PHYSICALLY INTERACTING WITH FOUR DIMENSIONS

Hui Zhang

Submitted to the faculty of the Graduate School

in partial fulfillment of the requirements

for the degree

Doctor of Philosophy

in the Department of Computer Science

Indiana University

December 2008

Accepted by the Graduate Faculty, Indiana University, in partial fulfillment of the requirements of the degree of Doctor of Philosophy.

Doctoral Committee

Andrew J. Hanson, Ph.D.
(Principal Advisor)

Chen Yu, Ph.D.

David B. Leake, Ph.D.

Yuqing Wu, Ph.D.

April 28, 2008

To my wife, my daughter Jasmine and our loving parents.

Acknowledgements

I am exceedingly grateful to my supervisor, Dr. Andrew Hanson for his support and advice over the years. Especially near the "the end," he was crucial in helping me to produce this thesis. A lot of thanks go to Dr. Chen Yu; I have been lucky to have him on my PhD committee, and lucky to have been working with him on a couple of very exciting research projects in the department of Psychology and Brain Science. To my entire committee, Dr. David Leake and Dr. Yuqing Wu, I am very thankful for much inspiration and guidance from them.

Many friends have been important over these years in Lindley Hall. Those in this department include: Sidharth Thakur, Yin Wu, Xin Xiang, Yi Huang, Qian Wang, and Yiwen Zhong. Those who have been helping and encouraging me even though they have been geologically far from me include: Jianguang Weng, Qibo Zhu, Lingxuan Hu, Can Zheng, and Dong Xin. You are all treasures in one way or another, and I have been gifted to know you all.

For financial support, I would like to thank the National Science Foundation, the Computer Science Department, the Psychology and Brain Science Department, and Indiana University, Bloomington.

My family has been very supportive and loving. I thank my wife Ling Jiang, our daughter Jasmine,

and our loving parents. It is to them that this thesis is dedicated.

Abstract

People have long been fascinated with understanding the fourth dimension. While making pictures of 4D objects by projecting them to 3D can help reveal basic geometric features, 3D graphics images by themselves are of limited value. For example, just as 2D shadows of 3D curves may have lines crossing one another in the shadow, 3D graphics projections of smooth 4D topological surfaces can be interrupted where one surface intersects another.

The research presented here creates physically realistic models for simple interactions with objects and materials in a virtual 4D world. We provide methods for the construction, multimodal exploration, and interactive manipulation of a wide variety of 4D objects. One basic achievement of this research is to exploit the free motion of a computer-based haptic probe to support a continuous motion that follows the *local continuity* of a 4D surface, allowing collision-free exploration in the 3D projection. In 3D, this interactive probe follows the full local continuity of the surface as though we were in fact *physically touching* the actual static 4D object.

Our next contribution is to support dynamic 4D objects that can move, deform, and collide with other objects as well as with themselves. By combining graphics, haptics, and collision-sensing physical modeling, we can thus enhance our 4D visualization experience. Since we cannot actually

place interaction devices in 4D, we develop fluid methods for interacting with a 4D object in its 3D shadow image using adapted reduced-dimension 3D tools for manipulating objects embedded in 4D. By physically modeling the correct properties of 4D surfaces, their bending forces, and their collisions in the 3D interactive or haptic controller interface, we can support full-featured physical exploration of 4D mathematical objects in a manner that is otherwise far beyond the real-world experience accessible to human beings.

Table of Contents

Acknowledgements **v**

Abstract **vii**

1 Introduction **1**

1.1 Mathematical Visualization 1

1.2 Contributions of the Thesis 3

1.2.1 4D Visualization 3

1.2.2 Computer Haptics 6

1.2.3 Mathematics 8

1.3 Overview of the Thesis 10

2 4D Visualization **13**

2.1 4D Visualization in the Traditional Media 13

2.2 Computer 3D Printing and Concrete Mathematics 15

2.3 Computer Systems for 4D Visualization 17

2.4 Film and Video 20

3 KnotExplore: Introducing Dimensions 22

3.1 Dimensions . . . 22

3.1.1 Fechner, Plato, and Shadow Images . . . 22

3.1.2 Abbott, Illusions and Miracles . . . 23

3.1.3 Dimensional Progressions . . . 26

3.2 Overview of Shadow Manipulation Scenario . . . 30

3.2.1 2D Example . . . 30

3.2.2 Motivation . . . 32

3.3 Details of Implementation Models . . . 36

3.3.1 Shadow Space Force Modeling . . . 37

3.3.2 Collision Avoidance and Repulsive Forces . . . 38

3.3.3 Multimodal Navigation . . . 42

3.3.3.1 Haptic Feedback . . . 42

3.3.3.2 Auditory Feedback . . . 44

3.3.4 Selecting Viewable and Touchable Knot Images . . . 46

3.3.5 Representation and Display . . . 48

3.3.5.1 Rendering 2D Knot Crossing-Diagrams . . . 49

3.3.5.2 Rendering 3D Smooth Knots . . . 49

3.4 User Applications and Feedback Results . . . 51

3.4.1 Knot Exploration Interface . . . 51

3.4.2 Motor Assistance . . . 52

3.4.3 Pseudo-Haptic 2D Knot Diagrams . . . 52

3.4.3.1 Mouse Pointer Control 55
3.4.3.2 Constrained Grid Drawing Interface 55
3.4.3.3 Interacting with Mathematical Knots and Links 55
3.5 Summary 57

4 Touching the Fourth Dimension **61**

4.1 Introduction 61
4.2 Motivation 63
4.3 Haptic Methods 71
4.3.1 Simulated Sticky Stylus 71
4.3.2 Constrained Navigation on the Local Surface Model 73
4.3.3 Adding Force Suggestions 75
4.3.4 Experiments and Results 78
4.4 Auditory Cues 79
4.4.1 The Problem 80
4.4.2 The Solution 81
4.5 4D Haptic Rolling Manifold 82
4.5.1 Maximizing Viewable and Touchable Aspects at Each Step 82
4.5.2 Rotating between Facets 83
4.5.3 Tunneling the Probe to Target the Surface 84
4.5.4 Fixing the Stylus using a “Rubber Band Line” 85
4.6 User Environment and Further Examples 87

4.7 Summary 90

5 Collisions in Four Dimensions **93**

5.1 Intersecting Shadows from Higher Dimensions 93

5.1.1 2D Example 93

5.1.2 3D Example 94

5.1.3 Various Modes for Studying Intersecting Shadow Images 95

5.1.3.1 Visual Modes 95

5.1.3.2 Applying General Rotations 100

5.1.3.3 Haptic Method 101

5.2 Geometry in Higher Dimensions 102

5.2.1 Vector Operations and Points in 4D 102

5.2.2 Distances in 4D 105

5.3 Lifting in Four Dimensions 111

5.3.1 Constrained Motions and Special Effects 112

5.3.2 4D Collision Detection between Rigid Objects 113

5.3.2.1 Collision Response 114

5.3.2.2 Pair Reduction 115

5.3.3 Example: Lifting a Chain in Four Dimensions 115

5.4 Conclusion 117

6 Editing 4D Cloth-like Objects **118**

6.1 Motivation 118

6.2 Physically Interacting with 3D Shadows of 4D surfaces 120

6.2.1 Collision Mechanism for Non-rigid 4D Surfaces 121

6.2.1.1 Collision in Four Dimensions 121

6.2.1.2 4D Collision Avoidance 123

6.2.2 4D Cloth-Environment Simulation 124

6.2.2.1 4D Mass-Spring System 124

6.2.2.2 Choosing the Shadow Plane 126

6.2.2.3 Forces 126

6.2.2.4 Integration 130

6.2.2.5 Haptic 4D Cloth Rendering 130

6.2.3 Examples 131

6.2.3.1 Visualizing 4D Collisions. 131

6.2.3.2 Tightening the 4D Spun Trefoil 131

6.2.3.3 Deforming the 4D Torus 134

6.2.3.4 Physical Lifting in Four Dimensions 134

6.2.4 Multimodal Exploration of the Fourth Dimension 136

6.3 Implementation Environment and Preliminary User Studies 136

6.4 Summary 139

7 Example: A Framework for Physically-Based Sphere Eversion 141

7.1 Introduction 141

7.2 Circles Cannot Be Everted 142

7.2.1 Physically Modeling the 2D Circle 144

7.2.1.1 Raising the Sides of a 2D Circle 145

7.2.1.2 The Mesh 145

7.2.2 Physically Interacting with a 2D Circle 146

7.2.2.1 Dynamics and Forces 146

7.2.2.2 Integration 148

7.2.3 Rules 149

7.2.3.1 Self-Intersection is Legal 149

7.2.3.2 Pinch-Points are Illegal 149

7.2.4 Results 154

7.3 Turning a 2D Circle Inside Out in 3D 154

7.3.1 Interacting with a 3D String 154

7.3.2 A 2-ribbon Problem 156

7.3.3 Turning a 2D Circle Inside Out 159

7.4 Sphere Eversion Using a Partial Model 160

7.5 Summary 160

8 Conclusions and Future Work **165**

A Rotations in Four Dimensions **169**

B Mathematical Background of Sphere Eversion **172**

C History of Sphere Eversion **174**

C.1 A “Turning Number” for Surfaces 174

C.2 Smale's Proof 174

C.3 Subsequent Attempts at Eversion 175

Bibliography **177**

List of Figures

1.1 4D visualization 5

1.2 Computer haptics 6

1.3 Applications of computer haptics 8

1.4 SensAble's FreeForm ®modeling system 8

2.1 Computer 3D printing and concrete mathematics 16

2.2 Geomview 19

2.3 Meshview 19

2.4 Knotplot 20

2.5 Film and video on 4D 21

3.1 Plato's allegory of the cave 24

3.2 Abbott's own drawing of the house of "A Square" 25

3.3 Dimension progression 28

3.4 Sun clock 31

3.5 2D and 3D representations of a trefoil knot 31

3.6 2D shadow diagram, 2-$\frac{1}{2}$ knot diagram and 3D image 33

3.7 Shadow manipulation scenario 35

3.8 Mental model during 2D shadow editing 35

3.9 Dynadraw 40

3.10 Force model for 2D sketching 41

3.11 Haptic knot creation 41

3.12 $knot_9^6$, $knot_8^{18}$, $knot_9^{33}$ 42

3.13 Haptic exploration of knots 43

3.14 Visual cues and sound cues 45

3.15 Sound cues during haptic exploration 46

3.16 Choices of knot projection 47

3.17 Knot viewpoint quality measure 47

3.18 Tubing model for generating thickened curves 50

3.19 KnotExplore 53

3.20 Motor assistance interface 53

3.21 Cursor rendering in pseudo haptic interface 54

3.22 Constrained 2D drawing interface 56

3.23 Relaxation of mathematical links 58

3.24 Screen images of knot manipulation 59

4.1 Geometry locator panel in Meshview 64

4.2 Physical model representing 3D projection from 4D 66

4.3 Intersecting surfaces resulting from the projection of a 4D embedded object 67

4.4 Surface constraint force model 72

4.5 Haptic exploration on intersecting 3D projection from 4D 74

4.6 Haptic exploration of the fourth dimension 75

4.7 Kalman filtering 78

4.8 Auditory cues during haptic exploration of the fourth dimension 81

4.9 Maximizing viewable and touchable aspects during 4D touch 83

4.10 Tunneling the probe to target the surface 84

4.11 Tunneling the probe using opacity modulation and screen-door transparency 85

4.12 Tunneling the probe to more complex 4D topological surfaces 86

4.13 Screen image of a haptic interface to maximize viewable and touchable aspects . . 88

4.14 Overview of the multimodal 4D exploration interface 89

4.15 Exploration of the 4D torus 89

4.16 Exploring the ordinary torus using geodesic path 90

4.17 Touching the 4D Torus with transparent view 91

5.1 Intersecting 2D shadow images 94

5.2 Intersecting 3D shadow images 95

5.3 Direct 3D projection from 4D 97

5.4 Screen door rendering of 3D projection from 4D 98

5.5 Transparent rendering of 3D projection from 4D 100

5.6 Depth coloring and cutting away view of 3D projection from 4D 101

5.7 Applying general rotation to 3D shadow images 102

5.8 Real 4D link structure with general rotation applied to its 3D shadow images 103

5.9 Haptic method of exploring intersecting shadow images 104

5.10 Closest points between 4D line segments 106

5.11 Partitioning of the st-plane by triangle domain D 110

5.12 Various level curves $Q(s,t) = V$ 112

5.13 The haptic interface for lifting a 4D chain element 116

5.14 The haptic interface for physically disassociating the 3D projected images of two unlinked 4D embedded objects. 116

6.1 Mental model during 3D shadow editing 121

6.2 4D collisions 122

6.3 Flattened ($w = 0$) 4D cloth behaves like 3D cloth 127

6.4 A 2D point trapped inside a 2D ring 128

6.5 A point embedded in 3D can escape from the 2D ring after rotation 129

6.6 Visualizing 4D collisions 132

6.7 Tightening the 4D spun trefoil 133

6.8 Deforming the 4D embedding of the torus 135

6.9 Threading a 4D ribbon into 4D cloth 137

6.10 Threading a 4D ribbon into 4D sphere 138

7.1 Thurston's sphere eversion 142

7.2 Circles can't be everted 143

7.3 Standard and inverted circles 144

7.4 The interconnections of cloth springs 146

7.5 The interconnection of three types of springs for a 2D circle 147

7.6 Self-intersection is legal in immersion 149

7.7 Super-elastic bending springs 150

7.8 Adjustment of a "super-elongated" or "super-shrunk" bending spring 153

7.9 No cusps when poles pulled through 155

7.10 Playing with a 3D string. 157

7.11 Playing with a 3D string using Thurston mode. 158

7.12 Modeling a 2D circle as two poles connected by two 3D thickened strings. 159

7.13 Turning a 2D circle inside out by pushing the poles. 161

7.14 Turning a 2D circle inside out using Thurston's motion. 162

7.15 Turning a partial model of the sphere inside out, part 1. 163

7.16 Turning a partial model of the sphere inside out, part 2. 164

8.1 2D/3D Reidemeister moves 166

8.2 3D/4D Reidemeister moves 167

B.1 A continuous deformation (homotopy) of a doughnut into a coffee cup 172

C.1 Boy's surface 175

List of Tables

4.1 Local Model vs Overall Model 79

6.1 Participants' evaluation of knot exploration system 139

1

Introduction

1.1 Mathematical Visualization

Mathematical visualization is the art of creating a tangible experience with abstract mathematical objects and concepts. While this process has been a cornerstone of the mathematical reasoning process since the times of the ancient geometers, the advent of high-performance interactive computer graphics systems has opened a new era whose ultimate significance can only be imagined.

