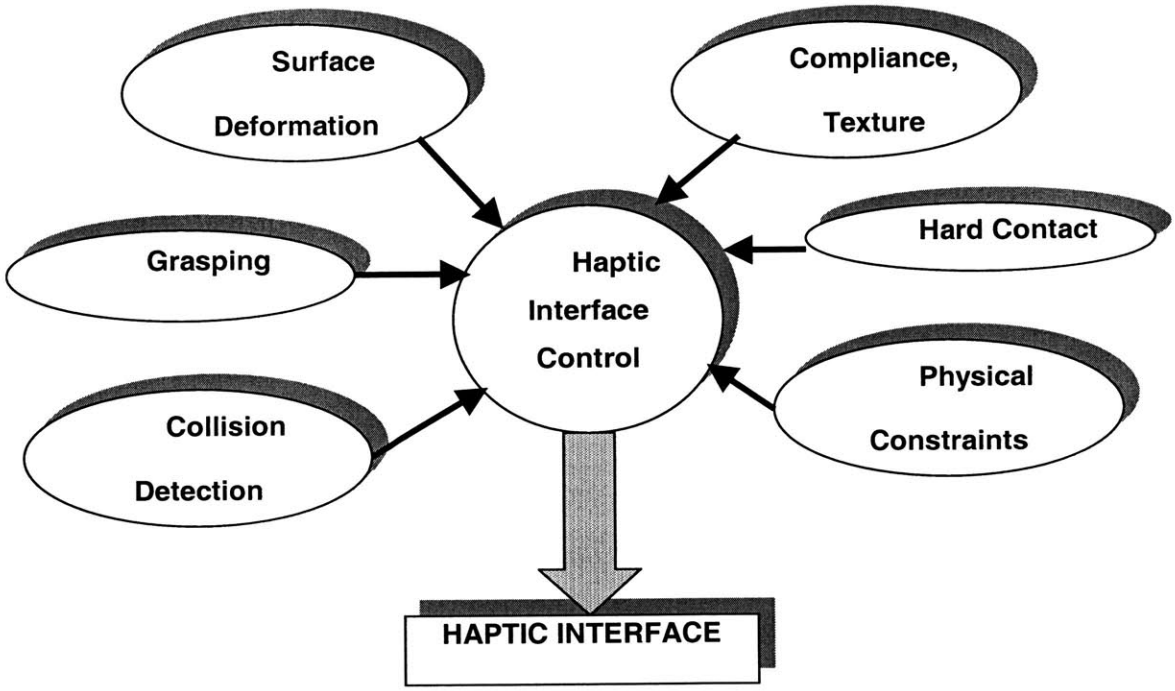

Figure 3-1: Physical Modeling of Virtual Worlds [adapted from Burdea 1996]

Collision detection (also referred to as interference detection or contact determination) is the core of any haptic interaction as it determines whether or not contact has been established between the virtual objects and therefore is essential in deciding the appropriate haptic feedback. The problem is encountered in computer-aided design and manufacturing (CAD/CAM), robotics and automation (motion path planning), computer graphics and animation, and computer simulated environments. It is useful for any task that involves contact analysis and spatial reasoning among static or moving objects in a simulated environment.

In the case of the haptic stencil, the collision detection algorithm would have to identify whether or not contact occurs between the haptic probe-controlled tool object and the stationary part object. The collision response if contact is detected is the subject of the next chapter. A detailed review of the collision detection algorithm employed for the haptic stencil has been included here due to the fact the choice of the collision response (collision remediation methodology) depended on the collision detection methodology.

3.1 Collision Detection Methodology

3.1.1 Requirements

To enable a haptic system to represent the reaction forces involved between two three-dimensional objects, the underlying collision detection algorithm must be able to detect contact between objects and formulate an appropriate force response. The collision detection method must not be limited in the types of solids that may participate; both convex and non-convex objects should be applicable. Furthermore, the force response must be representative of the reaction force created by the contact (including both rotational and translational forces for 3D interaction), and the collision detection and force response computations must be completed within 1 millisecond [Massie 1996].

One of the reasons that haptic technology is so difficult to pull off convincingly is that the human sense of touch is far more demanding than our senses of sight or sound. A motion picture need only play at 30 frames per second to trick the eye into thinking that it's watching real life. A monitor only has to refresh 70 times or so per second for us not to perceive a flicker in the display. On the other hand, many touch sensations require actuators to be updated every 1000 times per second to realistically recreate the feeling of contact situations.

To drive full body haptic responses in the device, the governing program must detect when the interacting solids move and/or collide, determine the extent of the collision, and determine the proper response to the collision. These three stages are essential for a realistic haptic simulation.

There are many approaches to addressing the collision detection problem. Three main factors contribute to the quality of a particular methodology: speed, accuracy, and wealth of information. Methods that are very good at one factor tend to be less satisfactory at the others.

3.2 Stencil Collision Detection Methodology

The collision detection methodology employed in the haptic stencil is the one developed by Stephen Ho at the Rapid Autonomous Manufacturing Laboratory, MIT [Ho 1999]. The reason for using Ho's methodology over Sriram's [Sriram 2001] is that even though the latter is a more generic approach and therefore applicable to a larger variety of geometries, the former is particularly suited for collision detection between an axi-symmetric body and an arbitrarily shaped object. The axi-symmetric assumption and the fact that the haptic device used in Ho's methodology is a 5 DOF device make it very suitable for the haptic stencil setup.

This section describes the primary features of Ho's collision detection methodology and its application to the haptic stencil.

3.2.1 Data Structures

Due to the fact that the stencil setup follows an impedance-impedance causality structure, we are primarily interested in displacements and determining the forces as a result of these displacements. Also, since our aim is to control a milling machine tool, both the tool and part need to be modeled as infinitely stiff objects. This places stringent constraints on the modeling. The data structures would not only need to accurately and completely define the two solid representations but would also need to display high stiffness. Any voids or deformations would lead to inaccuracies.

The data structures used in the stencil collision detection methodology are:

- Implicit equation BSP tree to represent the tool object.
- Point cloud - triangulated surface combination representation to represent the part object.

3.2.1.1 Implicit Equation Representation:

$$f(x) = 0 \tag{3-1}$$

An implicit equation representation uses an equation such as equation (3.1) to represent the geometry of the object. As an example, consider the implicit equation for a unit sphere:

$$x^2 + y^2 + z^2 - 1 = 0 \tag{3-2}$$

Points satisfying the implicit equation lie on the surface. Points to one side evaluate to a negative number while points on the other side return a positive value. In this way interior information for a solid model is available. For complex geometries, multiple implicit

equations are used together through AND (intersection) and OR (union) combination of equations.

3.2.1.2 Binary Space Partition Tree (BSP Tree):

This divides space into regions using the implicit equations of the surfaces. At the root of the tree, space has not been divided; as you progress downwards the implicit equation of the surfaces become nodes and the results of their inequalities become the child nodes. Each of the leaves is then assessed and designated as either being part of the interior or exterior of the solid. Note that, for simplicity, surface boundary of the object is considered part of the interior.

The stencil collision detection methodology uses an Implicit Equation BSP tree to represent the tool object. Although it is time consuming to come up with the implicit equation representation for an object, it is balanced by the fact that the same tool will be used in conjunction with various part objects and therefore is a one-time effort for a model that will be repeatedly used.

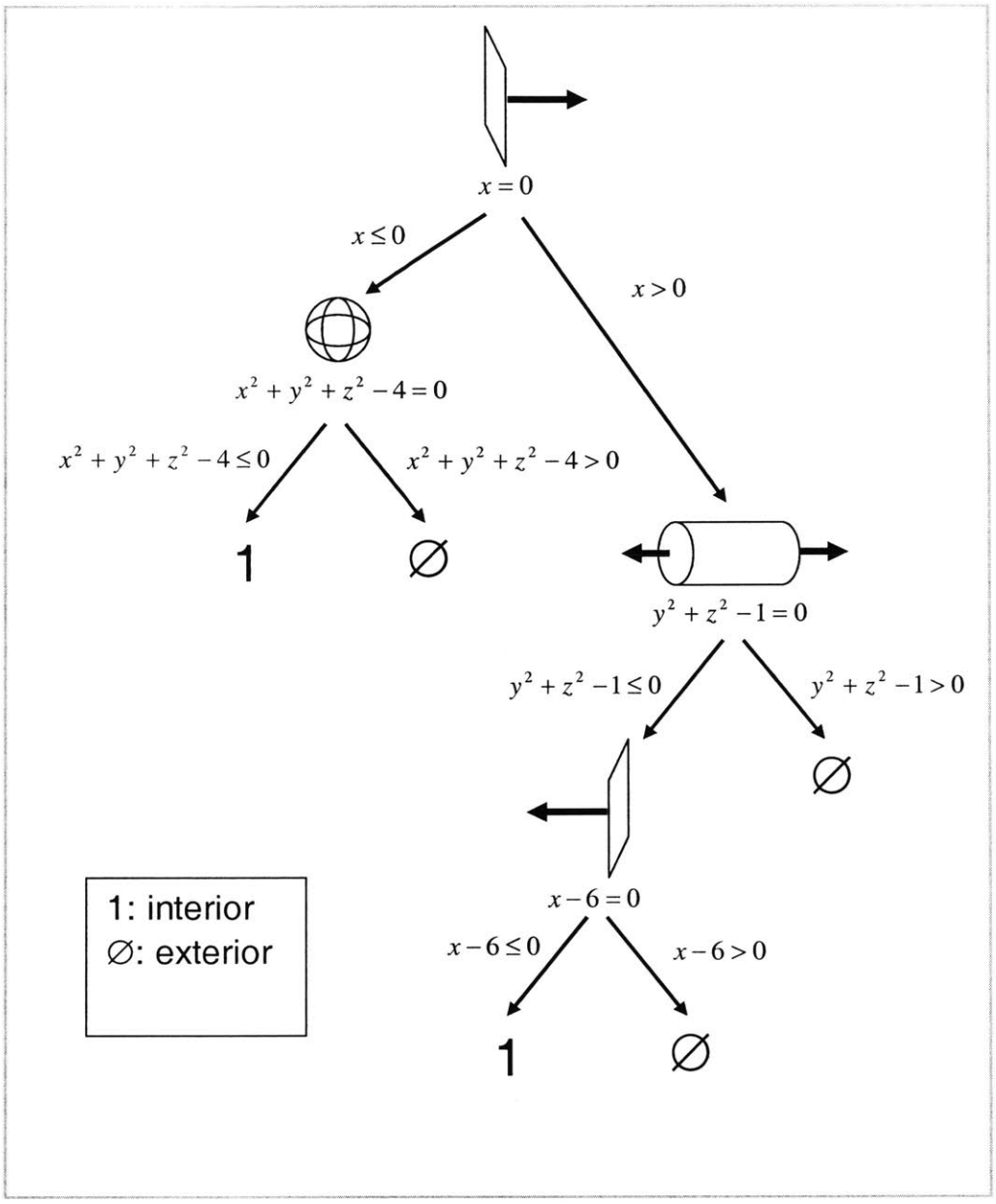

Figure 3-2: Binary Space Partition tree (adapted from Ho 1999)

3.2.1.3 Point Cloud Representation:

Point clouds are a subset of unstructured polygonal models (in terms of the taxonomy introduced in chapter 2).