Typical geometric problems of interest to mathematical visualization applications involve both static structures, such as real or complex manifolds, and changing structures requiring animation, such as sphere eversion. In practice, the emphasis is on manifolds of dimension two or three embedded in three or four-dimensional spaces due to the practical limitations of holistic human spatial perception - it is extremely challenging to construct intuitively useful images of anything more complicated! General approaches to visualizing N-dimensional spaces are at best piecemeal, so that algebraic manipulations often remain our most powerful tool for high dimensions. Nevertheless, despite the apparent limitations of visual representations, their utility is far from being completely exploited;

we may still gain significant intuitive value by pushing our visual understanding of relatively simple geometric objects as far as our imagination can take us.

Our goal is to show the nature of the inter-relationship between mathematics and computer science, especially computer graphics and computer haptics. In this thesis research work, we adopt for the most part a computer scientist's perspective on the progress, techniques, and prospects of mathematical visualization, emphasizing those areas of 3D and 4D geometry where interactive paradigms are of growing importance. Without demanding detailed mathematical expertise of the reader, we first present a selection of the domains with which we are familiar: we start with prototyping a family of methods in 3D as a tutorial education system, describe some of the critical visualization problems involved in 3D, and then discuss how various techniques can be extended to approach the solutions of our target problems in four dimensions.

The thesis begins with some general background and then turns its attention to some of the visualization methods that have been used to bring computer graphics technology to bear on mathematical problems of low-dimensional topology and geometry. The remaining sections discuss the design philosophies for a 4D visualization system generalized from 3D concepts along with various research approaches and prospects for the future. Examples of computer-generated images are supplied throughout, and sources of additional background information on visualizable mathematics can be found in the Appendices.

1.2 Contributions of the Thesis

This thesis research, combining both interactive computer graphics and mathematical visualization, has resulted in the creation of useful and powerful techniques for the exploration of numerous problems in mathematics and related fields, such as the physics of 3D knots, 2D manifolds embedded in 4D, 4D knotted spheres, and sphere eversion. This thesis is novel in the sense that it is concerned with both the development of the computer aided visualization techniques themselves, a domain within the bounds of computer science, as well as the application of those techniques to areas outside of traditional computer science.

The remainder of this section highlights the scientific contributions of the thesis work to the areas of study in computer science (primarily interactive computer graphics) and mathematics. In addition to these "main-stream" disciplines, the thesis research has also found applications in psychology and cognitive science, some of them quite far removed from traditional computer science or mathematics.

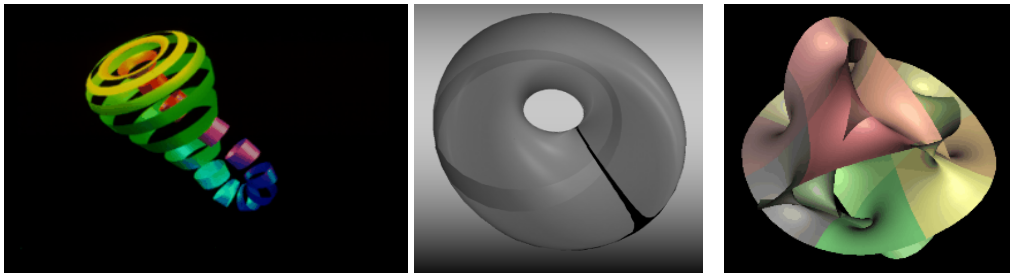
1.2.1 4D Visualization

This thesis contributes to the technology of 4D visualization in several important ways. The first of these contributions concerns the issue of *physics in four dimensions*. 4D visualization systems date back to the 1970's, when Thomas Banchoff and his collaborators pioneered a mathematical visualization program at Brown University. They achieved interactive, real-time geometrical visualizations of surfaces projected from 4D to 3D with a custom-built matrix multiplier and a fast-refresh vector graphics display for wire-frame modeling. They also explored techniques such as enhancing

depth perception by rotating the resulting 3D projective object at a constant angular velocity, and produced many animations of classical objects such as projective planes and tori projected from 4D to a 3D graphics depiction (for example, Figure 1.1(a) shows an image of a classic Klein bottle generated by this group that utilizes two separate visualization methods: color coding keyed to 4D depth and surface rendering with alternating transparent ribbons to reveal internal structure of the self-intersecting surfaces).

At the University of North Carolina/Chapel Hill, the Fourphront system developed by David Banks is another interactive system for the study of surfaces in 4D [Banks(1992)]. This system, which ran on the high-speed massively parallel Pixel-Planes 5 graphics engine, provided user control of transparency, depth cueing, intersection highlights, and 2-sided paint as well as supporting user control of rotation and translation. An illustration of Fourphront's alternative approach to the Klein bottle display is shown in Figure 1.1(b). Among other representative 4D visualization systems, a 4D viewer with a different philosophy is the MeshView program designed by Hui Ma and Andrew J. Hanson of Indiana University. This system supports the 4D surface mesh data format, along with a high-speed mouse-driven 4D rotation interface and a utility for locating particular points on a projected surface relative to the abstract parametric mesh coordinates. Figure 1.1(c) illustrates a MeshView display of the $n = 4$ case of a closed-form construction developed by Andrew J. Hanson for representing the complex "Fermat" equations $(z_1)^n + (z_2)^n = 1$ [Hanson(1994a)]; the surface is projected from 2 complex dimensions to 3 real dimensions from any desired viewpoint. Large families of complex surfaces can be displayed interactively in MeshView using this technique.

Typically, 4D visualization methods as cited above employ a projection to 3D as the first step; this



(a) (b) (c)

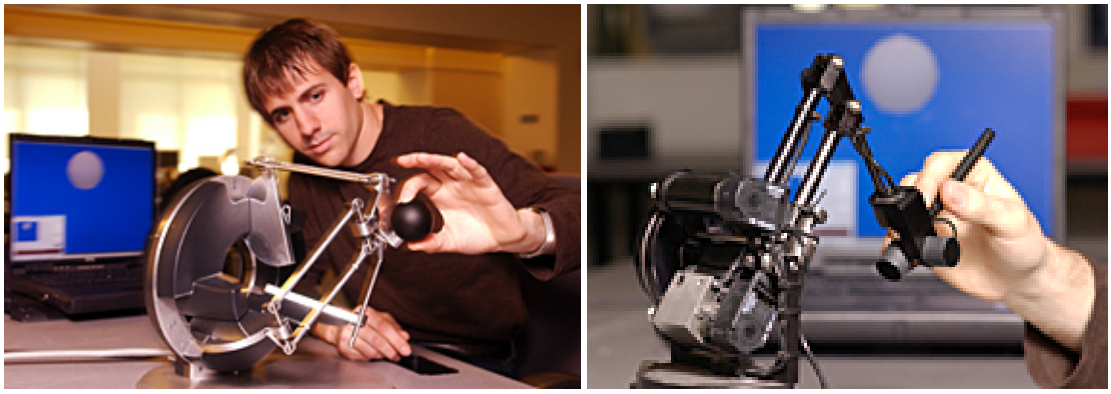
Figure 1.1: (a) Image of a Klein bottle with color-coded 4D depth and ribbon slicing to reveal interior structure. (T. Banchoff and N. Thompson, Brown University). (b) Semi-transparent Klein bottle displayed by Fourphront contains a Möbius band (opaque). The projection of the surface to 3D contains an intersection line, highlighted in black. (D. Banks, University of North Carolina and NASA's Langley Research Center). (c) The $n = 4$ Fermat surface projected to 3D using the MeshView interactive 4D viewer. The colors encode the relative complex phase of different patches of the surface. (H. Ma and A. Hanson, Indiana University.)

helps the viewer to identify salient global features of the four-dimensional object, and provides structural continuity when rotating either the object's (rigid) 3D projection, or, with more difficulty, the 4D orientation matrix, which causes non-rigid deformations in the 3D projection. Our thesis suggests that it is important to go beyond this to fully understand four dimensions. That is, it is necessary to produce dynamic pictures that respect the *physics* in four dimensions.

We are therefore concerned with 4D objects that have a virtual physical existence as real as any 3D entities that we see and touch in our everyday life; if the results of manipulation or interaction are to be believable, they must be consistent, e.g., with correct 4D physical modeling. Our approach, therefore, generalizes and extends traditional 3D interactive, physically-based modeling systems (see, e.g., [Terzopoulos and Witkin(1988)]).

1.2.2 Computer Haptics

Haptics, the science of touch, lets computer users interact with virtual worlds by feel. Some commercial computer games already use haptic devices, like the force-feedback steering wheels that react to driving on bumpy virtual roads. Gaming was one of the first applications of computer based haptics. Figure 1.2 shows a force-enabled version of virtual ping-pong called "Haptic Battle Pong," programmed by two students in Dr. Kenneth Salisbury's experimental haptics course at Stanford University. Interest in that game caused an Internet traffic jam that shut down the haptic interface manufacturer's website for a day. But haptics isn't all fun and games. Haptics can not only simulate the impact of a golf club hitting a ball, but also the response of an individual carbon nanotube in an atomic force microscope [Dobrinov and Eichhorn(2007)] and the texture of food [Iwata et al.(2004)Iwata, Yano, Uemura, and Moriya].



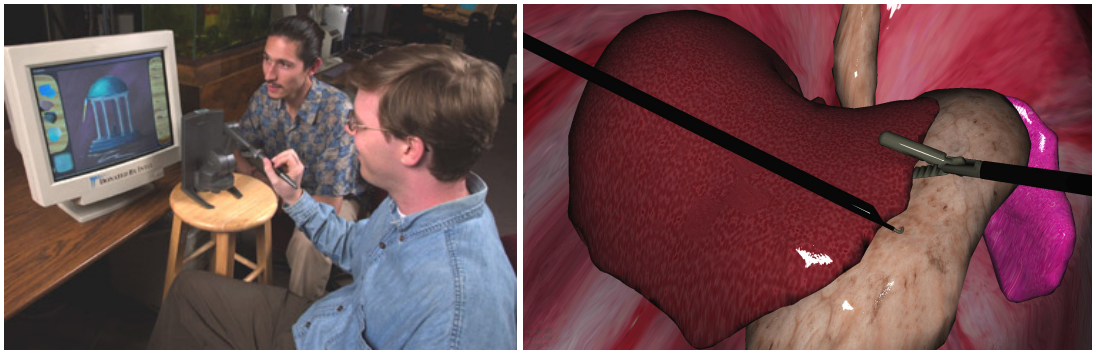
(a) (b)

Figure 1.2: (a) Graduate student Francois Conti demonstrates a device that lets computer users feel and manipulate the objects depicted on their screens. This spider-like robot can take tactile "pictures." In the laboratory of Kenneth Salisbury, professor of computer science and of surgery, Conti and other researchers explore haptics – the science of touch. Photo: L.A. Cicero (b) The computer communicates sensations through interfaces such as the PHANTOM^{TM} Haptic Interface, produced by SensAble Technologies, Inc. of Woburn, Mass.

Using haptic technology in arts and design, William Baxter of the University of North Carolina at Chapel Hill developed a novel system that utilizes a PHANTOM^{TM} device to simulate the sensation of painting on a virtual surface using paintbrushes [Baxter et al.(2001)Baxter, Scheib, and Lin](see Figure 1.3(a)). Computer based haptics is exploited in simulated surgery (see Figure 1.3(b)). Just as commercial pilots train in flight simulators before they're unleashed on real passengers, surgeons can practice their first incisions without actually cutting anyone [Chen et al.(2006)Chen, Barner, and Steiner].

Other representative efforts include touch-sensitive interfaces for 3D CAD systems, e.g., SensAble's FreeForm ®modeling system, which applies a pen-like device connected to a flexible arm. Holding this pen, the user can sculpt a block of virtual clay, using a virtual sculpting tool that is connected to the pen. The surprising element is that the pen gives a force feedback: the more clay the user wants to move, the more force he has to apply (see, e.g. Figure 1.4).

As a second contribution of this thesis research, we exploit computer-based haptic probes to support physical interactions with the shadow images of 4D objects. By combining graphics and collision-sensing haptics, as well as modeling the correct properties of 4D surfaces, bending forces, and their collisions in the 3D haptic controller interface, we can support full-featured physical exploration of 4D mathematical objects in a manner that is otherwise far beyond the experience accessible to human beings. As far as we are aware, this thesis reports the first interactive system with force-feedback that provides "4D haptic visualization" permitting the user to touch rigid 4D objects and manipulate 4D cloth-like objects.



(a) (b)

Figure 1.3: (a) Haptic painting system setup, a 3D stylus and the space bar provide the simple user interface. ©University of North Carolina at Chapel Hill. (b) Using surgical simulation, doctors can hone fine motor skills, practice procedures, and train other professionals without the risk in the operating room. ©Rensselaer Polytechnic Institute.

Figure 1.4: SensAble's FreeForm ®modeling system.

1.2.3 Mathematics

In his autobiography [1] Bertrand Russell characterizes mathematics as follows: "It seems to me now that mathematics is capable of an artistic excellence as great as that of any music, perhaps greater; not because the pleasure it gives (although very pure) is comparable, either in intensity or in the number of people who feel it, to that of music, but because it gives in absolute perfection that

[1] See Bertrand Russell (1872-1970), Autobiography, George Allen and Unwin Ltd, 1967, vol. 1(1872-1914), pp.158-159

combination, characteristic of great art, of godlike freedom, with the sense of inevitable destiny; because, in fact, it constructs an ideal world where everything is perfect and yet true." Actually the perception of mathematical research as an artistic discipline has a long history and a significant number of today's mathematicians share this view. In contrast, however, Russell himself dedicated large parts of his life to defending logicism, that is, the view that mathematics is reducible to logic, and together with Alfred Whitehead he proposed in his influential *Principia Mathematicae* an axiomatic system upon which to build all of mathematics.

While the mathematical content of this thesis is elementary compared to the state of the art in mathematical research, mathematicians can benefit from the interactive techniques presented here. Part of this is empowering them to be able to construct, manipulate and transform their objects of study easily. Methods for topological drawing can be used by mathematicians as an aid in their research. The computer system acts as a bookkeeping and consistency checking device. Some operations and calculations formerly done by hand may now be performed easily by computer. This is similar to other work that uses computers to access the complexity of mathematics, for instance, computer algebra systems (e.g., [Monagan(1996), Wolfram(1991)]), interactive topological drawing systems (e.g., [Scharein(1998)]), multimodal interfaces to access mathematics (e.g., [Fitzpatrick and Karshmer(2004), Sánchez and Sáenz(2005)]). It is important to emphasize that intelligent intervention by people with some understanding of the subject area is often crucial to the success of such systems.

1.3 Overview of the Thesis

A brief history and review of 4D visualization is given in Chapter 2. Chapter 3 introduces KnotExplore, our vehicle for exploring knot theory that serves as a 3D tutorial education system to prototype our later methods in higher dimensions (usually 4D). Chapter 3 provides a general overview of this 3D educational system and various applications such as haptic methods of exploring 2D knot diagrams, modeling the physics of 3D knots, and interactive manipulation of knot diagrams; most of KnotExplore's features are extended to four dimensions later in this thesis.

Chapter 4 is concerned with haptic exploration of 2D manifolds embedded in 4D. Building on the 2D knot diagram exploration tool in Chapter 3, Chapter 4 presents a multimodal paradigm for exploring topological surfaces embedded in four dimensions. Just as 2D knot diagrams contain interrupted curves, 3D graphics projections of smooth 4D topological surfaces contain interrupted surfaces where the surface projections intersect with one another. We exploit the free motion of a computer based haptic probe to support a continuous motion that follows the *local continuity* of the object being explored. Chapter 4 also presents additional sensory cues to provide supplementary or redundant information. For example, we can use audio tags to mark the relative 4D depth of illusory 3D surface intersections produced by projection from 4D, as well as providing automated refinement of the tactile exploration path to eliminate jitter and snagging, resulting in a much cleaner exploratory motion than a bare uncorrected motion. Visual enhancements provide still further improvement to the feedback: by opening a view-direction-defined cutaway into the interior of the 3D surface projection, we allow the viewer to keep the haptic probe continuously in view as it traverses any touchable part of the object. Finally, we extend the static tactile exploration framework using a

dynamic mode that links each stylus motion to a change in orientation that creates at each instant a maximal-area screen projection of a neighborhood of the current point of interest. This minimizes 4D distortion and permits true metric sizes to be deduced locally at any point.

Chapter 5 examines the vector operations in four space as extensions of their three-space counterparts. Two main algorithms are given in this chapter for computing the distance between line segments embedded in four dimensions, and the distance between a point and a triangle embedded in four dimensions. These algorithms are the key to detecting collisions between rigid 4D objects. Based on our 4D collision detection and avoidance mechanism, we introduce a touch-based interface to control 4D surfaces through their 3D shadows while providing feedback simulating weight, deformation resistance, and momentum.

Chapter 6 treats 4D objects in a different way. It addresses one specific problem, that of modeling the correct physics of 4D objects by modeling a cloth-like class of 4-dimensional objects. Chapter 5 and Chapter 6 are complementary in some ways, Chapter 5 concerns collision between 4D rigid objects and Chapter 6 takes care of self-collision for deformable 4D objects. These two approaches combine to reveal the full richness of our four-dimensional physical modeling.

Chapter 7 discusses a mathematical process, the *homotopy*, that lends itself particularly well to real-time interactive computer manipulation. This leads us to consider interactive computer implementations of a closely related problem, that of everting a two-sphere embedded in 3D. To evert a sphere is to turn it inside out by means of a continuous deformation that allows the surface to pass through itself, but forbids puncturing, ripping, creasing, or pinching the surface. We study the

sphere eversion problem using physical modeling and interactive mouse-based methods that resemble our 4D physical surface techniques. By modeling the correct physics and bending energy of a circle and a sphere composed of cloth-like material, we are able to answer questions like "can circles embedded in 2D be everted?", and "can a two-sphere embedded in 3D sphere be everted?".

2

4D Visualization

Mathematics has long been connected with visualization. In the nineteenth century, almost every mathematics department had a set of plaster and screen models of topological objects. The famous work of popular mathematics, *Anschauliche Geometrie* (English *Geometry and the Imagination*) by Hilbert and Cohn-Vossen [Hilbert and Cohn-Vossen(1952)], contains photographs and illustrations of many such examples, and in fact its appendix contains the classic algebraic form of the projective plane embedded in 4D that is a favorite of mathematical visualizers. In this section we discuss the general historical background of mathematical visualization of interest to this thesis.

2.1 4D Visualization in the Traditional Media

If we consider diagrams acting as visual aids, then almost any mathematical paper might be considered a form of mathematical visualization. To find examples we might go back as far as Gauss, Euler, or even Euclid. However, generally these diagrams are not intrinsic to the proof of theorems; i.e., the diagrams could be deleted without any effect on the correctness of the results. It seems

more appropriate therefore to limit the scope of mathematical visualization to those areas where the interesting features of the study are *primarily visual*. In this stronger sense, mathematical visualization started in this century when mathematicians began to make special efforts to convey their ideas in a highly visual manner. As noted, one of the early works using this approach to popularize mathematics was David Hilbert and S. Cohn-Vossen's *Anschauliche Geometrie* (*Geometry and the Imagination*). This book is contains many hand-drawn illustrations and diagrams, as well as images of hand-built models executed long before the availability of computer graphics. These diagrams are useful to current mathematical illustrators as examples of effective depiction of mathematical objects.

The Hypergraphics symposium was held in 1978 and brought together artists, architects, mathematicians, and computer programmers in order to share insights on higher-dimensional geometry and its graphical representation. The proceedings, published as a book [Brisson(1978a)], contain a number of interesting reflections on the visualization of higher dimensional objects [Brisson(1978b), Noll(1978), Banchoff and Strauss(1978)] such as hyper-cubes and also an impossible 4D illusion constructed by Scott Kim [Kim(1978)].

Several books have been published in the past decade that are superb examples of mathematical visualization. Perhaps the best of these is *A Topological Picturebook*, George Francis' beautiful book [Francis(1987)] on many different aspects of topology. Most of the illustrations are hand-drawn diagrams of 3D objects, but he does include a chapter on higher-dimensional entities such as the projective plane. Francis is an expert at drawing *Seifert Surfaces* [Francis(1983)]. These are orientable surfaces that span a knot, i.e., they have the knot as a boundary (see

also [van Wijk and Cohen(2005), van Wijk and Cohen(2006)]). Equally beautiful, but intended for a more general audience, is Thomas Banchoff's recent book *Beyond the Third Dimension* [Banchoff(1990)], which popularizes some mathematical concepts and contains numerous high quality illustrations of 4D objects. Scott Carter's *How surfaces intersect in space* [Carter(1995)] is also a marvelous book of pictures illustrating the fundamental concepts of geometric topology in a way that is very friendly to the reader.

2.2 Computer 3D Printing and Concrete Mathematics

Tangible realizations of computer-generated shapes in science and mathematics can now be created using so-called Rapid-Prototyping (R-P) printers that can render a 3D CAD model in physical materials. Automated Fabrication technologies are appearing, among them: Stereolithograph (3D Systems), Selective Laser Sintering (DTM Corp.), Fused Deposition Modeling (Stratasys), Laminated Object Manufacturing (Helisys) and the several licensees of the MIT 3D Printing Consortium, Direct Shell Production Casting (Soligen) and Z-Corporation.

These various technologies all build three-dimensional objects via the common principle of dividing the object in software into a sequence of horizontal slices, from bottom to top, which the machine constructs in a physical material and binds together. Thus, these technologies can be categorized as "Layer-manufacturing," each of which uses some kind of three-dimensional computer object slicing software. The actual 3D object provides a mathematical model accessible to the general public; the tactile experience of such an object can enhance our understanding of the mathematical object [Dickson(2000)] (see, e.g., Figure 2.1).

(a) (b)

(c) (d)

(e) (f) (g) (h)

Figure 2.1: (a) The Hyperbolic Paraboloid as a physical object may contain no abstract information to someone unfamiliar with its mathematics. (b) The Klein bottle and two halves manufactured via Stereolithograph. (c) The Wente Torus manufactured in collaboration with the Rapid Prototyping and Manufacturing Institute, Georgia Institute of Technology, Andrew Layton. (d) 3D printing of the Smyth Constant Mean Curvature Surface with Three Legs by Andrew Inman, Digital Fabrication Lab, College of Architecture, UNC at Charlotte. (e) The Quaternion Julia Set manufactured in collaboration with the Center for Rapid Product Realization at Western Carolina University. (f) Bernard Morin with models of the Optiverse during an International Colloquium on Art and Mathematics in Maubeuge, Northern France on September 22, 2000. (g)-(h): Examples of physical models representing a highly self-intersecting surface, constructed by 3D projection from the four-dimensional mathematical description. (Model courtesy of Stewart Dickson.)

Computer 3D printing technology has also been used to construct objects that are 3D projections from four-dimensional space (see, e.g., Figure 2.1(g)-(h)). However, as we can see from the images, these models contain highly self-intersecting surfaces in 3D, despite the fact that they are smooth and continuous in the fourth dimension.

2.3 Computer Systems for 4D Visualization

This section gives an overview of software specifically designed to support computer aided 4D visualization systems.

An early attempt at producing a 4D visualization system was undertaken by Thomas Banchoff and his collaborators, who achieved interactive, real-time geometrical visualizations of surfaces projected from 4D to 3D with a custom-built multiplier and a fast-refresh vector graphics display for wire-frame modeling.

At the University of North Carolina/Chapel Hill, the Fourphront system by David Banks is another interactive system for the study of surfaces in 4D. This system, which ran on the high-speed massively parallel Pixel-Planes 5 graphics engine, provides user control of transparency, depth cuing, intersection highlights, and 2-sided paint as well as supporting user control of rotation and translation.

In 1987, the Geometry Supercomputer Project was created at the University of Minnesota, and subsequently evolved to become the Geometry Center, a National Science and Technology Research Center, in 1991, and was dismantled in 1998. The Center served as a focal point for a number

of efforts in mathematical visualization; a widely used system distributed by the Geometry Center is *Geomview*, a very general surface viewer developed by Stuart Levy, Tamara Munzner, and Mark Phillips [Phillips et al.(1993)Phillips, Levy, and Munzner]. Its built-in functionality can be extended by customized user programs, called *external modules*. While Geomview is fundamentally a 3D viewer, the 4DView external module by Daeron Meyer supplied with Geomview accepts 4D data points, allows the user to change the 4D viewpoint, and includes tools for creating 4D slices. Another external module, NDView by Olaf Holt and Stuart Levy, interacts with objects of dimension 4 and higher using multiple projections into families of 3D subspaces. A typical Geomview surface display is shown in Figure 2.2.

A 4D viewer with a different philosophy is the MeshView program designed by Hui Ma and Andrew Hanson at Indiana University [Hanson et al.()Hanson, Ishkov, and Ma]. This system supports the Geomview 4D surface mesh data format, but in addition provides a high-speed mouse-driven 4D rotation interface and a utility for locating particular points on a projected surface relative to the abstract parametric mesh coordinates. Figure 2.3 illustrates a MeshView display of the $n = 4$ case of a closed-form construction developed by Andrew Hanson for representing the complex "Fermat" equation $(z_1)^n + (z_2)^n = 1$; the surface is projected from 2 complex dimensions to 3 real dimensions from any desired viewpoint. Large families of complex surfaces can be displayed interactively in MeshView using this technique.

Finally, the KnotPlot program authored by Robert Scharein has been designed for visualizing and interacting with 3D and 4D knots. Figure 2.4 shows a typical display of higher dimensional knot using KnotPlot. However, these high-dimensional knots are treated as rigid objects in 4D; KnotPlot

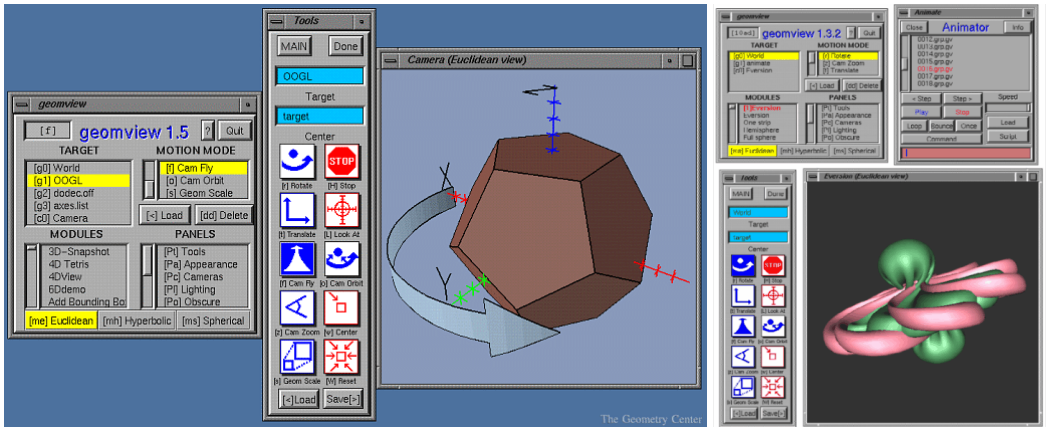
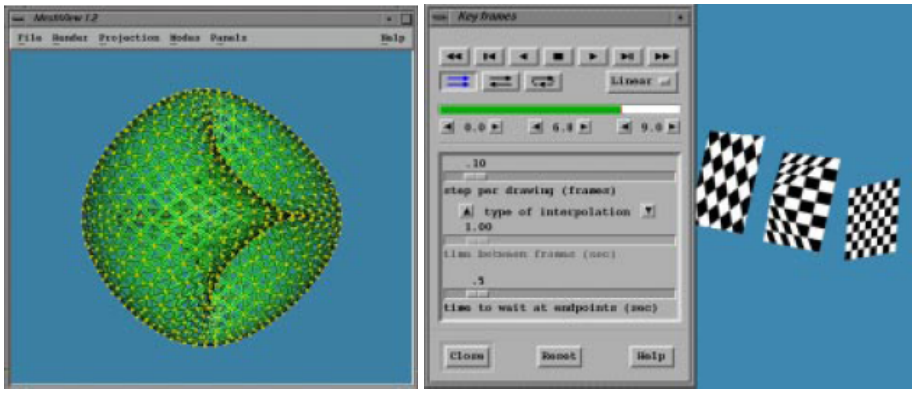


Figure 2.2: (a) Typical screen image of Geomview software. (b) A frame from the film "Outside In," a sphere halfway through the Thurston eversion. In this example of a typical Geomview application, the Animator external module is being used to control a "flipbook" of the animation. The main "Geomview" control panel at the upper left controls the viewing state, invokes modules, and brings up other control panels for control of lighting, object appearance, and so on. The basic mouse driven motion controls for changing the user's view are in the "Tools" panel on the left. (©Geometry Center.)