A point cloud representation consists of a set of *n* points given by their *x, y, and z* coordinates. It provides no topology or connectivity information and therefore has no

sense of exterior or interior of the model. The primary disadvantage of the point cloud representation is that it is not watertight. Its advantages include simplicity and ease of intersection detection with implicit equation defined solids.

A point cloud is generated using a random sample of points from a surface representation such as:

- Tessellated surface approximation;
- NURBS representation.

Point cloud density is selected by achieving a balance between accuracy (aided by larger density) and computation time (aided by lower density). The Stencil collision detection methodology employs a combination of Point cloud and triangulated-surface mesh to represent the part object.

3.2.1.4 Triangulated Surface Mesh:

Each point on the point cloud is associated with a triangle in the triangle mesh.

The triangulated surface mesh representation is a higher-level solid representation (refer to solid modeling hierarchy illustrated in Figure 3-2) and provides additional support to the point cloud representation.

With data structures as described above to represent the tool and part objects, collision detection can therefore be carried out by checking every point in the point cloud against/along the BSP tree to verify whether or not collision has occurred. However, in order to meet the speed and auxiliary information requirements associated with haptics, further modifications were made to the algorithm and data structures.

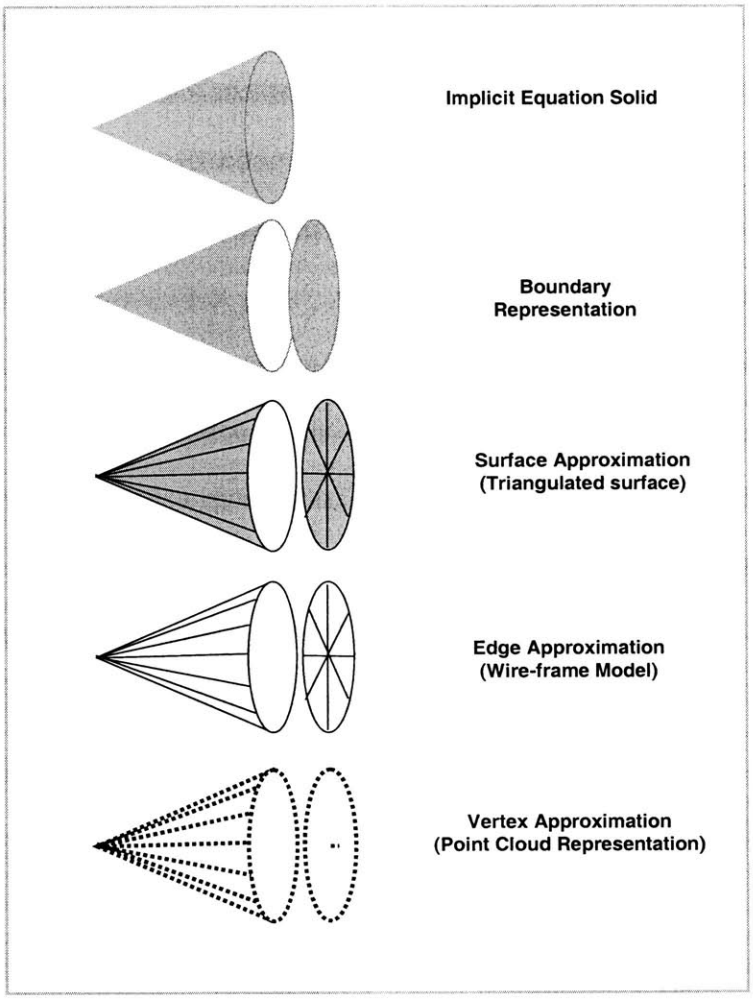

Figure 3-3: Solid Modeling Hierarchy

3.2.2 Efficiency improving Approaches

3.2.2.1 Computationally Efficient Data Structures

As stated in Chapter 2, a number of efficiency improving approaches have been suggested in the literature in order to meet the required haptic servo rates for complex object models. The two of interest in the context of this work are hierarchical databases and spatial partitioning.

The BSP trees, discussed in the previous section, that are used to divide space for both the tool and part object models are an example of spatial partitioning techniques.

Others include extensions to multi-space partitions, and spatial partitioning based on space-time bounds or 4D testing. (Lin & Gottschalk 1998)

Bounding volumes are a particularly good example of hierarchical databases and have been used in the stencil collision detection methodology. Their implementation for the haptic stencil is discussed below.

- **Bounding Volumes:**

To further improve efficiency, both the models are represented as bounding volume hierarchies. Specifically, Ho uses a bounding volume hierarchy called a k-discrete oriented polytope (k-DOP) tree.

A bounding volume is a geometric solid whose interior completely encloses the entity that the volume is designated to bound. It therefore plays the role of a simplified approximation of the actual entity. Also, the geometry of the bounding volume is generally less complex than the geometry of the entity that it bounds.

The idea behind bounding volumes is that if two bounding volumes do not overlap, then no entity within one bounding volume can overlap any entity of the other. However, when bounding volumes do overlap, nothing can be said about the state of collision of the bounded entities. As a result, they work particularly well for "rejection" tests. Unfortunately, when the two objects are in close proximity and can have multiple contacts, large numbers of bounding volume pairs need to be checked for collision resulting in a significant degradation in performance. In general, though, bounding volume hierarchical trees enable collision detection to be carried out faster in the average case.

The suitability of a particular bounding volume depends on its ability to conform to the shape of the bounded entity and the ease of determining whether two bounding volumes overlap. Improvement in one factor is usually at the cost of the other.

When there is overlap, then each point in the point cloud of the overlapping bounded volume is checked for collision against/along the implicit equation BSP tree to verify whether or not collision has occurred.

Recent work has been focused towards tighter-fitting bounding volumes. Gottschalk 1996 et al, have presented a fast algorithm and a system, called RAPID, for interference detection based on oriented bounding boxes (OBBs), which approximate geometry better than axis-aligned bounding boxes. Held 1998 et al, have used discrete orientation polytopes (k-DOPs), which also are superior approximations to bounded geometry.

- **AABB:** is an Axis-Aligned Bounding Box shown in Figure 3-4(a). This is a rectangular prism whose faces are aligned with the x, y, and z coordinate axes. If one plane doesn't overlap then all three don't. Therefore, at most 3 interval tests are needed to check the overlap status of the bounding boxes (one for each axis interval along which the bounding box exists).
- **OBB:** is an Oriented Bounding Box as shown in Figure 3-4(b). This is basically a modification of the AABB such that the faces of the rectangular prism are not restricted to be aligned with the coordinate axes. The disadvantage with this is that the overlap test becomes complicated since the simplification due to coordinate planes intersecting as in AABB no longer holds/works.
- **k-DOP:** is a k-Discrete Orientation Polytope shown in Figure 3-4(c). It is a further modification to the OBB by increasing the number of faces. A k-dop is a convex polytope whose facets are determined by half spaces whose outward normals come from a small fixed set of k orientations. It is constructed by the intersection of k half spaces. The number of interval tests for a k-DOP is k/2. Held et al [Held 1998] describe in detail the choices of k and the implementation of k-dop bounding volume trees.

An 8-dop (such as the one illustrated in 2D in Figure 3-4(c)) has eight fixed normals that are determined by the orientations at ±45, ±90, ±135, ±180 degrees. Axis-aligned boxes (in 3D) are therefore 6-dops with fixed normals at positive and negative orientations of the coordinate axes. Higher order k-dops are therefore constructed by adding more orientations for determining the associated fixed normals. By adding orientations considered particularly natural, some of the standard k-dops in literature progress in the order: 6-dops, 14-dops, 18-dops, and 26-dops. Ho [Ho 1999] tested collision detection using 14-dops, 18-dops, and 26-dops and found k=14 to be efficient in terms of memory requirements and computation time, at the same time providing reasonable accuracy in bounding the given geometric entity.

The haptic stencil therefore utilizes the k-DOP bounding volume with k = 14, which is constructed using the vectors (1, 0, 0), (0, 1, 0), (0, 0, 1), (1, 1, 1), (1, -1, 1), (1, 1, -1), and (1, -1, -1). Thus, this particular *k*-dop uses the six half spaces that define the facets of an AABB, together with eight additional diagonal half spaces that serve to "cut off" as much of the eight corners of an AABB as possible.

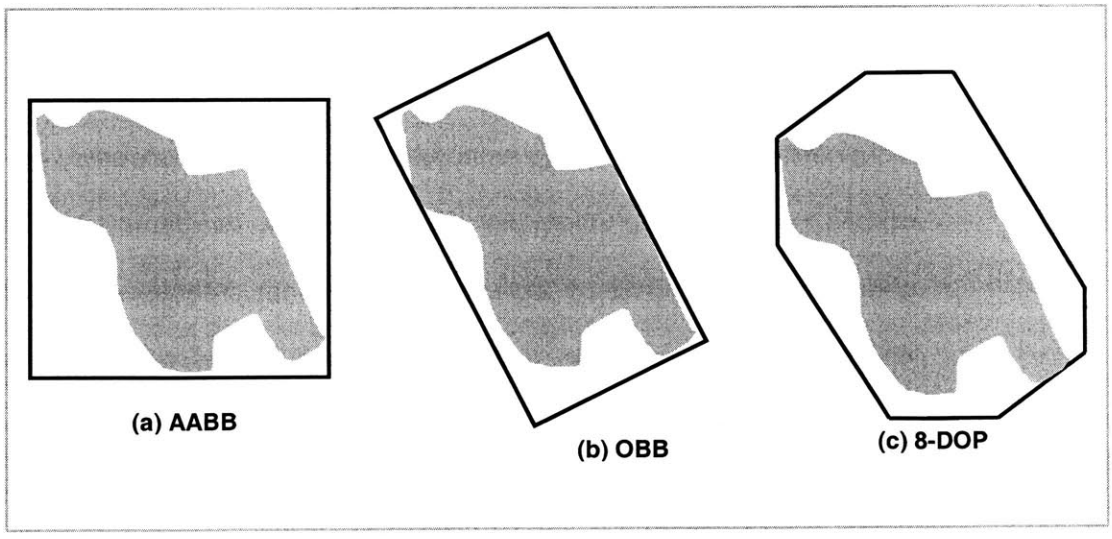

Figure 3-4: Types of Bounding Boxes (illustrated in 2 dimensions)

A binary tree of bounding volumes hierarchy is used for each of the objects to check for intersection in stages. Each object is divided in half, and each of the halves is bounded with smaller bounding volumes. In this way, children are only tested when the parent has tested positively for overlap.