(a) (b)

Figure 2.3: (a) Meshview's interface window with a two-torus embedded in 4D drawn using edges, vertices, and negative screen door transparency. (b) Meshview's key-frame animation interface controlling a set of animated polygons; the texture coordinates are also key-framed. (©Indiana University.)

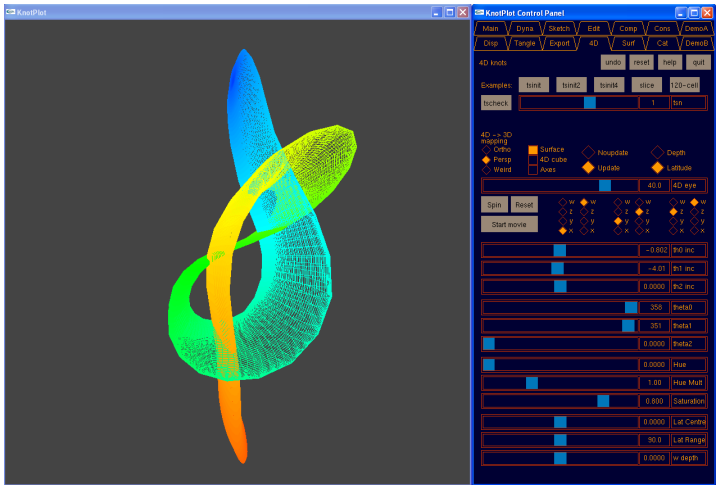


Figure 2.4: Display of a +2-twist spun trefoil knot using *KnotPlot*, part of Robert Scharein's PhD thesis work.

does not support 4D manipulation.

2.4 Film and Video

The emergence of computer graphics has also sparked a renewed interest in visual mathematics in the form of films and videotapes. Many of these efforts have made use of interactive graphics.

One of the early efforts in this that received wide attention was Thomas Banchoff's interactive, real-time geometrical visualization studio at Brown University. In the late 1970's and early 1980's, he and his associates produced computer animated films of 4-dimensional objects such as the award winning "Hypercube," "The Veronese Surface," and wireframe versions of his recent video animation, "The Hypersphere: Foliations and Projections." Several of these animations were shown at early Siggraph Conferences [Banchoff(1986), Banchoff(1990), Banchoff and Strauss(1978)].

At Indiana University, a scientific visualization effort led by Andrew J. Hanson has focused on

(a) (b)

(c) (d)

Figure 2.5: (a) *Knot*^{4}. (b) Visualizing Fermat's Last Theorem. (c) FourSight. (d) 4Dice. (©Indiana University)

finding new ways to represent and visualize Riemann surfaces, on rendering techniques using 4D light, and on the development of corresponding interactive methods; this group has produced a variety of short animations, including three shown at Siggraph, "Visualizing Fermat's Last Theorem," "FourSight," "4Dice" and "*knot*^{4}," and three others at IEEE Visualization conferences.

3

KnotExplore: Introducing Dimensions

3.1 Dimensions

3.1.1 Fechner, Plato, and Shadow Images

An early proponent of dimensional analogy was the 19th century psychologist and physiologist Gustave Fechner, who wrote a short story, *Space has Four Dimensions*, as part of his collection *Vier Paradoxe* published in 1846 under the pseudonym of Dr. Mises. Fechner described a two-dimensional creature, a "shadow man," projected to a flat screen. This shadow being could interact with other shadows, but could not conceive of a direction perpendicular to the screen. Fechner suggests that for such a being, time would be a third dimension, expressing the movement of his whole screen in a direction that he cannot comprehend spatially.

The idea of treating shadow figures goes back much further, at least to Plato's *Allegory of the Cave* in the seventh book of *The Republic*. There Plato introduces the allegory of the Cave, whose residents can only perceive the outside world via shadows thrown upon the walls, and who thus have only

limited knowledge of the objects in the world.

Imagine prisoners, who have been chained since their childhood deep inside a cave (see Figure 3.1): not only are their limbs immobilized by the chains, but their heads are chained in one direction as well so that their gaze is fixed on the wall. Behind the prisoners is an enormous fire, and between the fire and the prisoner is a raised walkway, along with puppets of various animals, plants, and other things are moved along. The puppets cast shadows on the wall, and the prisoners watch these shadows. When one of the puppet-carriers speaks, an echo against the wall causes the prisoners to believe that the words come from the shadows. The prisoners engage in what appears to us to be merely a shadow play. This, however, is the only reality that they know, even though they are seeing merely shadows of objects. They are thus conditioned to judge one another by their skill in quickly naming the shapes and interactions between shapes.

Plato does not suggest that the shadows have the capability of interacting with one another, and this is the essential extension put forth by Fechner's work.

3.1.2 Abbott, Illusions and Miracles

The idea of cross-dimensional understanding has developed in many directions since then. Just over one hundred years ago, a busy headmaster produced a slender book that was at the same time a satire on the limited social perspective in Victorian England and an introduction to the geometry of higher dimensions. This book, *Flatland*, has undergone several periods of popularity, including our present day, when five new editions have appeared in as many years and when several authors have written new books with *Flatland* as their inspiration. Although Edwin Abbott Abbott was not the first

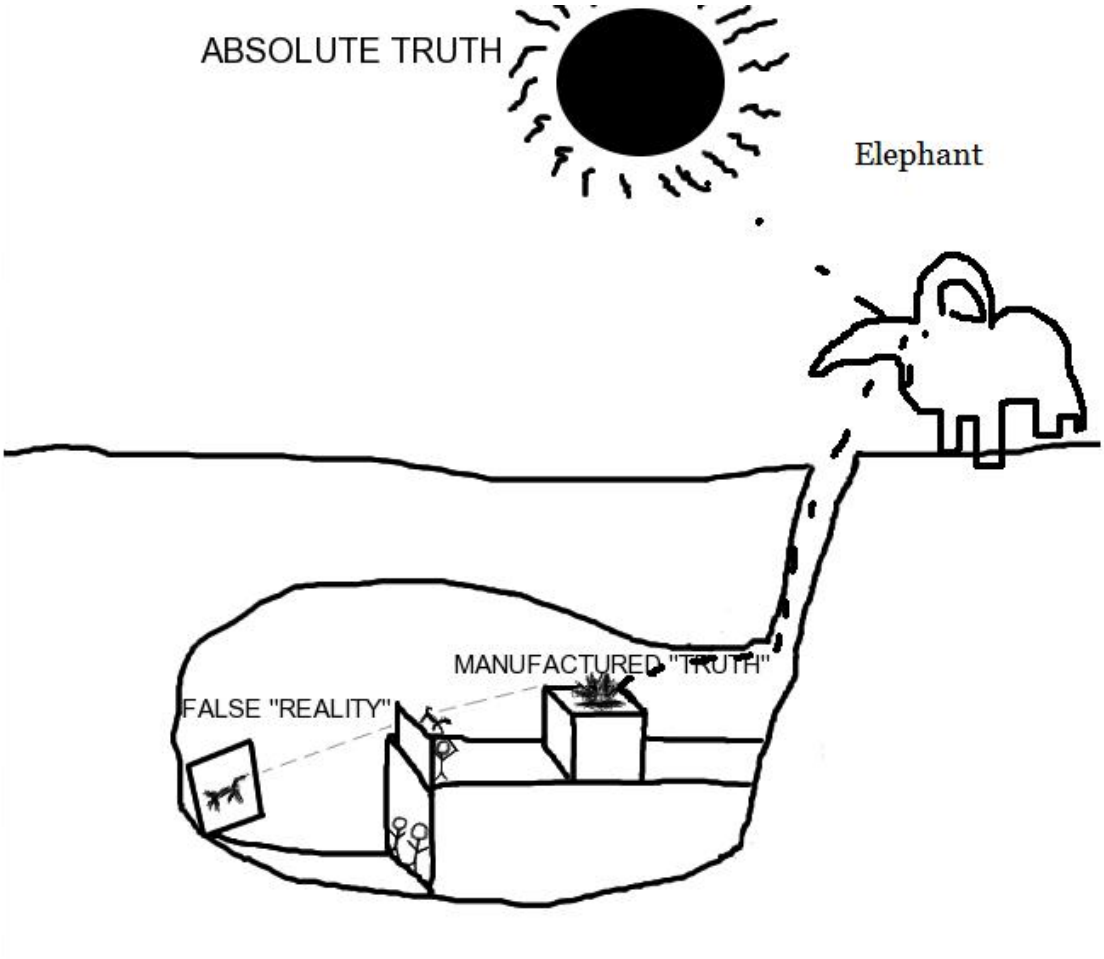


Figure 3.1: A schematic representation of Plato's allegory of the Cave. (This drawing is highly simplified and should only be used as an aid for grasping the picture the allegory creates; it does not represent the entire allegory.)

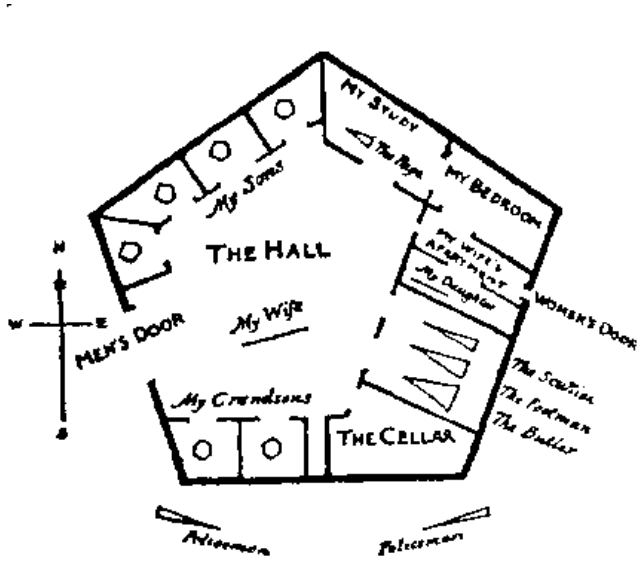


Figure 3.2: Abbott's own drawing of the house of "A Square," which summarizes the social structure of *Flatland*. "A Square's" wife and daughter are drawn as single lines, and, in ascending social order, the male servants, butler, footman and page, are triangles. The owner is A Square, and as each future generation adds a further angle, the Square's sons are the four pentagons and his two grandsons the two hexagons. The entrance doors to the house are of an appropriate width for the two sexes.

person to posit a two-dimensional universe inhabited by flat beings, he was the first to explore what it would mean for such individuals to interact with phenomena from a dimension higher than their own [Abbott(1952), Dewdney(1984)]. Today the development of high-speed computer graphics puts us face to face with higher-dimensional phenomena, and in our investigations we are all too often just as ill-equipped to understand them as was "A Square," the two-dimensional protagonist of *Flatland* (see Figure 3.2), more than one hundred years ago.

3.1.3 Dimensional Progressions

From Point to Hypercube. Abbott used dimensional analogies to great effect in raising questions about the way we see the world, especially when we come into contact with the truly transcendental. For over a century, mathematicians and others have speculated about the nature of higher dimensions, and in our day the concept of dimensions has begun to play a larger and larger role in our conception of a whole range of activities.

Analogy is still the dominant idea in the history of the concept of dimensions. If we truly understand a theorem in plane geometry, then we should be able to find one or more analogies in solid geometry, and conversely, solid geometry theorems will often suggest new relationships among plane figures. Theorems about squares should correspond to theorems about cubes or square prisms. Theorems about circles should be analogous to theorems about spheres or cylinders or cones. But if we learn a good deal by going from two dimensions to three, would we not learn even more by going from three dimensions to four?

Mathematicians began to follow different paths within this progression, developing sequences of analogous figures starting even further back along the dimensional ladder. One possible sequence started with a point, having zero dimensions, no degrees of freedom. A point moving in a straight line generates a segment with two endpoints, a fundamental one-dimensional object. A segment moving perpendicular to itself in a plane generates a figure with four corners, a square, the basic object in the second dimension. Proceeding to the third dimension, we move a square perpendicular to itself to form a cube, a basic three-dimensional object. Even though A Square could no longer

appreciate this process fully, he could follow along at a theoretical level and deduce certain properties of this cube that he could not see, e.g., that it has eight corners. Next we ask what appears if we move a cube in a fourth direction perpendicular to all its edges. We would get a basic four-dimensional object, a hypercube, and although we can no longer fully appreciate the process, we can predict that such a hypercube will have 16 corners. The number of corner points generates a geometric progression, and we can easily arrive at a formula for the number of corners of a cube in any dimension (see Figure 3.3).

Considering boundaries leads to another progression. A segment has two boundary points. A square is bounded by four segments. The boundary of a cube is six squares. Following this progression, we expect that the hypercube will be bounded by cubes, and that there will be eight of them. The formula for the number boundary pieces in any dimension can be found by using an arithmetic progression.

Physics in Higher Dimensions. Although mathematicians can predict the numbers of corners and boundary pieces of analogs of the cube in any dimension using dimensional progression, these numbers still leave something to be desired. In plane and solid geometry, the objects were not only real, they could also be represented as diagrams or models, capable of revealing significant relationships. How can we see what a hypercube would look like? If we cannot see it, how do we know that our assertions about it are true?