For the point cloud representation, the root bounding volume encloses all points of the point cloud. Hierarchy continues to subdivide points into child subsets until the number of points is less than some predetermined value, at which point the bounding volume is considered a leaf.

In the case of the implicit equation BSP tree which is already a binary tree, one single bounding volume is used to enclose the whole object.

3.2.2.2 Simplifications

Even though the collision detection method is exact, the models being approximate representations of the actual objects result in approximate results.

For extent of penetration information, only one of the representations/models needs to be a solid representation (the interior and exterior of which are defined) since the amount by which object A penetrates object B is the same as the amount by which object B penetrates A. This greatly reduces the space requirements and simplifies computation. Even though interior of point cloud is not defined, any point in the point cloud cannot penetrate the implicit equation defined object undetected, because when object A penetrates object B, object B also penetrates object A by the same distance. We therefore exploit the fact that the implicit equation defined solid has the interior defined.

- **Axi-symmetry and Profile of Revolution:** The axis of symmetry of the controlled object is exploited to reduce the complexity of the Binary Space Partition (BSP) tree. The controlled object can be defined as a volume of

revolution. This profile definition simplifies the BSP tree of the probe (controlled) object by representing the 3D solid as a 2D profile.

3.2.3 Approximations/Assumptions

The primary assumption in Ho's work was that the controlled entity is known, and that the shape can be constructed using a form of constructive solid geometry in which the surfaces are expressed as implicit equations.

Another assumption in Ho's work was that the controlled entity is axially symmetric, and this assumption has been used to advantage in the present implementation. Since the entity being controlled in our setup is a milling machine tool that is also axially symmetric.

3.2.4 Algorithm Procedure

The above listed requirements for good haptic simulation were met with the given collision detection algorithm.

The inputs to the algorithm are the part and tool models, and tool translations specified by user via a 5 DOF haptic device, and the output is corresponding tool translation in the simulation and appropriate force feedback to the operator. Auxiliary information such as penetration depth and direction vector for "decollision" is also computed by the algorithm. This information is very useful for collision remediation as will be discussed in Chapter 4.

The primary aspects of the algorithm are summarized in this section.

Ho's method detects collision between and *exact solid representation* and a *modified vertex representation*. The tool object is represented by an implicit equation binary space partition tree (exact representation) while the part object is represented by a Point cloud triangulated surface representation (modified vertex).

Procedure: The collision detection works by checking for overlap between the bounding volumes of the two objects. When there is overlap, then each point in the point cloud of the overlapping bounded volume is checked for collision against/along the implicit equation BSP tree to verify whether or not collision has occurred.

Results showed that for the specified tool object and a point cloud density of less than 13 points per unit area, the interference analysis and force response calculations run in less than one millisecond in the average case. Some of the factors that influence speed, space/ memory utilization and accuracy of this algorithm have been briefly outlined in the following section.

3.2.5 Influential Factors

Physical dimensions do *not* have an effect on computation time. The number of colliding points, however, is one of the most important factors to computation time. Therefore, entities with dense point clouds are likely to have more points in a state of collision than the same entity with more sparse point clouds.

Collision detection between the point cloud representation and the implicit equation representation works best when the implicit equation object is not significantly smaller than point cloud density. Since the point cloud representation is not watertight, to avoid grossly missing contact, the gap between points should be less than the smallest feature of the implicit equation defined object.

The given methodology is applicable for both convex and non-convex objects and is therefore useful for simulating haptic systems containing arbitrary solid objects. Furthermore, triangle-set methodologies do not even require that the object be a coherent solid; any set of triangles is applicable.

3.2.6 Pseudo code[17]

The function BVcheck(A, B) compares the bounding volumes of tree A to tree B to determine which leaves are in a state of overlap. When overlapping leaves are found, the function Check(A, B) is called to determine the state of intersection between a set of points A, and an implicit equation binary space partition B.

➢ **Pseudo code 1: BVcheck(A, B) Hierarchical search through Bounding Volume tree**

```
BVcheck (A, B)
if BV (A) overlaps BV (B)
   if A != LEAF
        BVcheck (RightChild (A), B);
        BVcheck (LeftChild (A), B);
   else
        if B != LEAF
             BVcheck (A, RightChild (B));
             BVcheck (A, LeftChild (B));
        else
             Check (PointCloud (A), BSPTree (B));
else
   skip
```

➢ **Pseudo code 2: Check(A, B)-Method for checking point cloud A against BSP-tree B**

Each point of point cloud A is verified using AuxCheck (p, B), where p is one point of the point cloud A. If B is not a leaf, the implicit function associated at the root of B is evaluated at point p. Depending on the sign of the evaluation, either the left child or right child of B is checked against p. When the process reaches a leaf, point p is evaluated

[17] Ho 1999

against the leaf and if p is found to be in the interior of the leaf then COLLSION has occurred.

```
Check (A, B)
for (i = 0; i < NumberofPoints (A); i++)
   if (AuxCheck (A[i], B) == COLLISION)
        return COLLISION
   else
        skip

AuxCheck (p, B)
   if B is a leaf
        if (B == 1) return COLLISION
        else return NO COLLISION
   else if implicit function of B evaluated at point p > 0
        AuxCheck (p, LeftChild (B))
   else
        AuxCheck (p, RightChild (B))
```

Chapter 4: COLLISION REMEDIATION

Collision remediation is a key and unique concept in haptic stencils. The depth and difficulty of this problem was first recognized by Golaski [Golaski 2001] and in this thesis we explore it in a further depth. In the context of this work, collision remediation refers to determining the appropriate (legal) position for the real tool if the tool position specified by the user (via the haptic probe) is found to be a state of collision (illegal).[18]

Collision remediation provides the stencil-like behavior by enforcing geometric constraints on the regions/surfaces by preventing penetration by the tool object. The results from the collision detection and collision remediation modules are used to control the desktop milling machine which cuts out a copy of the part object used in the haptic simulation from a given stock according to the motions specified on the haptic probe by the operator.

The primary requirements of an appropriate collision remediation methodology for the haptic stencil include:

- The methodology should *not* allow penetration of the geometry specified by the stencil.
- Offset between the haptic probe and probe-controlled entity should be minimized. This implies the "decollided" position specified by the method should always attempt to be in sync with the haptic probe.

Collision remediation is more than just penetration depth calculation. Once penetration depth has been calculated, remediation requires determining how to separate the part and

[18] The concept of collision remediation comes up in approximate terms in a few papers like [Basdogan et al 1999], but seems rarely to have been addressed formally.

the tool. This must really be done in a way that reflects reality – after all, haptics is a form of virtual reality. Strictly, this should be done by running a solid modeling simulation. However, this is difficult and slow, and depends entirely on the material properties of the solids we wish to model. Would modeling a rubber ball be very different from modeling a billiard ball? Does a visco-elastic material behave differently from a solid? Indeed these are important considerations, but here we reduce the problem to modeling stiffness, and getting some understanding of whether friction can be modeled. We start by studying penetration depth computation, below.

4.1 Penetration Depth Calculation

In conventional collision detection, a step response is used as feedback. In other words, a simple *collision* or *no collision* response is determined. However, for haptic applications, in order for the user to have an intuitive experience and to prevent instabilities, it becomes necessary to go beyond that step response approach. One way to do this is to scale the force fed back with penetration depth, since the user should ideally feel more resistance as penetration increases (when modeling a stiff surface at least). A simple mechanistic model such as Hooke's law (equation 4.1) could therefore be used.

$$Force = Stiffness * PenetrationDepth \quad \textbf{(4-1)}$$

In haptics, the main goal is to compute the interaction forces. These forces are usually scaled with the penetration depth. As a result, the depth of penetration and how the penetration evolves are important. Several methods have been proposed in the haptics and computer graphics community to determine/ express penetration depth. Some of the most common have been outlined in the following paragraph.

One simple way to determine the depth of penetration is to use the shortest distance between the probe tip and the object's surface (Massie, 1993; Massie & Salisbury 1994).

This approach works well for primitive objects such as a cube, sphere, cylinder, etc; however, the drawbacks of this technique are that it cannot display the objects that are thin or polyhedral (Massie 1993; Ruspini et al., 1996, 1997[19]). An alternative approach uses a constraint-based method (Zilles & Salisbury 1995). This method defines an imaginary point – the god-object point – that is constrained by the facets of the object. The penetration depth is then defined as the distance from the god-object to the probe tip. This approach can be applied to polyhedral objects, even when the objects are thin. However, this method requires different sets of rules to handle concave and convex objects. [Basdogan et al 1999]

Basdogan et al 1999 proposed the *neighborhood watch,* an efficient haptic-rendering technique that can handle both convex and concave objects uniformly. Although this algorithm is conceptually similar to the god-object algorithm, it follows a different approach.

The method used in the interference analysis for the haptic stencil interface employs auxiliary equations computed during the collision detection procedure, as described in the following sub-section.

4.1.1 Interference Analysis and Auxiliary Information

The interference analysis procedure follows the same procedure as the collision detection procedure presented in chapter 3 and employs the same part and tool object representations (i.e., the same data structures). Since the implicit equation BSP-tree incorporates region information about the solid interior, after a point cloud is identified as in the interior, the associated auxiliary functions are evaluated for the point's location, and the auxiliary information for the given point is determined.

---

[19] D. C. Ruspini, K. Kolarav, & O. Khatib (1996): "Robust haptic display of graphical environments. In J. K. Salisbury & M. A. Srinivasan (Eds.), *Proc. of the First PHANToM Users Group Workshop*, MIT-AI TR-1596 and RLE TR-612-- (1997) "The haptic display of complex graphical environments", *ACM (Proc. of SIGGRAPH)*, pp. 345-352.

Penetration depth information and a direction vector for the shortest path to the exterior of the penetrated object make up the auxiliary information computed during the interference analysis. This information is required by the collision remediation module to determine the appropriate collision response.

An auxiliary equation is associated with the interior of the implicit equation BSP-tree represented object (controlled entity) for the purpose of penetration depth calculation. It determines the penetration depth for each colliding point in the point cloud. The form of the equation must therefore be designed to give the appropriate depth for a given probe solid. For more complex geometries with multiple implicit equations for multiple surfaces, determining which penetration depth is of importance is necessary. The geometry is therefore divided such that:

- Each surface of the object has its associated region of points that have the surface as the closest;
- The divisions are represented as implicit equations too (which are then incorporated into the BSP tree).

In addition to the depth information the collision remediation module also needs *direction* indicating the shortest path to the solid's exterior.

- At the surface of the solid, this direction is defined as the surface normal at that point;
- In the interior of the solid, this direction is that of the shortest displacement to the surface.

Procedure: Given a point (x, y, z), the BSP tree is traversed to determine in which region the point resides. If the region is part of the interior of the probe, then collision has occurred, and evaluation of the auxiliary functions associated with the region determines penetration depth and direction. Once the penetration depth is calculated the appropriate collision response is determined and executed.

The problem usually associated with this method of penetration depth calculation in other "vector field methods" [Zilles & Salisbury 1995] is that force discontinuities are encountered when traversing volume boundaries. This problem is alleviated in our implementation due to the fact that the volume boundaries associated with the auxiliary equations for penetration depth (and therefore force) calculations are specified in the tool object which interacts with the part model primarily with its tip. As a result, not many volume boundaries are traversed. However, our method does share one other limitation usually associated with traditional "vector field methods" – **snap through** in thin objects. Because of the limited servo and mechanical stiffness, the haptic interface point must travel somewhat into the object before enough force is generated to make the object feel "solid". When this distance becomes greater than the thickness of the object, unrealistic feedback sensations are created and the user gets the impression of being pushed out the other side of the object. Small and thin objects do not have the internal volume required to generate convincing constraint forces [Zilles & Salisbury 1995].

4.2 Collision Remediation Techniques

> "Touch is unique among the senses because it allows simultaneous exploration and manipulation of an environment"
>
> [Zilles & Salisbury 1995]

As a result, simulating touch not only requires modeling the appropriate object properties but also providing a response to the user while he / she is exploring the simulated environment.

The type of collision response depends on the application. In the case of haptics, as explained earlier, the conventional step response is not sufficient. The collision response for the haptic stencil basically consists of a force feedback and collision remediation

(specifying a "decollide" position for the tool). How these two are combined to form the overall collision response forms the discussion in this section.

4.2.1 Inherent Collision Remediation[20]

From the interference analysis, a set of intersecting points from the point cloud is determined. Furthermore, the interference depth and direction of each point is known from the analysis. This information is then consolidated into one composite force/torque response. The force response is then fed back to the operator via the haptic device.

After a point has been identified as being in the interior of the solid and the appropriate auxiliary functions have been evaluated, the state of collision between that specific point and the implicit equation object is defined by the location of the point, the direction of the point outward normal vector, the depth value determined by the auxiliary function, and the direction defined by the vector auxiliary function. These factors are then combined to determine a force response (force magnitude and direction) appropriate for that point.

The magnitude of force for each intersecting point is determined by scaling with penetration depth, as shown in equation 4.2, where d is the penetration depth and k is a constant.

$$magnitude = kd \tag{4-2}$$

The direction of force is defined by the average of the two direction vectors available – the surface normal of the point in the point cloud and the direction of the shortest distance to the surface of the implicit equation defined object. Ideally, the direction of one should be the exact opposite of the other. The average approach is used to take care of

---

[20] Ho 1999

situations in which the two directions are not anti-parallel as shown in Figure 4-1(equation 4.3).

$$v_{df} = \frac{1}{2}(\hat{n}_{pc} - \hat{n}_{eq}) \qquad \textbf{(4-3)}$$

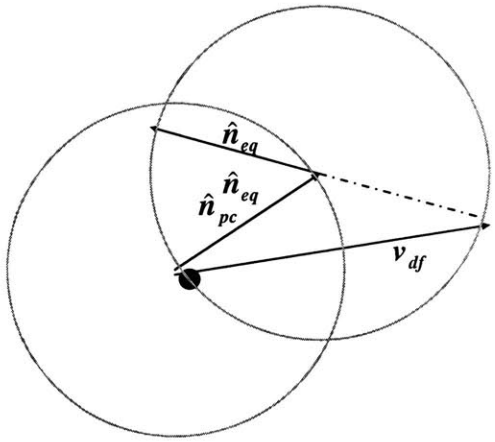

Figure 4-1: Force Direction Resolution

The collision response is then formed by combining the individual forces due to each of the colliding points in the point cloud. In order to compute a composite force response for a given state of collision, the *compliance center* is used.

> "The compliance center for a particular body is the point in space such that any translational force acting on the body that acts through this point results in pure translation (no rotation). Furthermore, any rotational force (torque) acting on the compliance center results in rotation only"
>
> [Asada1986].

The compliance center is therefore the natural choice for the location of resolving the many forces into one force/torque. Given the point coordinates (p_x, p_y, p_z), compliance center coordinates (cc_x, cc_y, cc_z), force direction $F = [f_x, f_y, f_z]$, and the force magnitude f_m, the equivalent force and torque acting at the compliance center due to a point are therefore given by:

$$F_{cc} = [f_m \bullet f_x, f_m \bullet f_y, f_m \bullet f_z] = f_m \bullet [f_x, f_y, f_z] \quad \textbf{(4-4)}$$

$$T_{cc} = [r_y f_m f_z - r_z f_m f_y,\ r_z f_m f_x - r_x f_m f_z,\ r_x f_m f_y - r_y f_m f_x] = f_m (R \times F) \quad \textbf{(4-5)}$$

In the equations 4.4 and 4.5, *R* is the distance vector between the point and the compliance center. It should be noted that these equations assume that all vectors and point coordinates are in world space coordinates.

It can therefore be seen that the translational force is equal to the force acting at the point, and the torque is proportional to the force magnitude and the distance between the point of action and the compliance center. The composite force and torque for all points is simply the summation of each of these calculated forces and torques.

In the haptic stencil model, a single point of the point cloud exerts only a translational force upon the object at the location of the point. This translational force may resolve to be the combination of the translational force and a torque at the compliance center, but at the point of contact only translational force is modeled. These force interactions represent the reaction forces between the two objects when they are in contact.

Forces are then fed back via the haptic device motors to the user. These resistive forces are what provide the inherent collision remediation in the simulation. This is because, the resistive forces which scale with penetration depth end up preventing the user from penetrating further, and in the quest to stop feeling the resistance the user ends up moving the probe to a decollide position automatically. For reasons of safety however, the resistive force cannot exceed 5N so that if the user continues to push hard enough penetration will continue.

> "Due to the inherent mechanical compliance of haptic interface devices, the maximum stiffness of any virtual object is limited."

[Zilles & Salisbury 1995]

The aim of the collision remediation module is to prevent penetration (at least by the real machine tool) under any circumstances. The following two subsections discuss some of the collision remediation methods that were developed and tested for the haptic stencil setup.

4.2.2 Sticking Issues

The "sticky" approach to collision remediation involves keeping the tool in the last legal position. This means that once penetration occurs, the controlled entity/object is held in last specified legal position, even if operator continues to move in the penetrated state. A primary limitation to this approach becomes evident when trying to move on a flat surface. Each time there is a slight penetration the haptic probe and controlled object lose synchronization and the result is the feeling of moving on a bumpy surface. This is undesirable since we don't want a situation where the tool is stuck in last the legal position while the operator is moving with deep penetration to an extent that the haptic probe and real tool are significantly offset.

4.2.3 Slipping Issues

"Slip" collision remediation, on the other hand, involves the controlled object following the motion of the haptic probe even after penetration except that the controlled object remains outside the boundary of the part object. In other words, the controlled object mirrors the motion of the haptic probe on the surface, with the probe specifying motions within the interior. This approach is harder to implement since it requires penetration depth and direction information as well as the surface geometry of the part (stationary) object.

4.3 Stencil Collision Remediation Methodology

In Figure 4-2, the red dots represent a sequence of illegal positions enforced by the operator. The blue dot represents the last legal position of haptic probe (operator). If we use the sticky approach, the tool will be stuck at the blue spot while the operator would have reached the third red dot. However, as already explained, it is preferable for the tool to follow the probe as closely as possible.

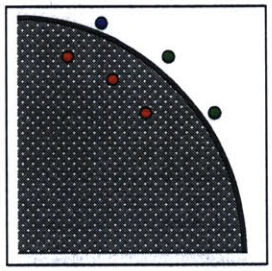

Figure 4-2: Collision Remediation Methods

The approach that was eventually used for the haptic stencil is based on "slip" collision remediation and attempts to make the tool follow from the blue to the two green dots in sequence (Figure 4-2).

The methodology being implemented employs the force resolution and penetration depth information from the interference analysis. The real tool is moved in the direction of the reaction force (sent to the operator via the haptic device) by a distance equivalent to the penetration depth.

Procedure: Check if present tool position specified by haptic probe is legal. If legal, transmit position data to real machine tool. If illegal, collect auxiliary information for given point and send real machine tool in direction of force feedback (figure 3) by a distance equivalent to penetration depth.

4.4 Graphics – Visual Update

The graphic screen displays the stock and tool providing visual feedback to the user. The screen refreshes every 0.05 seconds (about 20 Hz), updating the tool position and re-coloring regions of the part model "machined".

The Open Inventor toolkit was used for graphic rendering purposes. The libraries offer a number of very useful features. A callback function is used to update the screen with the latest tool and part properties and position information. As each of the triangles in the triangulated representation of the part are contacted (or "touched") by the tool they are painted a different color to give the user visual feedback. This section details some of the Open Inventor tools that were used to implement the graphics module.

- Sensors: A sensor is an inventor object that watches for various types of events and invokes a user-supplied callback function when these events occur. There are two types of sensors: Data sensors and Timer sensors. Timer sensors are used in our interface to respond to certain scheduling conditions. Scheduling a sensor means adding it to a queue so that it can be triggered at some future time. Sensors can be placed in one of two queues: Timer queues and Delay queues. A Timer queue is called when an alarm or timer sensor is scheduled to go off. A Timer queue is scheduled so that its callback is invoked at a specific time. Timer queue sensors are sorted within the timer sensor queue by time rather than by priority.

  General Sequence for Timer Queue Sensors:

  - Construct the sensor
  - Set the callback function
  - Set the timing parameters of the sensor
  - Schedule the sensor using the `schedule()` method
  - When finished with the sensor, delete it.

The unschedule () method can be used to remove a sensor from a queue, change its parameters, and schedule the sensor again.

- Callback functions: User written functions that are called under certain conditions. Callback functions for sensors usually take two arguments, one of type void* that can be used to pass extra user-defined data, and the second of type SoSensor* that holds the pointer to the sensor that caused the callback.

In the haptic stencil implementation, the sensor is set to go off at regular intervals (of 0.05 seconds).

5.1 Control System Overview

The control scheme for the haptic stencil interface can basically be divided into two loops: the haptic device control loop and the machine tool control loop, as illustrated in Figure 5-1.