Geometers in the last century devised methods for visualizing objects in higher dimensions, and these methods are worth looking at. Quite sophisticated in some ways but with limitations that were

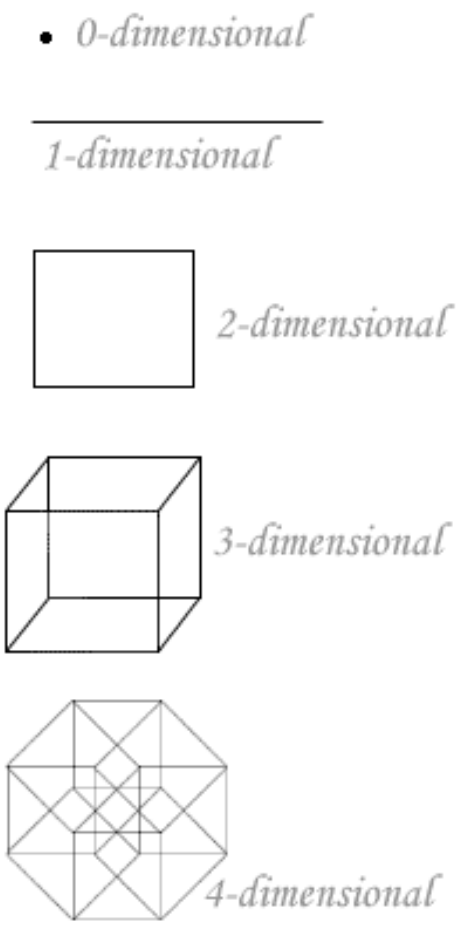


Figure 3.3: The analog of a cube in any dimension can be generated by moving the preceding lower-dimensional cube perpendicular to itself.

often frustrating, the imaging and modeling techniques of a hundred years ago were inadequate for interpreting complicated objects in four dimensions. Eventually higher-dimensional geometry came to be based not only on analogy but on coordinate geometry, which could translate geometric concepts into numerical and algebraic form. Although such formal methods put the mathematics on a firm footing, they did little to satisfy the desire to "see" the higher dimensions.

Modern Visualization Technology. For the task of visualizing objects in the fourth dimension, the ideal instrument is the interactive computer system. Interactive computer systems in fact work almost exclusively with shadows, i.e., representations of our 3D world cast upon 2D graphics screens by mathematical projection and rendering algorithms. Graphics methods allow us to add features to these shadows such as lighting, shading, and occlusion that we interpret via our learned perceptual models as being truly 3D, despite the fact that in truth their dimension is reduced.

Our task in this chapter is to show how one can fully exploit projections to lower dimensions and use physically reactive projection-based haptic controllers to transform the task of interacting with the fourth dimension to a new level of "shadow-driven" physical reality. We like to think of this method intuitively as existing in a "shadow world," a term widely used in the classic literature, with clear interactive implications and an ancient context adopted by authors such as Fechner. Although it is possible to confuse what we call "shadow space" with the conventional illumination-based shadows of computer graphics technology, this will typically not be an issue in our treatment. We start from the fairly familiar idea of a knot "crossing diagram" drawing executed with a pen and paper, and extend that idea for pedagogical purposes to a haptic interface that is restricted to a plane, but still empowered to sketch, touch, and take control of a 3D mathematical knot through its projection to

the controller plane. Having established the mechanisms and intuition of this artifice, we proceed to our full-featured extension, a 3D haptic system capable of manipulating 4D cloth-like surfaces with realistic physical characteristics, forces, and reactive collisions. We believe in this way, we can proceed to attack families of significant problems in 4D intuitive visualization such as untying the apparently knotted twist-spun trefoil, canceling pinch-points in topological constructions, and modeling 4D "chain" structures consisting of linked spheres and ribbons.

3.2 Overview of Shadow Manipulation Scenario

3.2.1 2D Example

People have long been fascinated by the positions of shadows, as we can easily see by examining the placement of temples in many ancient cultures. Nearly all religious structures were arranged to take into account the positions of the sun and of shadows at certain key times of the year. Many of these structures actually functioned as rudimentary observatories, precisely identifying crucial days like the summer or winter solstice by the positions of shadows cast by their monuments. Artisans and astronomers in many cultures devised sundials for precise measurement of time, effectively transforming the passage of time into the movement of a shadow across a plane (see Figure 3.4).

For the most part, shadows are images cast on a plane surface, like a wall or the ground, by some object located between the surface and a light source. In this chapter, we first introduce the reader to our approaches to interacting with higher dimensions using the 2D knot diagram (see, e.g., Figure3.5), a simple example of 2D shadows that exposes the richness of our approach.

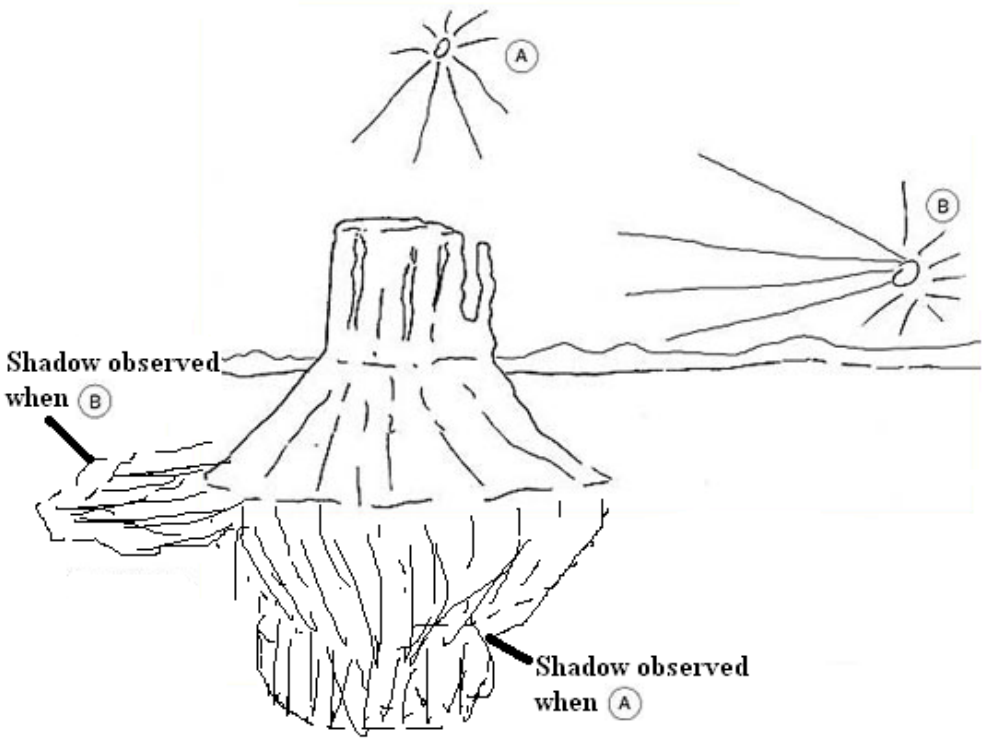


Figure 3.4: Transforming the passage of time into the movement of a shadow.

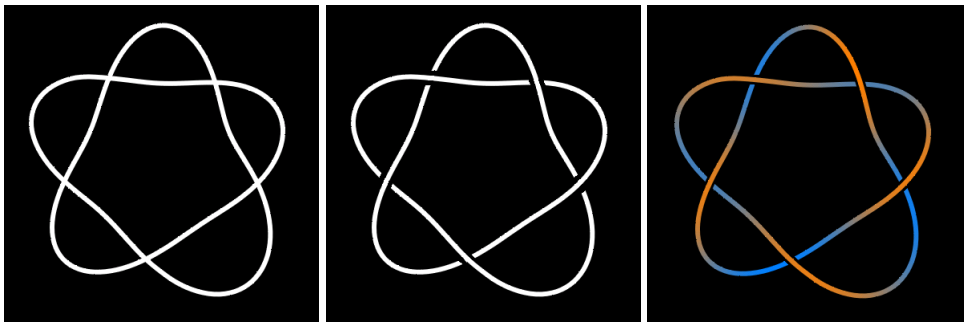
Figure 3.5: The 3D depiction of a trefoil knot and its 2D knot diagram representation.

3.2.2 Motivation

People learn about the everyday world by combining sensory modalities. Knowledge of shape comes from a synthesis of the sensations of sight and touch acquired during exploration. By combining computer graphics with computer haptics, which endow the environment with intuitive force feedback, we can begin to create a sensory bridge from the shadow world to the higher-dimensional world we are trying to comprehend. While our ultimate challenge is to understand the physical 4D world using 3D haptics in its shadow space, it is very useful, perhaps even cognitively necessary, to introduce the basic features of our approach using a simplified analogy based on 2D shadows of 3D curves.

We therefore begin by examining the 2D projection of a 3D curve, noting specifically the overlapped neighborhoods that contain the 2D crossing points of pairs of 3D curve segments. If all we can see is the equivalent of pen strokes of a drawing on paper or the actual physical shadow of the 3D curve, we find the result in Figure 3.6(a), which is devoid of 3D information. This problem is typically overcome by employing the "crossing diagram" method illustrated in Figure 3.6(b); this corresponds essentially to a depth-buffered rendering with some embellishments to emphasize discontinuities in depth. By thickening the curve and providing geometry and material features combined with lighting and shading methods, we can get the additional improvement shown in Figure 3.6(c).

Now we can begin to see how to exploit a haptic probe in shadow space to facilitate full geometry manipulation: when the probe is constrained to the 2D shadow plane, the user can still freely edit the 3D structure in the *shadow-plane directions*. Permitting the user to modify the 3D depth of any point along a projection ray by using alternate states or modifiers completes the ability to support



(a) (b) (c)

Figure 3.6: (a) The 2D shadow diagram of this mathematical link has no 3D cues. (b) The 2-$\frac{1}{2}$ D knot diagram provides sufficient 3D depth information to characterize the 3D geometry. (c) Rendering with light and material adds apparent 3D geometry, depth, and shape to the 2D image.

full-space geometric editing. We can thus, for example, use controls in the 2D shadow plane to support 3D interactions that can create an arbitrarily complex topological knot. The key ideas of the overall scenario, exemplified by the 2D shadow world, can now be summarized as follows:

- *Create a shadow space in one lower dimension.* For example, take a 3D curve, create a 2D projection, and constrain the haptic probe's action to that 2D plane.

- *Add an extra $\frac{1}{2}$ dimension.* The shadow space display can be enhanced by showing occlusions of objects farther from the projection point when crossings occur. Similarly, sketching in the shadow space can query the user for an over-or-under choice when the curve-drawing shadow-space cursor collides with a part of the curve at the same depth in the real space.

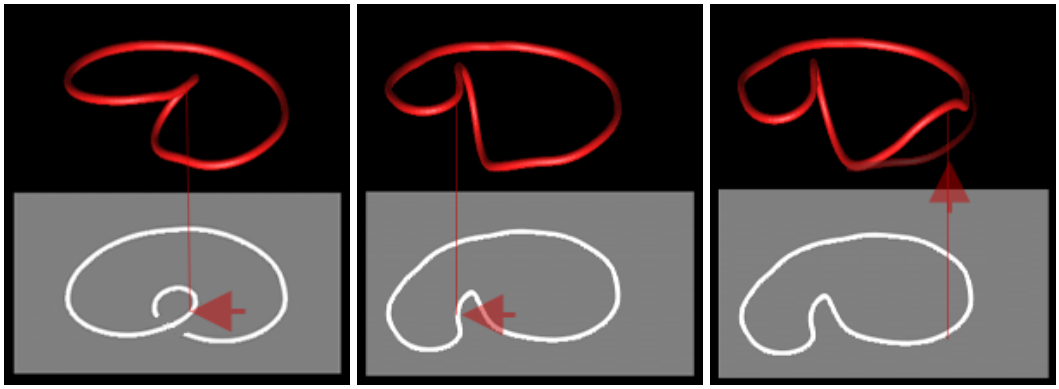
- *Enhance the geometry and depth information in the higher dimension.* Computer graphics and visualization methods can enhance the perception of the geometry by color depth coding and exaggerated occlusion or crossing diagrams; rough sketches similarly benefit from global smoothing.

- *Make it physically touchable.* By modeling the physical interactions, collisions, forces, stretching, and momentum accurately in the full dimension, and projecting these touchable interactions down to the shadow space, we can support a substantial sensation of physical reality.

- *Manipulate projected images of objects embedded in the higher dimension.* We can combine direct shadow space controls with projection-ray controls to drag or deform object segments to arbitrary new places in the full-dimensional scene.

- *Explore the modeled objects with the haptic probe.* Pre-defined object models or newly constructed models can be explored by constraining the haptic probe to the model domain. This avoids many problems of the real world, since the probe itself is not physical and is not bothered by illusory self-intersections of a continuous object in the shadow domain. Furthermore, the probe itself does not collide with parts of the object, which is a major problem with touch-based exploration of objects such as real, physical knots [Hanson and Zhang(2005)].

In summary, by mapping the lower-dimensional user space (projection or shadow) to the full-dimensional object along either the dimensions of the projected space *or* along the projection rays, we can manipulate object deformations in the full space while experiencing physical artifacts such as force, inertia, momentum, and collisions induced by the higher dimensional simulation. In Figure 3.7, we show a very simple example with an "outside view" of a 3D curve deformation corresponding to standard knot move, as well as a ray-aligned move, controlled from the shadow space. Figure 3.8 illustrates the appearance of the entire user interface, including the virtual ray from the projected curve on the 2D screen to the user's 3D mental model of the artifact being manipulated.



(a) (b) (c)

Figure 3.7: (a) (Above) 3D object projected to 2D shadow (below). (b) Grabbing a point in the shadow and moving it laterally in the 3D space. (c) Grabbing a point and moving it along the shadow ray.

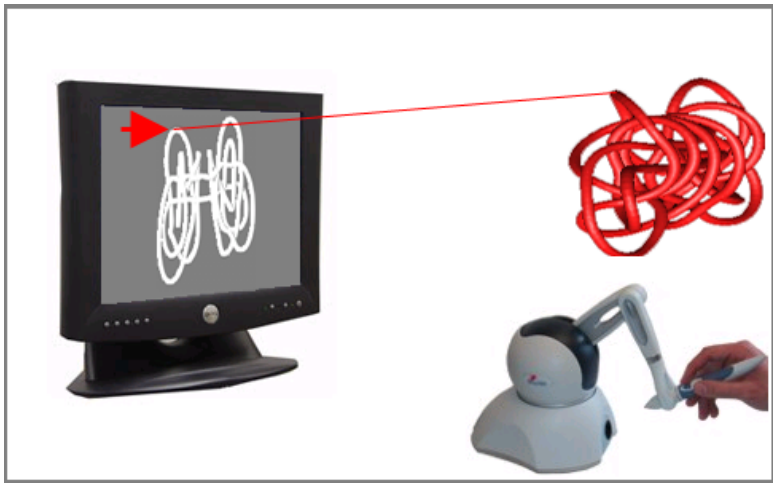


Figure 3.8: Screen image, haptic probe, virtual ray from the physical space to the shadow, and the user's corresponding mental model during shadow-driven editing.