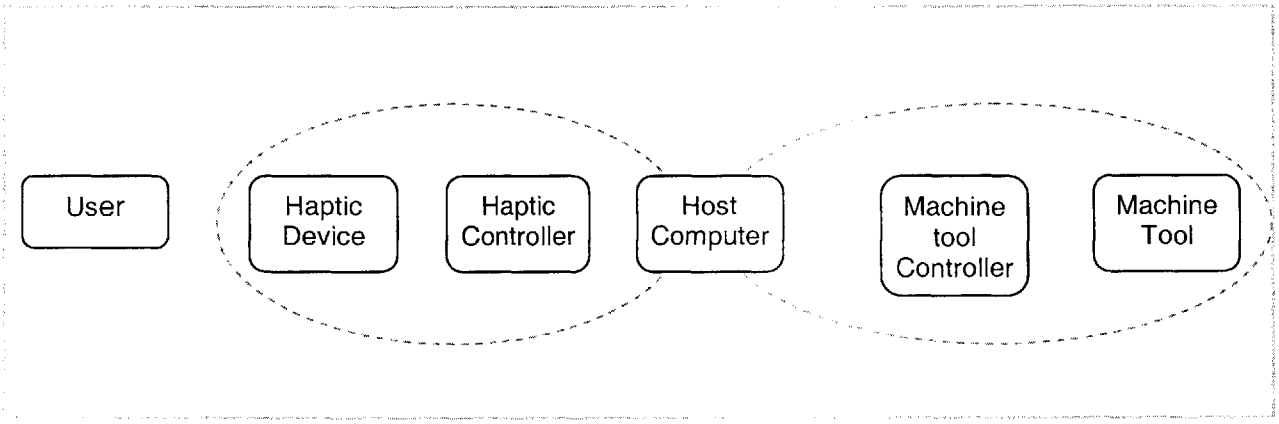

Figure 5-1: Two loop Control Scheme

5.2 Haptic Control

Hardware details for the motors and encoders in the haptic device are given in Chapter 6 – System Implementation.

As described in Chapter 2, there are basically two broad strategies available for controlling haptic interfaces: Impedance control and Admittance control. Due to the fact that our implementation requires a force response to the user, impedance control makes more sense. Subsequently, the control method implemented by Adachi (Suzuki Corporation, Japan) is Proportional or Stiffness control.

The forces computed from the simulation are sent to the haptic controller unit that drives the haptic interface. In this way, the user feels force feedback, and the haptic part

of the control loop is closed. The haptic controller simulates the forces by actuating the appropriate motors – if no resistance is met these forces translate to a new haptic probe position.

5.3 Machine Tool Controller

The MMC-4S motion control system consists of three basic elements: the controller unit, the multiplexers, and the drivers (MM 2.0). The controller contains a CY 545 (550) stepper motor controller. The multiplexer section allows the CY 545 to control up to four step motor channels by multiplexing the motion signals between the channels. The Cybernetics 545 microprocessor uses a "High-Level" command set (26 characters and symbols) to control all actions of the system.

In this system, the User Bits of the CY545 (USRB 0-7) are assigned to control both the 8 line output mux and the 8 line input mux. The I/O assignments for the User bits are listed in Table 5-1.

Table 5-1: Multiplexer User Bit Allocation

<table>
<tr><th colspan="2">USRB</th><th colspan="2">Input Ports</th><th colspan="2">Output Ports</th></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>0</td><td>driver enable/disable</td><td>0</td><td rowspan="3">Home sensors for each of the drivers</td><td>0</td><td rowspan="3">Motor/Driver addresses</td></tr>
<tr><td>1</td><td>data bit</td><td>1</td><td>1</td></tr>
<tr><td>2</td><td rowspan="3">Used to specify addresses</td><td>2</td><td>2</td></tr>
<tr><td>3</td><td>3</td><td>3</td></tr>
<tr><td>4</td><td>4</td><td rowspan="4">General Purpose</td><td>4</td><td rowspan="4">General Purpose</td></tr>
<tr><td>5</td><td>read bit</td><td>5</td><td>5</td></tr>
<tr><td>6</td><td>busy/CTS</td><td>6</td><td>6</td></tr>
<tr><td>7</td><td>write bit</td><td>7</td><td>7</td></tr>
</table>

When a channel is selected, the Step pulses and the Direction signal from the CY 545 are directed to a motor driver by the multiplexer. Additionally, selection shifts the

driver from Park power to Full power. The multiplexers also direct the signal from the Home sensor, for that motor, back to the controller.

Normally, one channel is selected at a time as the MMC (Multiple Motor Controller) card only generates signals for one step motor. Curves and 3D motions are therefore produced by single stepping the system and switching motors each step. A major advantage of this system is the ability to trace true point-to-point patterns at up to 1K steps per second.

5.3.1 Controller Microprocessor

The CY545 stepper motor controller microprocessor is a 40-pin, CMOS, +5V device that combines high performance with low cost. It is capable of generating step rates of up to 27,000 steps per second, using pulse and direction signals, which provide adaptability to full-step, half-step, quad-step, or micro-step applications. The chip's internal step register can count up to 16 million steps per motion, or the device can be set to step forever until an abort signal is received. Commands to the device are sent in ASCII or Binary formats, using Serial or Parallel interfaces, from a host CPU or from local external memory. Eight general purpose I/O lines can be addressed by command to control external events and can be read to trigger waits, loops, program branches, and stepping motions.

The chip supports programmable start rate, accelerate/decelerate, and slew rate, and absolute or relative position control. These and other parameters may be sent as commands from the host CPU, or may be entered manually from thumbwheel switches, which are directly supported by the CY545. Internal or external direction control and separate limit switch signals are also available to the application. A Home command directs the motor to step to a home-sensor, automatically compensating for mechanical

backlash, and a motor-moving signal provides a motor power driver with an automatic power-down or parking signal when the motor is stopped.

For the haptic stencil, the controller was setup to operate under the Serial command of a host computer. The high level ASCII command set was used to communicate commands between the host computer (haptic simulation) and the controller microprocessor.

The serial format is configured in the following manner: ASCII characters, Adaptive Baud, 8 data bits, no parity, and one stop bit. The CTS (Clear to Send) feature of the CY 545 (User Bit 6) is used as the hardware handshake to control communication between the host computer and the CY 545. When the 545 is busy, it will set the CTS signal to hold off transmission.

5.3.2 Serial Communication Module

This is the module within the haptic stencil that converts legal tool positions specified after collision detection and collision remediation to real tool translations. ASCII commands to the controller microprocessor are also sent using this module. In this module, the serial port is initialized, and parameters such as feed, speed and ramp speed are specified. Arena communication is established between the tool control module (tool child process) to get tool position information in the form of ASCII commands (using hexadecimal format) which are programmed to correct for the fact that the output from the simulation is tool motion but in the real tool the work piece is moved relative to the tool. Flow control of information is enforced using hardware handshaking with CTS/RTS (feature of RS232 communication).

CY545 serial interface works on a fixed format character, which is always 8 data bits, with no parity bit and one stop bit. All 8 data bits are used to interpret the command

character values. It is therefore important to send the proper 8 bit codes for the various ASCII command characters.

5.4 Controller Characterization and Modeling

A virtual tool is used as a proxy for the real tool for the purposes of decollision and control of the real machine tool. The virtual tool is a numerical representation of the machine tool's position. The input position specified by the user through the haptic device is checked against the collision detection, collision remediation and speed-limiting (discussed in the next section) algorithms. This way, the control of the real machine tool is decoupled from the haptic control loop. As a result the haptic refresh rates can be met despite the fact that the machine tool has finite velocities and accelerations and cannot meet the 1000Hz update.

> "By decoupling the haptic display control problem from the design of virtual environments, the use of a virtual coupling network frees the developer of haptic-enabled virtual reality models from issues of mechanical stability."
>
> [Adams 1999]

The concept of the virtual tool is similar to the God-object proposed by Zilles & Salisbury 1995 and the Intermediate Representation introduced by Adachi et al 1995. The idea behind the intermediate representation is using an intermediate space for controlling haptic interfaces by updating a virtual plane at a low frequency while maintaining a high update rate at the force-control loop of the interface. The virtual plane makes collision detection independent of the control of the haptic interface.

The god-object is a virtual model of the haptic interface which conforms to the virtual environment. The haptic interface can then be servo-ed to this virtual model. The god-object is placed where the haptic interface point would be if the haptic interface and object were infinitely stiff. In particular, this method is suitable for thin objects and arbitrarily

shaped polyhedra. The snap-through problem is solved because the god-object always remains at the surface of the object, so that the direction of the reaction force is never ambiguous. Additionally, this approach utilizes history to compute the new location of the god-object.

The feedback sent to the haptic device is the force representing the direction and magnitude of difference between position specified by user and the tool (represented by the virtual tool).

5.4.1 Open Loop Control of Stepper Motors

As can be seen from the foregoing discussion, there is no feedback regarding the motor rotor positions, i.e. once the motion commands are sent to the machine tool controller microprocessor, no information regarding command execution and tool position is sent back to the host computer. Thus, the control of the machine tool is an open loop control scheme. In this subsection, some of the issues usually associated with open loop control are discusses with respect to our implementation.

One of the biggest advantages of using stepper motors has been the simplicity with which it can be controlled in open loop (and the subsequent low cost). When the reliability of closed-loop is compared with that of open loop, however, the former begins to look very attractive because it eliminates many of the problems associated with open loop control (mechanical resonance, intolerance of load changes). In the case of our implementation, though, these issues do not come up. For one, the machining speed is maintained at reasonably constant values even if the haptic probe is moved at varying speeds. Secondly, the stepper motors are not operated at full-torque. Additionally, the excitation changes are made gradually to ensure that there is no permanent offset between the intended and achieved rotor position.

5.5 Speed Control

Speed control is necessary in order to ensure that motions from the human operator are not lost due to the different communication speeds between the various modules of this setup. Speed control also helps in providing as 'real-time' a machining experience as possible for a given part and stock combination. Finally, appropriate speed control would be required to identify and alleviate issues associated with certain ranges of velocities of probe motion that may cause instabilities in the haptic device or overall setup.

The speed-limiting module determines and specifies the acceptable velocity ranges of operation. Velocities that could cause a loss of sync or approach instabilities would be automatically filtered.

This module extracts velocity information from the arena and invokes a speed-limiting message. Need to quantify the speed limit for haptic probe and determine corresponding loss of sync between probe and real tool when snap-through occurs. Speed limiting is important for minimizing time lag between probe movements and machine tool motion. It therefore minimizes offset between legal position before and after deep penetration motion. Consequently, it makes collision remediation more efficient.

Speed control would depend on:

- Geometry of solids e.g. sharp corners. These make it important to ensure that machine update rate is on the same order of magnitude as the probe velocity.
- Machine update rate (number of times per second translation info can be sent to machine tool).
- Velocities at which instabilities are expected (from control theory point of view).

5.5.1 Using Machine Update Rate

Presently the method being employed to achieve this is by using the machine update rate. The maximum communication speed for the serial port connecting the host computer

(running the algorithms) to the real machine tool controller is 57600 baud per second. Experimentally, this has been found to translate to an update rate of approximately 0.03 seconds. Depending on the scaling used between the haptic space and machine space, this would translate to a certain number of steps (on the order of tens) in 0.03 seconds that in turn would translate to a certain velocity of probe motion.

System requirements in the context of stencil-like operation can be summarized as follows:

- Stenciled geometry is not machined (gouged) under any circumstances.
- All stock material apart from the stenciled geometry is machined.