3.3 Details of Implementation Models

In this section, we describe the families of models used to implement the interaction procedures and user interfaces. We again focus on 2D projections of 3D curves to give explicit context, with the assumption that typical features will be straightforward to extend to 3D projections of 4D curves and surfaces; aspects requiring specific variants for the 3D/4D situation will be treated explicitly in a later section.

Our fundamental techniques are based on a wide variety of prior art, including haptic interfaces focusing on virtual realism (see, e.g., [Baxter et al.(2001)Baxter, Scheib, and Lin]), the exploration of unknown objects by robotic fingers (see, e.g., [Okamura(2000), Okamura and Cutkosky(2001)]), and other variants on haptic exploration techniques ([L.Kim et al.(2004)L.Kim, Sukhatme, and M.Desbrun] and [Yu et al.(2000)Yu, Ramloll, and Brewster]). Relevant methods of force feedback and user assistance include, e.g., the work of [Zahariev and MacKenzie(2003)], [Kennedy(2002)], [Forsyth(2004)], [Park and Niemeyer(2004)], and [C.W.Reynolds(1999)]).

However, we have found many aspects of shadow-space manipulation to be unique, and thus we have been required to adopt customized hybrid approaches. For example, forces must be computed in the projection domain, but must maintain an accurate simulation of the higher dimensional physics to account for and create a haptic response to phenomena such as collisions and attendant over/under choices. Conversely, *apparent* collisions in the shadow of components that are physically separated in the physical dimensions must be ignored.

The basic modeling methods, components, and features characterizing our interface are summarized in the sections below.

3.3.1 Shadow Space Force Modeling

Our basic force model simulates a "sticky" stylus [Hanson and Zhang(2005)] in the shadow space using a damped spring configuration model; the probe can move freely in the shadow surface, but cannot move along a ray (which is technically possible in the 2D/3D situation, but impossible in 3D/4D).

The damped spring force model calculates the point $\mathbf{C}'$ as the projection of the haptic device proxy $\mathbf{C}$ on the shadow plane [Sen(2004)]. The difference $\mathbf{N} = \mathbf{C}' - \mathbf{C}$ is used to compute a generalized Hooke's law force

$$\mathbf{F}_m = H|\mathbf{N}|^{1+\beta}\hat{\mathbf{N}} \,. \tag{3.1}$$

Here H is a constant, and $\beta = 0$ for an ideal (linear) spring. This mechanical restoring force is applied whenever the stylus is displaced from the surface, but we allow force-free motion along the tangent direction to the surface to facilitate exploration of the surface structure. The user must apply substantial effort to overcome this force, so the stylus feels stuck to the surface. The damping force is taken to be

$$\mathbf{F}_d = -K_d\mathbf{V} \,, \tag{3.2}$$

where $\mathbf{V}$ is the radial velocity used to smooth the force feedback.

3.3.2 Collision Avoidance and Repulsive Forces

Sketching Shadow Images. When sketching 2D shadows of 3D curves, collision detection is applied to detect whether the proxy apparently collides with other parts of the shadow image, and a repulsive force is rendered to prevent the haptic proxy from passing through segments in the shadow figure that actually collide in the full dimension; collisions are handled by making explicit over/under-crossing decisions.

Collision handling methods (see, e.g., [Larsson and Akenine-Möller(2001)] and [Gottschalk et al.(1996)Gottschalk, Lin, and Manocha], to mention only a few) detect a collision between virtual objects when they have just begun to penetrate each other. However, in a haptic interface, the colliding pair positions have physical manifestations, so one cannot simply shift both positions to undo the collision. Therefore we use a dynamic repulsive force to avoid collisions. The force model that we use to physically detect an impending collision and prepare for an over/under-crossing choice is

$$\mathbf{F}_r = -HS^{-1-\beta}\hat{\mathbf{V}} \,. \tag{3.3}$$

Here S represents the distance between the probe itself and the impending collision with the shadow figure, and $\hat{\mathbf{V}}$ is the direction of the radial velocity. This force slows down the haptic proxy's velocity as it approaches an existing shadow image segment, thus allowing the system to detect and manage collisions in a physically realistic manner.

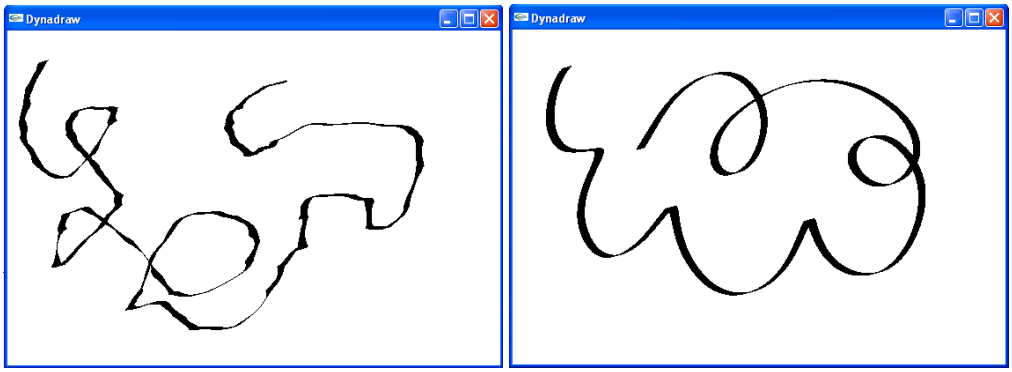
When a collision occurs between a piece of an edited object and an existing object in shadow space, users must make explicit over and under choices, e.g., by using modifier keys.

The resulting displacement may be smoothed using the minimum distance energy method [Huang et al.(1996)Huang, Grzeszczuk, and Kauffman]. We thus have a "$2\frac{1}{2}$D" collision avoidance that effectively leverages skills developed from work with pen and paper, and exploits intuitive force feedback to aid the drawing process. This is closely related to the (non-haptic) shadow-driven methods for sketching and manipulating 3D curves advocated by Scharein [Scharein(1998)] and by Cohen et al. [Cohen et al.(1999)Cohen, L.Markosian, Zeleznik, Hughes, and R.Barzel].

Smoothing the Sketching Process. The $2\frac{1}{2}$D interface in principle is sufficient to allow the user to sketch a knot diagram on the given 2D shadow surface. However, in practice, this free-hand 2D constrained drawing introduces significant jitter (human and mechanical). To improve on this, we follow Haeberli's *Dynadraw* method [Haeberli(1989)], connecting a virtual mass to the cursor position via a damped spring. As the user moves the cursor, the literal path is modified to create smooth, calligraphic strokes. From *Dynasculpt* (Snibbe [Snibbe et al.(1998)Snibbe, Anderson, and Verplank]), a haptic variant of *Dynadraw*, we adopt the method of attaching a virtual mass-spring system to the haptic probe position to smooth the free-hand results.

Our $2\frac{1}{2}$D drawing interface is unique in that the force feedback is rendered to *physically* constrain the haptic device to drawing on the virtual plane, while collisions in the higher-dimensional world are *physically* sensed through the haptic proxy; thus the knot must be created along the shadow-space path taken by the haptic device, constrained by the virtual physical object in space. We therefore have exploited a customized hybrid approach for smoothing the sketching process.

The implementation is calculated using Hooke's law with a damping term, adapting Equations (3.1)



(a) (b)

Figure 3.9: Left: A typical jittery result from free hand drawing. Right: Smoothed drawing resulting from adapting the *Dynadraw* method, which connects a virtual mass to the cursor position via a damped spring.

and (3.2) to give

$$\mathbf{f} = -H(\mathbf{P}_m - \mathbf{P}_f) - K_d\mathbf{V} \ , \tag{3.4}$$

where H is the spring constant, K_d is the damping constant, $\mathbf{P}_m$ is the position of the virtual mass, and $\mathbf{P}_f$ is the real-world finger-tip position as measured by the haptic system. The position of the virtual mass is updated using Newton's laws, $\mathbf{f} = m\mathbf{a}$, where m is the chosen mass, and we solve the second order differential equation using Euler's method.

This force model is illustrated in Figure 3.10. The finger tip is in effect constrained to the 2D Flatland and creates smooth pen strokes. This is very much compatible with the nature of the interaction with shadows in the experience of the 2D shadow man; when a collision occurs between the haptic proxy and a piece of edited knot diagram segment in the shadow space, the shadow man encounters a physical collision, and must make explicit over and under choices before passing through the apparent interruptions.

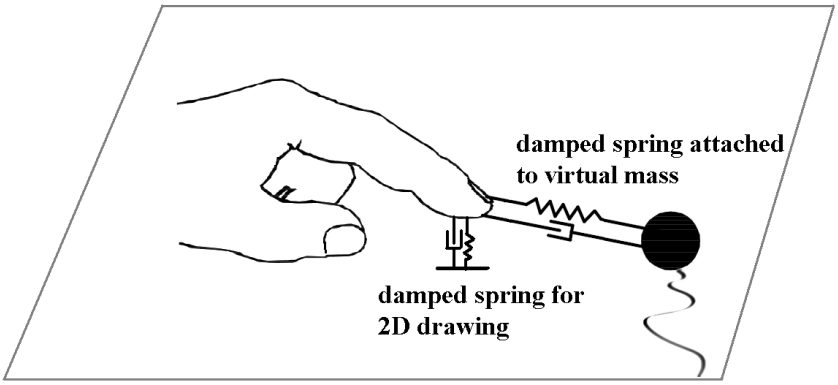
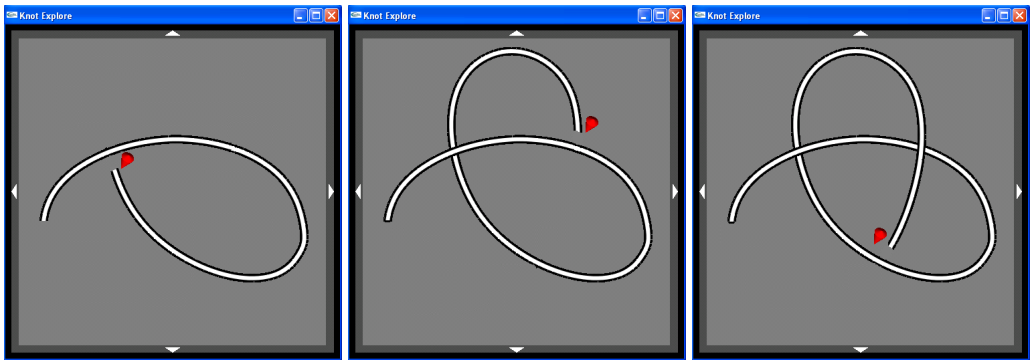


Figure 3.10: The force model for 2D shadow-driven sketching adapted from Dynasculpt's dynamic model.

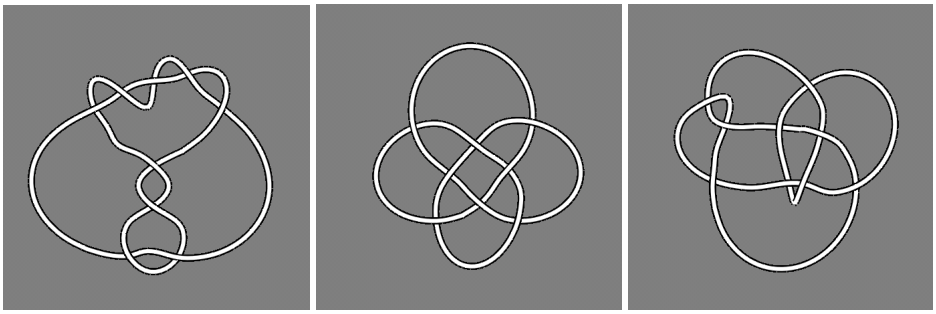


(a) (b) (c)

Figure 3.11: Haptic knot creation via a sequences of under, over, under . . .

Example: Creating a knot. In Figure 3.11, we illustrate typical steps for the shadow-space creation of a trefoil knot. Sample distances are interactively adapted, e.g., via bounding sphere checking, to make the final curve segments close to the same size and well-behaved (see [Brown et al.(2004)Brown, Latombe, and Montgomery]).

Figure 3.12 shows examples of more complex results.



(a) (b) (c)

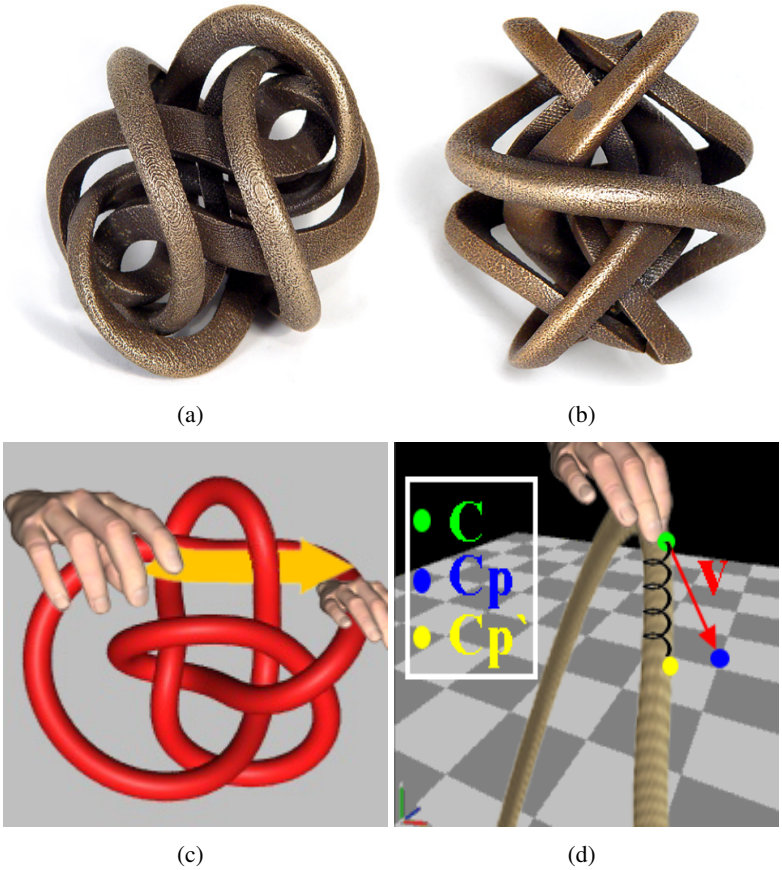
Figure 3.12: More sketched knot samples: $knot_9^6$, $knot_8^{18}$, $knot_9^{33}$

3.3.3 Multimodal Navigation

3.3.3.1 Haptic Feedback

The overall experience can be improved by constraining the probe to follow the local continuity of the object being explored. Tracing a real physical knot with one's finger results in collisions of the rope with the fingertip, forbidding smooth navigation; the projected image of a knot can contain massive interruptions of visual continuity as well, as shown in Figure 3.13(a-b). The computer-based haptic interface, however, can do something real-life cannot do, which is to support a continuous motion that follows the continuity of the object being explored without encountering physical obstructions while overriding visual obstructions. The haptic navigation method resolves the apparent conflict between the continuous structure of the actual 3D knot and the visual discontinuities at occlusion boundaries in the 2D shadow domain [Hanson and Zhang(2005)], as shown in Figure 3.13(c).