Complete machining is ensured in the haptic stencil interface by implementing the software such that the program automatically exits / closes once all points are contacted or "touched". How the first two requirements are met forms the bulk of this chapter.

Additional design specifications arise from haptic simulation requirements, namely:

- 1000 Hz refresh/update rate for collision detection and remediation. (Refer to section 3.1.1)

Finally, as a result of the fact that the stencil is used for machining and implemented on a machine tool, a controller programmed to receive motion commands from the haptic simulation (virtual environment) and transmit them to the machine tool is required.

This chapter presents how all the concepts presented in the earlier chapters come together as the Haptic Stencil. Mapping and synchronization between the different modules are discussed in section 6.1, the hardware used for the implementation is presented in section 6.2, and the overall system architecture is outlined in section 6.3.

6.1 Mapping/Scaling/Synchronization

Workspace dimensions for haptic workspace and machine tool workspace. The part model in the graphics module is defined such that it scales itself and the tool to fit into a tight box defined by the dimensions of the haptic workspace. Additionally, it is ensured that if the haptic probe is "homed" correctly at the beginning of the simulation, the tool (in

both the graphic and haptic workspaces) would be able to reach all features on the part. Homing between the haptic and machine spaces is presently achieved manually by always setting the machine tool axes at particular positions at the beginning of the simulation. This can easily be extended to automatic homing by setting up homing switches and implementing it in the communication module between the simulation and the machine tool controller.

The user is mapped to the tool representation/model/object through the haptic probe. Coordinate frames for haptic, graphic and machine spaces are shown in Figure 6-1.

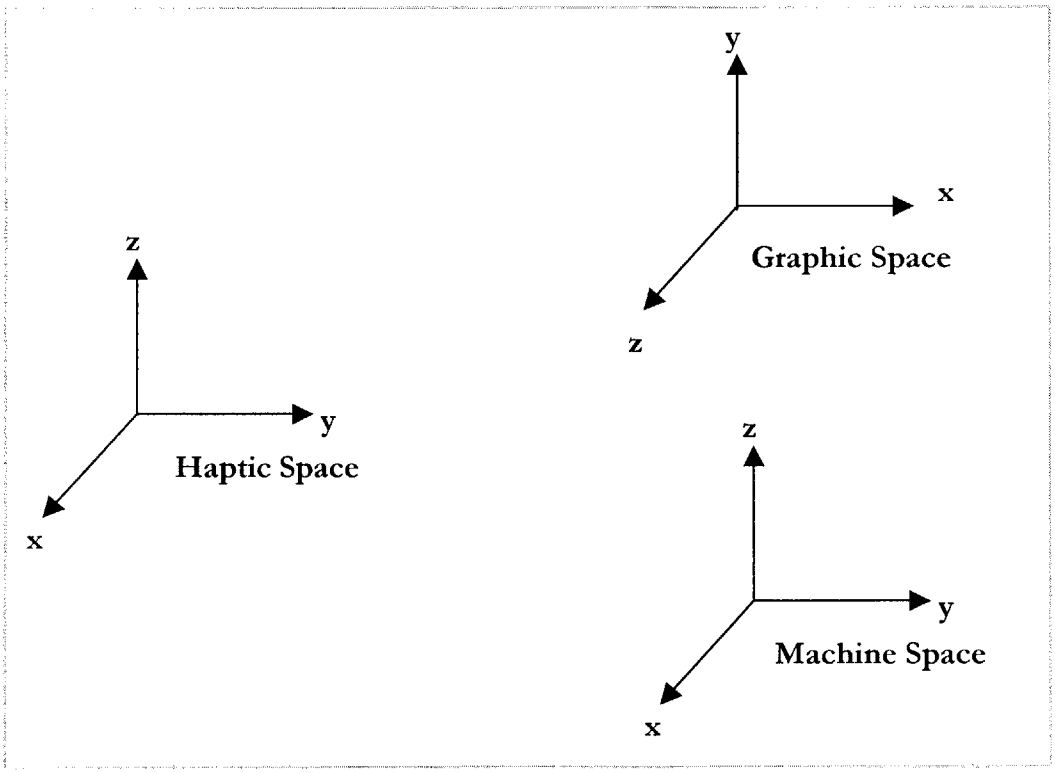

Figure 6-1: Synchronization of Reference frames in different Coordinate spaces

The haptic space dimensions are approximately 50cm X 25cm X 25cm. The haptic space and virtual tool space are automatically synchronized at the beginning of the run such that the virtual tool and part fit snugly into the graphic rendering space and so that tool tip can reach every point in workspace. However, in order to allow the user to move smoothly in the haptic space, the workspace dimensions in the machine space are scaled down by a factor. Experimentally, a factor of 10 was found to provide a reasonably intuitive experience. As a result, the machine space dimensions are 50 mm X 25 mm X 25 mm (or 2" X 1" X 1"). The machine tool specifications are given in Table 6-1

Using *SCALE* to represent the scaling factor between the haptic space and machine space, the relationship between the number of steps to required for a displacement *x* in the haptic (user) space is given as:

x inches in haptic space = (x / scale) inches in machine space

if m = x / scale, then

∴ m inches = N steps = m inches * 16000 steps / inch = 16000*m steps

∴ N steps = 16000 * (x / scale)

∴ N = STEPSPERINCH * (x / SCALE)

6.1.1 Causality Structure

Using the terminology developed by Adams 1999, given in chapter 2, the causality structure of the system implemented in the haptic stencil can be illustrated as shown in Figure 6-2. Both the haptic display type and the virtual environment type employ the *impedance* approach.

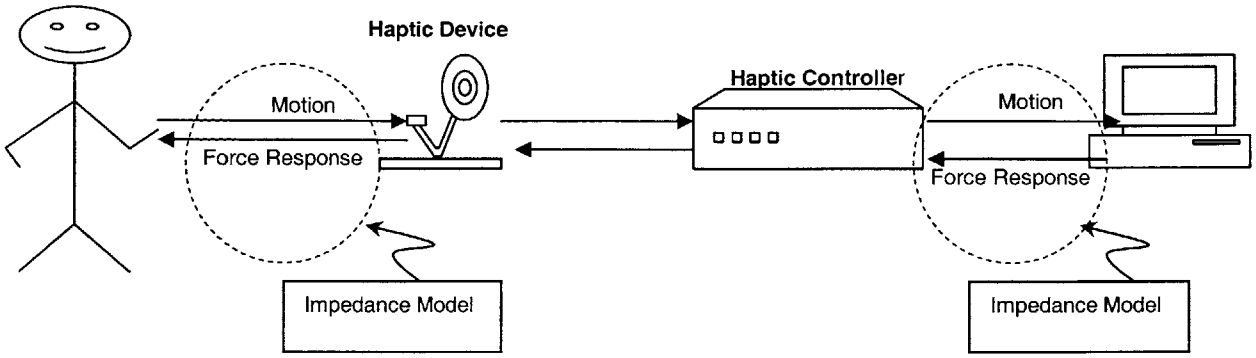

Figure 6-2: Causality Structure

6.1.2 Distributed Computing

In a computer-controlled loop, high bandwidth and low simulation latencies require large computing power and fast communication links - the more complex the virtual world, the higher the computation load requirements. It is thus necessary to replace traditional single CPU machines with distributed computing on multiprocessor or multi-workstation architectures. As a result a dual processor CPU was utilized for the haptic stencil implementation with the graphics and machine tool control running on one processor and the collision detection and interference analysis running on the other processor.

Multiprocessing (using the command `fork ()`) is used to execute the different modules of the interface/simulation. A *process* is an executing program. When a program is run, it is read from the peripheral storage area, where it resides, into memory and then executed. A new process is created when a running program calls the `fork ()` system call. It creates a new process that is a copy of the current process giving two running instances of the same program. The program that calls `fork ()` is called the *parent*, and the newly created process is called the *child*. Commands like `blockproc ()` and `unblockproc ()` are used for synchronization of the processes.

Inter-process communication is established using "shared memory arena". A shared arena is a segment of memory that can be made part of the address space of more than

one process. It is basic to all IRIX Inter-Process Communication (IPC) mechanisms. IRIX semaphores, locks and barriers are all represented as objects within a shared arena. Each process that would need to share access to the arena does so by specifying the pathname of the file.

6.2 Hardware

The experimental setup consists of a 5 DOF haptic device linked to a Sherline desktop 3-axis milling machine tool via a dual processor IRIX 6.0 processor SGI. The machine tool is controlled using a Cybernetics CY545 stepper motor controller (MMC 4S) which can be used for up to 4 stepper motors.

6.2.1 Sherline desktop mill

The machine tool being controlled by the haptic stencil is a Sherline three-axis milling machine, shown in Figure 6-3. The axes are anodized aluminum, with sliding dovetail bearings. Steel lead-screws turn in brass nuts. Travel is approximately 22cm in the x-direction, 12cm in y, and 16cm in Z. The spindle is ½ HP, with a top speed of 2800 rpm. The update rate for the machine tool is about 30Hz.

The specifications for the stepper motors in the mill are given in Table 6-1.

Table 6-1: Sherline Mill Stepper Motor Specifications

Max. Machine Speed (each axis)	1.4 mm/sec	3 in/min
Motor Resolution	640 steps/mm	16000 steps/in
Backlash (x)	0.125 mm	0.005 in
Backlash (y)	0.05 mm	0.002 in
Backlash (z)	0.125 mm	0.005 in
Workspace (x.y.z)	50 x 25 x 25 mm^3	2 x 1 x 1 in

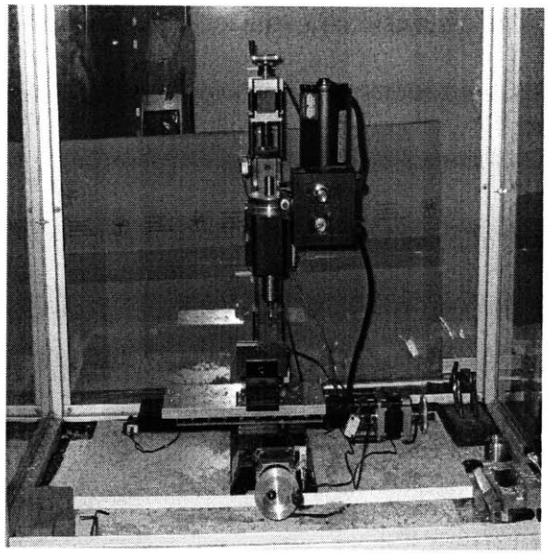

Figure 6-3: Desktop Milling Machine tool

6.2.2 Stepper motor controller

The machine tool had been shipped fitted with a Flashcut CNC control package, but this proved inadequate for our implementation. An alternative package was therefore developed for our system using the MMC – 4S multiple motor control systems The Motion Group[21] which uses a Cybernetics CY545 stepper motor controller microprocessor. The multiplexer section allows the CY 545 to control up to four step motor channels by multiplexing the motion signals between the channels. All actions of this system are controlled by high-level CY 545 commands. Serial (RS 232) port communication is used to communicate commands from the haptic simulation to the controller.