Haptic navigation can be assisted by force suggestions that constrain the allowed motion, while assisting and guiding the user's fingertip (the probe) towards the predicted position. The following



(a) (b)

(c) (d)

Figure 3.13: (a-b) Tracing a real knotted rope is difficult, even when you are holding the physical object in your hands (`www.bathsheba.com/sculpt/clef/`). (c) A computer-based haptic probe supports a continuous unobstructed exploration of the *local continuity*. (d) Damped-spring forces can improve the exploration experience.

steps describe our haptic servo loop model for adding force suggestions, as shown in Figure 3.13(d):

1. Get current haptic device coordinate $\mathbf{C}$, velocity $\mathbf{V}$, and the instantaneous update rate of the device R.

2. Compute the predicted haptic device coordinate

   $\mathbf{C_p} = \mathbf{C} + \mathbf{V} \cdot \frac{1}{R}$. ($\frac{1}{R}$ is the time step.)

3. Compute $\mathbf{C_p}'$ as projection of $\mathbf{C_p}$ on curve image $\mathbf{I}$.

4. Apply a damped spring force between $\mathbf{C}$ and $\mathbf{C_p}'$.

3.3.3.2 Auditory Feedback

Adding multimodal feedback can provide useful supplementary conceptual information. For example, if one explored a knotted curve with a probe constrained to the curve, but with no visual feedback, all knots would be similar — just a path coming back to itself. In practice, knot structure is encoded by the location and character of the crossings (over and under crossings in the projection). We can use this information to either replace or redundantly supplement the visual display by adding an auditory signal that distinguishes each over and under crossing in the projection, as illustrated in Figure 3.15. Similar over and under crossings occur for surfaces projected from 4D to 3D, and these can be similarly signaled [Hanson and Zhang(2005)].

If one were to travel on the knot itself, all one could tell would be that it is one continuous closed loop, so a knot is not interesting unless one views it from the outside. The continuous haptic exploration of 3D knots alone does not help to build such an *o*utside-view mental image, incorporating

(a) (b) (c)

Figure 3.14: (a) With visual feedback, the user is acutely aware of sliding through visual interruptions in the 2D knot crossing-diagram. (b) Without using visuals, however, one would be totally unaware of encountering a knot crossing. (c) Sound cues supplement or replace visual cues to assist in building a clear mental model of the mathematical knot.

locations of crossings and interactions. The visual feedback is undoubtedly important for understanding complex spatial relationships and structures. We are thus led to exploit sound, using word labels such as “over” and “under” to perceptually mark those points at which the continuous topological motion of the haptic proxy passes a visual disruption in the graphics image; this in principle will give users additional richness in the visualization (see Figure3.14).

The Solution. Using these multiple sensory modalities to present information can overcome contradictions in the visual feedback, or even make the visual feedback superfluous. When performing a haptic exploration revealing the local continuity of the physical dimension, users can feel the shape of the object via the multimodal interface as described above. They can also trace out the shape of a path with their computer-idealized finger, supplemented by a change in pitch (or volume) indicating the 3D depth, and specific auditory tokens or spoken narration triggered at special locations such as visual obstruction transitions (see Figure 3.15).

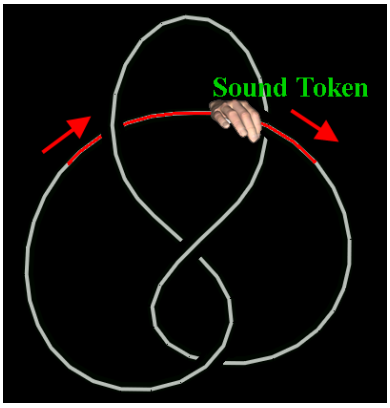
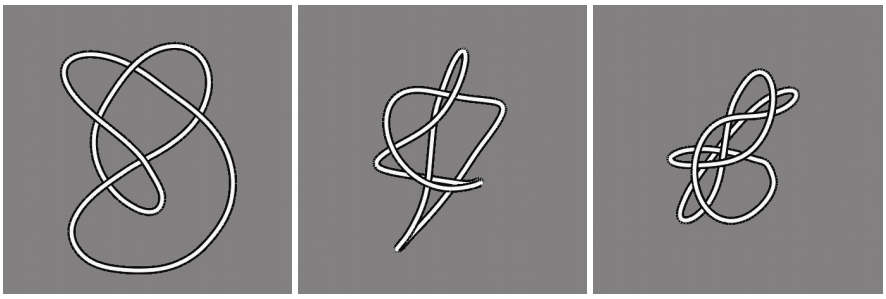


Figure 3.15: Sound cues can be added to tell the user when crossings occur during exploration.

3.3.4 Selecting Viewable and Touchable Knot Images

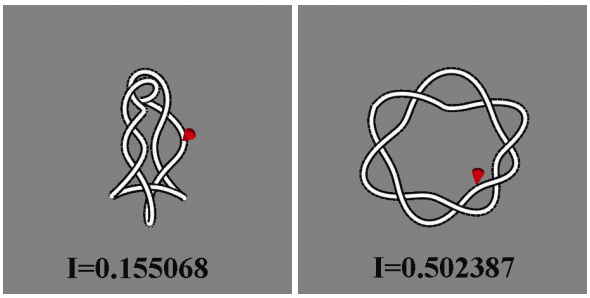
Different choices of projection can result in very different images; 2D knot diagrams are not invariant under changes of projection. Thus, for some tasks, we may wish to vary the chosen projection to optimize the view. One way is to optimize some aspect such as the projection size of the segment currently being touched by the probe; this has the advantage of making both the image size and the haptic-sensitive path correspond to a maximal local metric curve length relative to the point being probed [Hanson and Ma(1995b)]. Figure 3.16 illustrates the perceptual variations possible with a single topological knot, where Figure 3.16(a) has the fewest interruptions as well as the maximal projected area.

We approach the problem by adapting methods used in knot theory, as well as in multidimensional data visualization and viewpoint selection. For example, Kamada and Kawai [Kamada and Kawai(1988)] consider a viewing direction to be good if it minimizes the number of degenerate faces under orthographic projection, Hlavác et al.



(a) (b) (c)

Figure 3.16: The rigid 4_1 knot can be represented with different projections from 3D to 2D, some of which are very difficult to understand if too many lines cross in the image.



(a) (b)

Figure 3.17: Computing the knot viewpoint quality measure (**I**) of a 3D knot projected to a 2D plane.

[Hlavác et al.(1996)Hlavác, Leonardis, and Werner] optimize the exploration of a set of object images, and Vázquez et al. [Vázquez et al.(2001)Vázquez, Feixas, Sbert, and Heidrich, Vázquez et al.(2002)Vázquez, Feixas, Sbert, and Llobet] use an information theoretic measure for viewpoint entropy. Starting from this background, we synthesize a model that is closely related to the requirements of the haptic exploration task for knot diagrams. Our optimization measure, composed of the projected curve length and segment visibility in the knot images, is given by

$$I(K,c) = \sum_{i=0}^{N_s} \left(\frac{L_i}{L_t} \log \frac{L_i}{L_t} + V(i) \right) . \tag{3.5}$$

Here L_i represents the projected length of curve segment i and L_t is the total length of the knot curve embedded in 3D; $V(i)$ is the *visibility test function* for curve segment i, where $V(i) = -1$ if the segment is crossed by another segment, and otherwise $V(i) = +1$. Figure 3.17 shows two typical projected images of the 7_1 knot and their optimization measures. The measure is maximal, hence better, for Figure 3.17(b).

3.3.5 Representation and Display

We next give the details of how exactly knots are rendered and displayed. Our system has two major display modes. Any knot may be displayed in either a "2D knot crossing-diagram" mode or in a "3D smooth tube" mode (see Figure 3.6(b–c)). Since the "2D knot crossing-diagram" is common practice in textbooks on knot theory, our interface uses it as the default representation for knots. However, many users also like the "smooth look," with the understanding that the apparently smooth curves and surfaces realistically represent the 3D knot structure. In this section, we describe

the methods used to rendering a mathematical knot in both the 2D and 3D representations.

3.3.5.1 Rendering 2D Knot Crossing-Diagrams

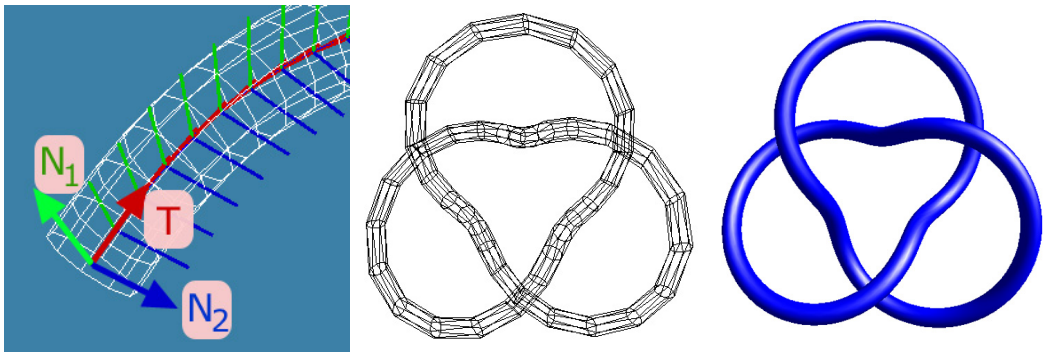
The classical approach to describing a 3D knotted curve is to draw a two-dimensional projected curve that is broken each time it is occluded by a piece of the whole curve that is nearer to the 3D projection point. At these occlusion points, the part of the curve nearest the projection point is continuous and the part passing underneath is interrupted for a short interval to each side of the occluding curve at the crossing (see Figure 3.6(b)). Our method of rendering such a 2D knot diagram is to attach a *thickened* curve segment in background color behind each of the curve segments that are rendered in foreground color, so that a *visual break* is created on each side of an under-crossing. Program 1 describes our 2D knot crossing-diagram rendering method.

Program 1 Procedure for rendering a 2D knot crossing diagram

```
glDisable(GL_LIGHTING);
For each curve segment
  glColor3f(BG.r, BG.g, BG.b);
  glLineWidth(δ + ε);
  draw curve segment;
  glColor3f(FG.r, FG.g, FG.b);
  glLineWidth(δ);
  glDepthFunc(GL_LEQUAL);
  draw curve segment;
  glDepthFunc(GL_LESS);
glEnable(GL_LIGHTING);
```

3.3.5.2 Rendering 3D Smooth Knots

A curve $\mathbf{C}(t)$ can be "thickened" by enveloping it in a tube. A polygonal model for tubing can be computed by attaching orientation frames at points sampled along the curve, and by tracing a 2D



(a) (b) (c)

Figure 3.18: Tubing model for generating thickened curves. (a) The orientation frames along a curve segment. (b) The polygonal wire frame model. (c) Smooth knot tubing rendered with 3D light and material properties.

profile (such as a circle) in the plane of each frame perpendicular to the curve.

An orientation frame is represented in the form of a 3×3 orthonormal rotation matrix $[\mathbf{T} \;\; \mathbf{N}_1 \;\; \mathbf{N}_2]$.

Here, $\mathbf{T}(t) = \mathbf{C}'(t)/||\mathbf{C}'(t)||$ is the normalized tangent vector determined directly by the curve geometry. $(\mathbf{N}_1(t), \mathbf{N}_2(t))$ are a pair of orthonormal vectors spanning the plane perpendicular to $\hat{\mathbf{T}}$(t) at each point of the curve $\mathbf{C}$(t) (see Figure 3.18(a)). To generate a tube, we sweep the chosen set of frames through each curve point $\mathbf{C}(t)$ to produce a set of connected points $\mathbf{X}(t)$ on the tube (see Figure 3.18(b)):

$$\mathbf{x}(t,\theta) = \mathbf{C}(t) + \cos(\theta)\hat{\mathbf{N}}_1(t) + \sin(\theta)\hat{\mathbf{N}}_2(t) \; .$$

The resulting structure is sampled in t over one full 2π period in θ to produce a tessellated tube. The base frames at each point of the curve can be computed by a variety of methods such as the Frenet-Serret [Gray(1993)] or the Bishop (parallel transport) method [Bishop(1975)].

3.4 User Applications and Feedback Results

Our implementations employ a SensAble Technology Omni PHANToM force-feedback haptic device combined with a high-performance graphics card supporting OpenGL. The user interface and graphics rendering are based on OpenGL and the haptics system is based on SensAble's OpenHaptics API. The software runs on a Dell PC desktop with a 3.2GHz Intel Pentium 4 CPU. The haptic frame rate remains above 1000Hz for most of tasks we have encountered (a haptic device requires a refresh rate of about 1000Hz in order to give a kinesthetic sense of stiff contact). The basic technical framework for haptic curve manipulation and understanding described here has been integrated into several distinct user interface environments; we mention two below.