6.2.3 Haptic device

The haptic device used is a 5 DOF haptic device developed by Suzuki Japan. The device reports its position and posture, and can apply forces in three translational and 2 rotational degrees of freedom. The maximum force available, for safety reasons is 5N.

---

[21] The Motion Group Inc.: www.motiongroup.com

Figure 6-4 shows a picture of the device and Table 6-2 shows the datasheet. Physical inspection reveals an additional encoder at the end-effector from MTL, with product number: SE M-17-300AB.

Table 6-2: Haptic Device Stepper Motor Specifications

Diameter (mm)	Model	Rated Output (W)	Rated Voltage (V)	Rated speeds (rpm)	Gearhead
∅25 Series	NC-258102	22.5	24	500	SFP
Motor only; With Encoder (2000 pulse/ 3ch); With tachometer generator.					

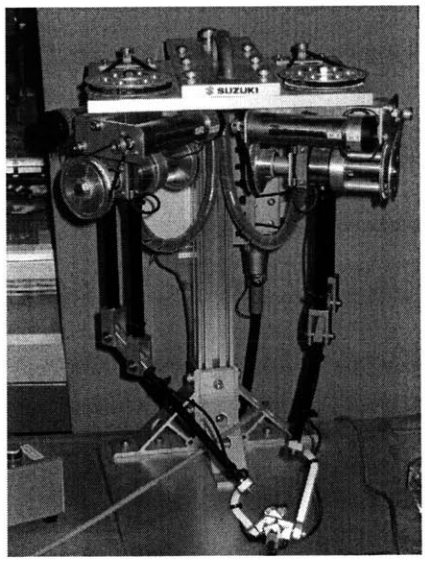

Figure 6-4: Haptic Device

6.2.4 Processor – Host computer

All the modules of the haptic stencil algorithm were implemented/ run on a Silicon Graphics Inc. OCTANE workstation. The program utilizes the dual 250MHz MIPS R10000 processors by allocating one processor (processor 1) to handle operating system needs and graphic display and machine tool control while the other processor (processor 2) handles interference analysis and force response calculations.

6.3 Overall Implementation

Figure 6-5 illustrates the components of the haptic stencil system and their operation.

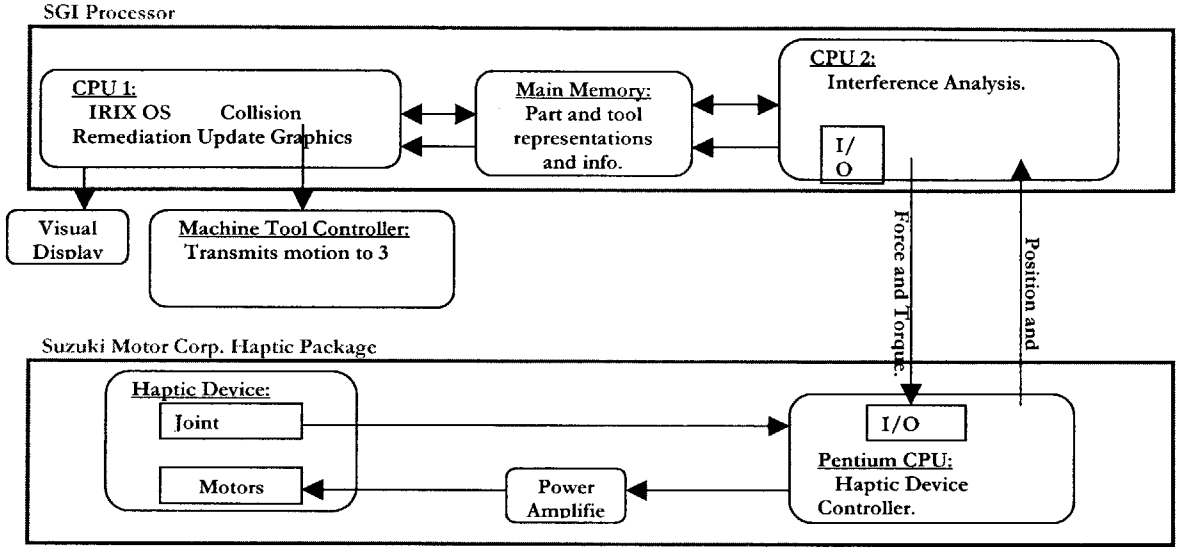

Figure 6-5: Overall System Implementation Schematic

The user manipulates the handle of the haptic device, which can move in all directions. Joint encoders on the haptic device measure the movements and can sense motion in all three translational as well as two rotational directions. The rotational axis coinciding with the axis of the handle is not measured. This data constitutes the position and orientation of the implicit equation BSP-tree defined object (tool object) in the simulation. The data is conveyed to the Silicon Graphics OCTANE computer via PIO shared memory cards in the haptic device controller and the OCTANE computer.

Based on the position and orientation information from the haptic device, the program running on processor 2 of the OCTANE performs interpenetration analysis between the two objects whose data structures are stored in the main memory of the OCTANE. The process also relays the position information to the graphics process running on processor 1. Based on the interpenetration analysis, the process on processor 2 formulates a force response that is fed back to the user via the haptic device (through the PIO shared

memory cards). The controller of the haptic device reads the force response and calculates the appropriate motor torque needed to create the desired force response. These signals are sent to the power amplifier that powers the motors at each joint of the haptic device. Processor 1 of the OCTANE reads the position and orientation information from the SGI main memory and uses that information to update the graphics display.

Tool and Part models are mathematically defined and represented. The input solid model for the part is in the Stereo-Lithography Tessellation Language (.STL) file format. The STL file format specifies the 3D object as a boundary representation constructed entirely of triangular facets. The particular format used in the haptic stencil implementation is laid out as follows:

```
FACET NORMAL {x-coordinate} {y-coordinate} {z-coordinate}
   OUTER LOOP
        VERTEX {x-coordinate} {y-coordinate} {z-coordinate}
        VERTEX {x-coordinate} {y-coordinate} {z-coordinate}
        VERTEX {x-coordinate} {y-coordinate} {z-coordinate}
   END LOOP
ENDFACET
```

Each loop consists of one triangle of the solid. The "FACET NORMAL" is the vector perpendicular to the surface of the triangle and the "VERTEX" lines of code are the coordinates of the three vertices of the triangle. The file ends with ENDSOLID {filename}. Since it uses triangles to describe every detail, the STL file format is suitable for implementing the triangulated object representation of the part. The STL file is then converted to a TOB file using simple conversion software

The TOB file format is used to convert the triangulated representation from the STL format to a triangulated surface point cloud representation. The conversion software

enables the user to specify the number of points for the point cloud. Then, the software places a point on each vertex and one point in the center of each triangle. It then places random points on the surfaces of the triangles until it reaches the specified number of points. The advantage of using the TOB file format is that it eliminates redundancy in the representation by not repeating common vertices.

The tool model, on the other hand, is created using C++ code that algorithmically follows the Constructive Solid Geometry (CSG) approach of unions (AND) and intersections (OR) of implicit equations to build the model. OpenGL and Open Inventor are used for graphic representation and visual update (or the GUI for the simulation).

The envisioned sequence of steps for operation would be:

1. Given part, identify sequence of machining steps required and corresponding tools.
2. For each machining step, define part geometry (for stencil) and tool to be used.
3. Input part and tool geometry files into anti-collision algorithm.
4. Fixture stock. Switch on controller and machine tool.
5. Start application and switch on haptic device.
6. Operator moves haptic probe → tool model moves in algorithm →algorithm tests for collision and returns appropriate tool position → tool position sent to machine tool, graphics updated and appropriate force response sent to operator.

Chapter 7: CONCLUSIONS & FUTURE WORK

Real time machine tool control using the haptic stencil setup has been successfully achieved. The setup exhibits desired stencil-like behavior. The haptic stencil interface is flexible, combines human judgment with computer accuracy and speed. Additionally, it provides an innovative method for rapid prototyping and the interface presented in this work can be tailored to a variety of applications especially in many areas of human-machine interactions.

It has been shown that a combination of collision detection and collision remediation can be successfully used to control a machine tool.

This chapter summarizes the primary contributions of this work and suggests avenues for further investigation.

7.1 Contribution:

The primary contribution of this work is the collision remediation methodology used to actively control the machine tool. In addition, the development and implementation of the overall system by tying together the collision detection methodology developed by Stephen Ho, the preliminary work in controlling a machine tool using a haptic device by Edmund Golaski, and the collision remediation and control methodology proposed, culminates into the Haptic stencil. The ability to achieve and meet haptic experience/simulation requirements yet executing a time-consuming operation by separating haptic module from the operation module and maintaining optimum communication rates between modules with appropriate filtering of data has been achieved using the Haptic stencil. Also, incorporating an independent collision remediation

module to enforce stencil operation without interfering with collision detection and the inherent collision remediation associated within haptic systems is a new approach.

7.2 Challenges

Some of the critical challenges that were encountered in the development of the haptic stencil include:

- 1000 Hz haptic refresh rate requirement;
- Collision remediation;
- Synchronization of the different modules of the setup;
- Quantification of system parameters.

As explained in chapters 3 and 4, the refresh rate requirement was met using Ho's collision detection methodology and interference analysis approach. Presently, the collision remediation module is directly linked to the collision detection module and generally performs at the same rate. However, in cases where large motion commands are sent to the machine tool in one go, the collision remediation module may slow down the overall refresh rate of the setup (this problem is discussed further in the following sub-section).

The development and implementation of the collision remediation approach for the haptic stencil went through different stages. First, testing the feasibility and/ or applicability of some of the methods that have been proposed in literature; the sticking methodology was found to provide non-intuitive haptic experience, while the slip methodology was not feasible for our implementation due to the fact that the amount of information required would make the algorithm computationally intensive. The final methodology was developed keeping in mind the collision detection module running it. As a result, an approach based on the philosophy behind the slip remediation approach but using the

information available from the collision detection and interference analysis module was used.

Due to the fact that the interface developed is a combination of different modules: collision detection, collision remediation, visual/ graphics update, and machine control, the overall implementation depends heavily on synchronization between these modules. Also, since the different processes have different work spaces and individual representations of the tool and part (for example the haptic space versus the graphic space), mapping and scaling of the tool and part in each of these representations is very important. In general, this was achieved by carefully considering the operation parameters of each of the modules and incorporating that information in the overall algorithm.

Quantification of system parameters has been done to some extent. However, in order to have a complete specifications sheet for the interface, further experiments would be required. Some of the parameters that could be better defined are the surface finish for different materials, geometric limitations (features on the part that are hard to machine), and repeatability.

7.3 Future Work

Incorporation of artificial friction and damping would improve haptic fidelity. Real time graphics update is another factor that would definitely add to the quality of the experience, providing visual feedback to the operator regarding the status of the operation and displaying the material removed. Furthermore, the present graphic user interface could be extended to allow the user to select/adjust different parameters (e.g. stiffness, upper limit for forces, even tool sizes).

Penalty-based speed limiting is an alternative approach for the speed control that could be investigated. This would increase the damping when velocities approached unacceptable values and effectively slow the user/operator down. One problem that could

be encountered with this method of compensation is that the user may be confused. On the other hand, this method of speed control may be more intuitive in the sense that the user would 'experience' the consequences of moving too fast or incorrectly rather than the computer just adjusting the velocity to an acceptable value within the simulation.

Alternatively, the kinematics of the haptic manipulator and machine tool could also be used to determine velocities of operation at which instabilities may occur. A thorough control analysis of the haptic control loop and machine tool control loop could be carried out to develop a more robust speed control methodology.

From a hardware point of view, better/faster communication ports would result in reduced time-lag between the simulation and the machine tool. Using online memory storage on the machine tool controller would decouple the refresh rates of the haptic device from the machine tool command execution. This would effectively enable the speed control of the simulation to be independent of the machine refresh rate. However, this would also result in the two loops operating at different rates and therefore not necessarily simultaneously.

Finally, this interface has been developed primarily for rapid prototyping applications. Other applications such as assembly operations or tele-operation (using remote communication) could also be investigated. For example, an application like Laser cutting using this interface would have the advantage that the "tool" kinematics play less of a role in the overall system performance and the feature size of the setup would no longer be limited by the size of the "tool" being used.

REFERENCES

[1] Adams 1999
R. J. Adams, B. Hannaford, "Stable Haptic Interaction with Virtual Environments", *IEEE Transactions on Robotics and Automation*, 15(3), June 1999.

[2] Adachi 1995
Y. Adachi, T. Kumano, K. Ogino, "Intermediate Representation for Stiff Virtual Objects", *IEEE*, 1995.

[3] Basdogan, Ho, Srinivasan 1999
C. Basdogan, C. Ho, M.A. Srinivasan, "Efficient Point-Based Rendering Techniques for Haptic Display of Virtual Objects", *Presence,* Vol. 8, No. 5, MIT Press, pp. 477-491, 1999.

[4] Basdogan & Srinivasan 1997
C. Basdogan, M. A. Srinivasan, "Haptics in virtual environments: Taxonomy, research status, and challenges", *Computers and Graphics*, 21(4), pp. 393-404, 1997.

[5] Basdogan, Ho & Srinivasan 1997
C. Basdogan, C. Ho, M. A. Srinivasan, "A Ray Based Haptic Rendering Technique for Displaying Shape and Texture of 3D Objects in Virtual Environments", *Proceedings of the ASME Dynamic Systems and Control Division: 1997 ASME Intl. Mechanical Engineering Congress and Exposition*, November 1997.

[6] Balasubramaniam 1999
M. Balasubramaniam, "Tool selection and path planning for 3-Axis rough machining", *S. M. Thesis*, Massachusetts Institute of Technology, 1999.

[7] Balasubramaniam et al 2002
M. Balasubramaniam, S. Ho, S. Sarma, Y. Adachi, "Generation of collision-free 5-axis tool paths using a haptic surface", *Computer Aided Design*, v 34, n 4, p 267-79, April 2002.

[8] Burdea 1996
G. C. Burdea, "Force and Touch Feedback for Virtual Reality", 1996

[9] Cohen 1995
J. D. Cohen, M. C. Lin, D. Manocha, M. K. Ponamgi, "I-COLLIDE: An Interactive and Exact Collision detection system for Large-Scale Environments", *Proceedings of ACM Int. 3D Graphics Conference*, 1995.

[10] Colgate 1994
J. E. Colgate, G. Schenkel, "Passivity of a Class of Sampled-Data Systems: Application to Haptic Interfaces", *Proceedings of the American Control Conference*, June 1994.

[11] Clover 1997
C. L. Clover, G. R. Luecke, J. J. Troy, W. A. McNeely, "Dynamic Simulation of Virtual Mechanisms with Haptic Feedback Using Industrial Robotics Equipment", *Proceedings of the IEEE International Conference on Robotics and Automation*, April 1997.

[12] Cohen 1995
J. D. Cohen, M. C. Lin, D. Manocha, K. Ponamgi, "Interactive and Exact Collision detection for Large-scale environments", Proceedings of the ACM International 3D Graphics Conference, 1995.

[13] de Berg 1997
M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, "Computational Geometry: Algorithms and Applications", Springer Verlag, 1997.

[14] Dobkin 1990
D. P. Dobkin & D. G. Kirkpatrick, "Determining the separation of preprocessed polyhedra – A unified approach", *Proc. 17th International Colloq. Automata Lang. Program.*, vol. 443 of *Lecture Notes Comp. Sci.*, pp 400-413, Springer-Verlag, 1990.

[15] Foley 1996
J. D. Foley, A. Dam, S. K. Feiner, J. F. Hughes, "Computer Graphics – Principles and Practice", Addison-Wesley Publication Co., 1996.

[16] Goertz 1954
R. Goertz and R. Thompson, "Electronically controlled manipulator", *Nucleonics*, pp. 46-47, 1954.

[17] Golaski 2001,
E. W. Golaski, "Direct Haptic Control of a 3-Axis Milling Machine", *S. M. Thesis,* Massachusetts Institute of Technology, June 2001.

[18] Gottschalk 1996
S. Gottschalk, M. C. Lin, D. Manocha, "OBB-Tree: A Hierarchical structure for Rapid Interference detection", *Proc. of ACM SIGGRAPH '96 Conference Proceedings*, 1996.

[19] Held 1998
M. Held, J. T. Klosowski, J. S. B. Mitchell, H. Sowizral, K. Zikan, "Efficient collision detection using bounding volume hierarchies of k-DOPs", *IEEE Transactions on Visualization and Computer Graphics,* Vol. 4, No. 1, 1998.

[20] Ho 1999
S. Ho, "Real-time detection of geometric interference: application to full-body 5-axis haptics", *S. M. Thesis,* Massachusetts Institute of Technology, June 1999.

[21] Hogan 1993
N. Hogan, H. Krebs, J. Charnnarong, P. Srikrishna, A. Sharon, "MIT-MANUS: a workstations for manual therapy and training II", *Proceedings SPIE Conference on Telemanipulator technology*, SPIE, 1833, pp. 28-34, 1993.

[22] Hollerbach 1997
J. M. Hollerbach, E. Cohen, W. Thompson, et al, "Haptic interfacing for Virtual prototyping of mechanical CAD designs", *ASME Proc. Design for Manufacturing Symposium*, September 1997.

[23] Immersion
Immersion Corporation, www.immersion.com

[24] Jimenez 2000
P. Jimenez, F. Thomas, C. Torras, "3D Collision Detection: A Survey", Computers & Graphics, vol. 25, pp 269-285, 2000

[25] Lin 1993
M. C. Lin, "Efficient Collision Detection for Animation and Robotics", *Ph.D. Thesis*, University of California, Berkeley, December 1993.

[26] Lin & Canny 1991
M. C. Lin & J. F. Canny, "Efficient algorithms for incremental distance computation", *IEEE Conference on Robotics & Automation*, pp 108 – 1014, 1991.

[27] Lin & Gottschalk 1998
M. C. Lin, S. Gottschalk, "Collision Detection between Geometric Models: A Survey", *Proceedings of IMA Conference on Mathematics of Surfaces*, 1998

[28] Massie & Salisbury 1994
T. H. Massie, J. K. Salisbury, "The PHANTOM Haptic Interface: A Device for Probing Virtual Objects", Proceedings of the ASME Winter Annual Meeting, Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, November 1994.

[29] Massie 1996
T. Massie, "Initial Haptic Explorations with the Phantom: Virtual Touch through Point Interaction", *S.M. Thesis*, Massachusetts Institute of Technology, February 1996.

[30] Minsky 1995
M. D. R. Minsky, "Computational Haptics: The Sandpaper System for Synthesizing Texture for a Force-Feedback Display", *Ph.D. Thesis*, Massachusetts Institute of Technology, June 1995.

[31] Minsky et al 1990
M. Minsky, M. Ouh-Young, O. Steele, F. B. Brooks, M. Behensky, "Feeling and seeing: Issues in Force display", ACM Computer Graphics, 24-2, pp 235-243, 1990.

[32] Mirtich 1998
B. Mirtich, "V-Clip: Fast and Robust Polyhedral Collision Detection", *ACM Transactions on Graphics*, pp. 177-208, July 1998.

[33] Moore & Wilhelms 1988
M. Moore, J. Wilhelms, "Collision detection and response for Computer animation", *Computer Graphics*, vol. 22, No. 4, pp. 289-298, 1988.

[34] Sankaranarayanan 2003
G. Sankaranarayanan, S. Weghorst, "Role of Haptics in Teaching Structural Molecular Biology", IEEE Proc. of the11th Symp. On Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS'03), 2003

[35] Sensable
Sensable Technologies, www.sensable.com

[36] Schlaroff & Pentland
S. Schlaroff, A. Pentland, "Generalized implicit functions for Computer Graphics", *Computer Graphics*, Vol. 25, No. 4, pp. 247-250, July 1991.

[37] Sheridan 1992a
T. Sheridan, "Tele-robotics, Automation, and Human Supervisory Control", *MIT Press*, Cambridge, MA, 1992.

[38] Sheridan 1992b
T. Sheridan, "Musings on Telepresence and Virtual Presence", *Presence-Teleoperators and Virtual Environments*, Vol. 1, No. 1, MIT Press, pp. 120-126, 1992.

[39] Sitti 1999
M. Sitti, "Teleoperated 2-D Micro/Nanomanipulation Using Atomic Force Microscope", *Ph.D. Thesis*, Dept. of Electrical Engineering, University of Tokyo, Tokyo, Sept. 1999.

[40] Sriram 2001
K. Sriram, "Rapid detection of shallow penetration between non-convex polyhedra", *S. M. Thesis,* Massachusetts Institute of Technology, Sept. 2001.

[41] Sutherland 1965
I. E. Sutherland, "The Ultimate Display", *Proceedings of IFIPS Congress*, New York, Vol. 2, pp. 506-508, May 1965.

[42] Yokokohji et al 1996
Y. Yokokohji, R. L. Hollis, T. Kanade, "What you see is what you can feel – development of a visual/haptic interface to virtual environment", Proc. of IEEE Virtual Reality Annual Intl. Symp., pp 46-53, Las Alamitos, CA, 1996.

[43] Zilles & Salisbury 1995
C. B. Zilles, J. K. Salisbury, "A constraint-based god-object method for haptic display", *IEEE International Conference on Intelligent Robots and System, Human Robot interaction, and Co-operative Robots*, IROS, 3, 146-151, 1995.