3.4.1 Knot Exploration Interface

A touch-based pedagogical tool was developed to assist students taking a basic undergraduate topology class. The user interface was designed to help students understand the correspondence between the strict 2D approach to representing knots using crossing diagrams and the corresponding structures in 3D space (see Figure 3.19). A group of eight subjects was instructed in the use of the system and then explored a variety of 2D knot diagrams and their 3D counterparts, both before and after being exposed formally to knot diagrams in the classroom. A set of tasks involving identifying apparently different but topologically identical knots and untangling unknotted curves was presented to the subjects. The subjects were asked to give verbal descriptions and evaluations of their experiences, and the general response to the system was that it was of significant value in creating a clear mental model associating the 2D knot diagram with the 3D version of the knot. The following is a

summary of the common responses of the participants who used our knot exploration system:

- Features of the application participants liked most: force feedback during navigation on the knot, the ability to rotate and observe knots from different view points, and audio feedback indicating over and under crossings. One participant indicated the usefulness of the cast shadows of the knots in discriminating depth and height in 3D.
- Suggestions and comments: three participants indicated they would like the ability to edit knots in addition to exploration. One participant found it difficult to switch between the mouse and haptic stylus during exploration, while another user found the haptic device tiresome over extended durations.

3.4.2 Motor Assistance

A user interface was developed in which the knot data structures were supplemented by haptically-traceable curves, letters, and numbers that could be chosen easily by, for example, typing in one's name. This system has been adapted for use at a laboratory that is devoted to assisting motor-impaired children to develop curve-tracing and letter-tracing skills; initial reports are that this application is quite successful. A snapshot of the application is displayed in Figure 3.20.

3.4.3 Pseudo-Haptic 2D Knot Diagrams

The proposed haptics technique to explore the 2D knot diagrams has led to a new interaction technique to simulate 2D knot images in desktop applications without a real haptic interface. The real haptic technique can be replaced by adaptive motion of the mouse cursor on the computer screen,

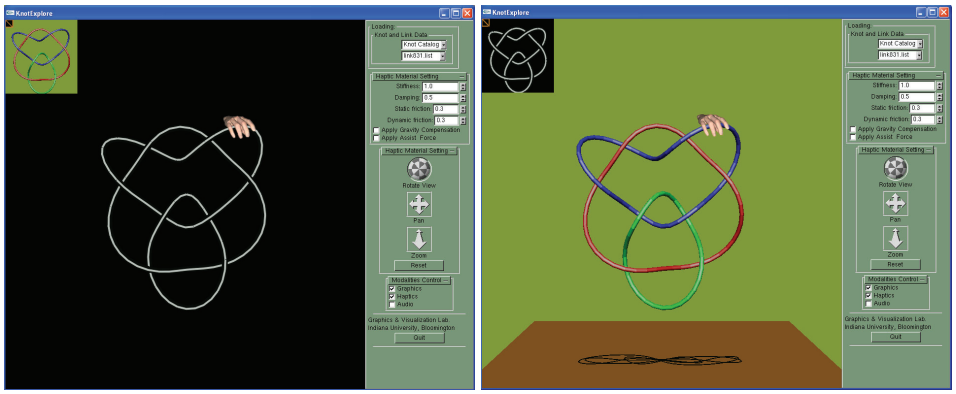


Figure 3.19: User interface for Knot exploration, which won recognition and a runner-up award in the 2005 SensAble Corporation competition: left, the 2D knot diagram mode, and right, 3D knot structure and exploration mode.

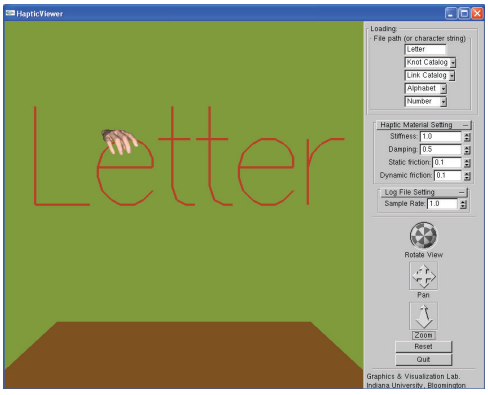


Figure 3.20: Touch-based interface to help develop repetitive motion skills such as tracking curves and letters.

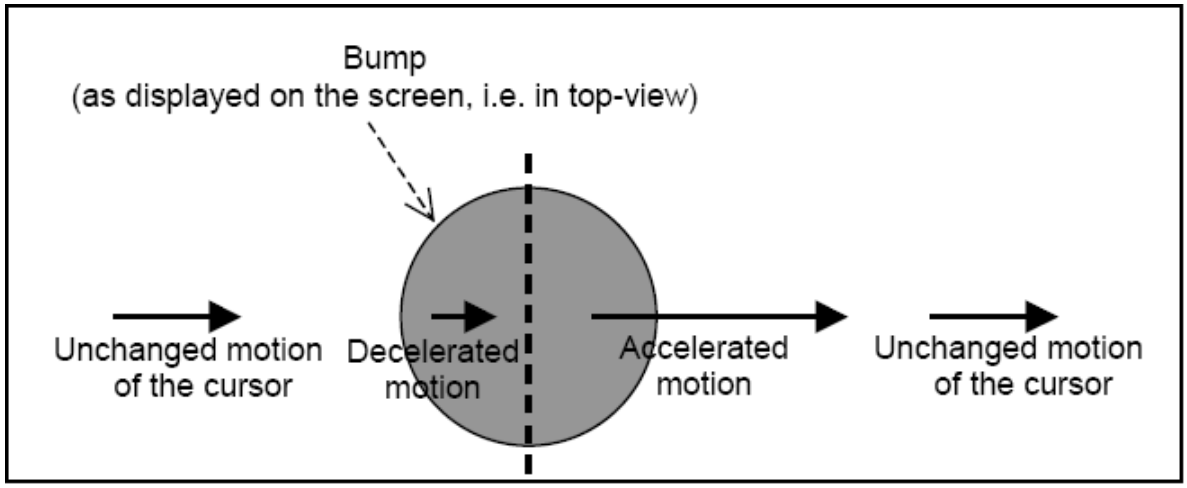


Figure 3.21: Modification of the speed of the cursor when passing over a bump.

e.g., slowing the response to simulate collision.

For our principal test case of smooth 3D knot structures projected to 2D, the main idea of our pseudo-haptic interface is to modify the motion of the mouse cursor displayed on the computer screen during the manipulation of the input device by the user. When sketching 2D knot diagrams in a real haptic interface, for example, the user would physically sense the collisions while holding the haptic stylus that renders appropriate forces; in a pseudo-haptic interface, this can be represented as a deceleration of the mouse cursor followed by a full stop to indicate that a collision has been encountered. The visual motion of the mouse pointer, which is the cornerstone of the pseudo-haptic interface, helps the user to sense the bumps, gradients, and collisions in the 2D images (see Figure 3.21) [Lécuyer et al.(2004)Lécuyer, Burkhardt, and Etienne, Lecuyer et al.(2000)Lecuyer, Coquillart, Kheddar, Richard, and Coiffet,

Sreng et al.(2006)Sreng, Lécuyer, Mégard, and Andriot].

3.4.3.1 Mouse Pointer Control

A simple method for managing the mouse movement is to position the mouse using the function *SetCursorPos* as illustrated in Program 2.

Program 2 Procedure for controlling the mouse pointer

```
void SetMousePointer(int x_ogl, int y_ogl)

POINT ptOld_scr, ptNew_scr;
GetCursorPos(&ptOld_scr);
int x_diff = x_ogl - fgetmousex();
int y_diff = y_ogl - fgetmousey();

ptNew_scr.x = int(ptOld_scr.x + x_diff);
ptNew_scr.y = int(ptOld_scr.y + y_diff);

SetCursorPos(ptNew_scr);
```

3.4.3.2 Constrained Grid Drawing Interface

A typical problem encountered in drawing on 2D screens is how to maintain and guide a user's absolute position in space. Previous work on 2D or 3D drawing has found that users could draw straight lines quite easily [Snibbe et al.(1998)Snibbe, Anderson, and Verplank], so our interface employs drawing constrained to a simple 2D grid.

3.4.3.3 Interacting with Mathematical Knots and Links

Users should be able to construct an initial object model using the interaction techniques described above. This frees them to concentrate on the important topological features of whatever they are constructing. Because these initial constructions may be crude in an aesthetic sense, it is often

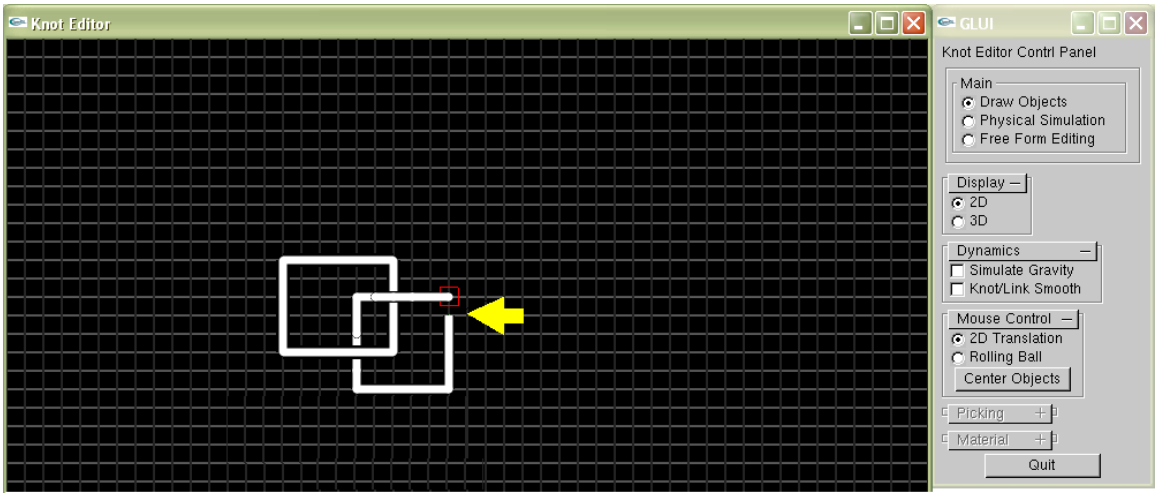


Figure 3.22: Constrained 2D drawing interface using 2D grids.

necessary to manipulate and refine the configuration. Of course, this requires a dynamical model to be set up for the embedding, and appropriate forces should be applied in order to improve the initial configuration.

Dynamical Model. For our dynamics, we employ the pseudo-physical system from Knotplot [Scharein(1998)]. The dynamical entities in the simulation are the "beads" or vertices in the knot. It is to these beads that all forces are applied. During the simulation, each bead simply moves in the direction it is compelled to by the forces that are applied to it. The "sticks" or edges in the knot are not directly involved in the dynamics, although they provide constraints on the movement of the beads.

Relaxing the Embedding. To relax the knot structure, the following sequence of events occurs in each time step during the simulation:

1. Sum the forces currently active on each bead

2. For each bead in sequence:

   (a) Determine the new position of the bead if no topological constraints were active, using the total force on that bead and the current physical model for interpretation of that force (i.e., $F \propto v$ or $F \propto a$)

   (b) If the distance moved is greater than d_{max}, clamp the distance moved at d_{max} in the same direction

   (c) Check to determine if moving the bead to its desired location will cause the knot to move into an unsafe position

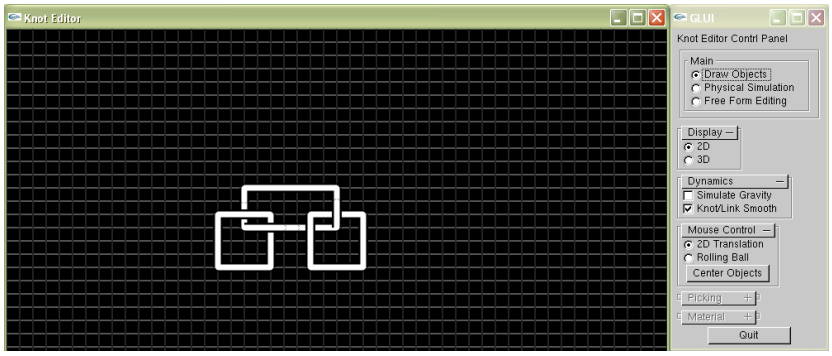
   (d) If the bead can be moved so that the knot is still in a safe position, update the position of the bead. Otherwise declare it "stuck" and do not update the position.

Figure 3.23 shows the relaxation for a simple case, from the initial embeddings being sketched by hand, to the refined embedding after being relaxed.

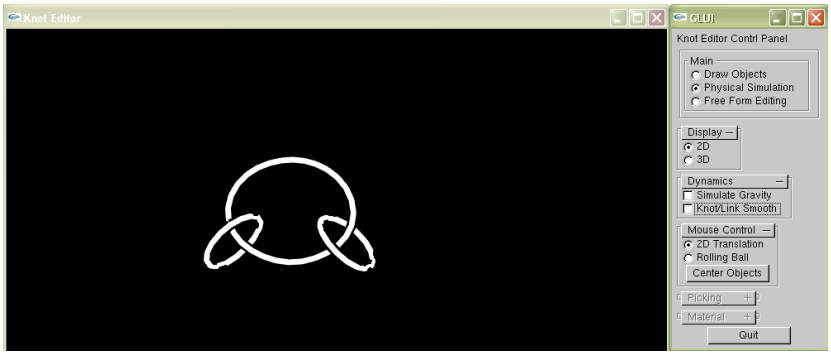
The 2D/3D system has been successfully used to manipulate a variety of knots; an example manipulation is shown in Figure 3.24.

3.5 Summary

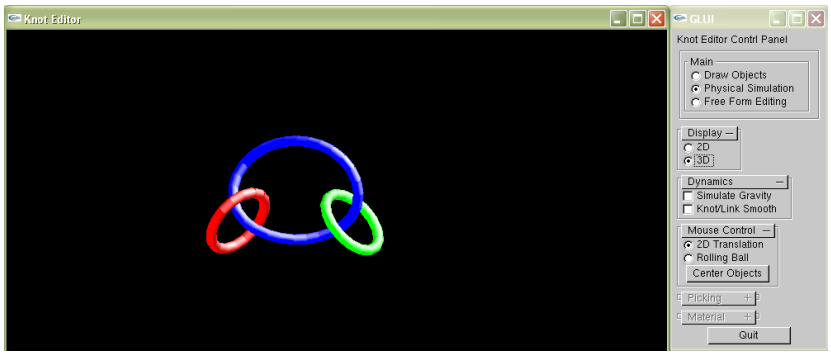
Just as we can work with two-dimensional floor plans to communicate 3D architectural design, we can exploit reduced-dimension shadows to manipulate the higher-dimensional objects generating



(a)



(b)



(c)

Figure 3.23: Relaxation of simple link. (a) Sketch an initial configuration of a mathematical link. (b) Relax the constructed link. (c) 3D display.

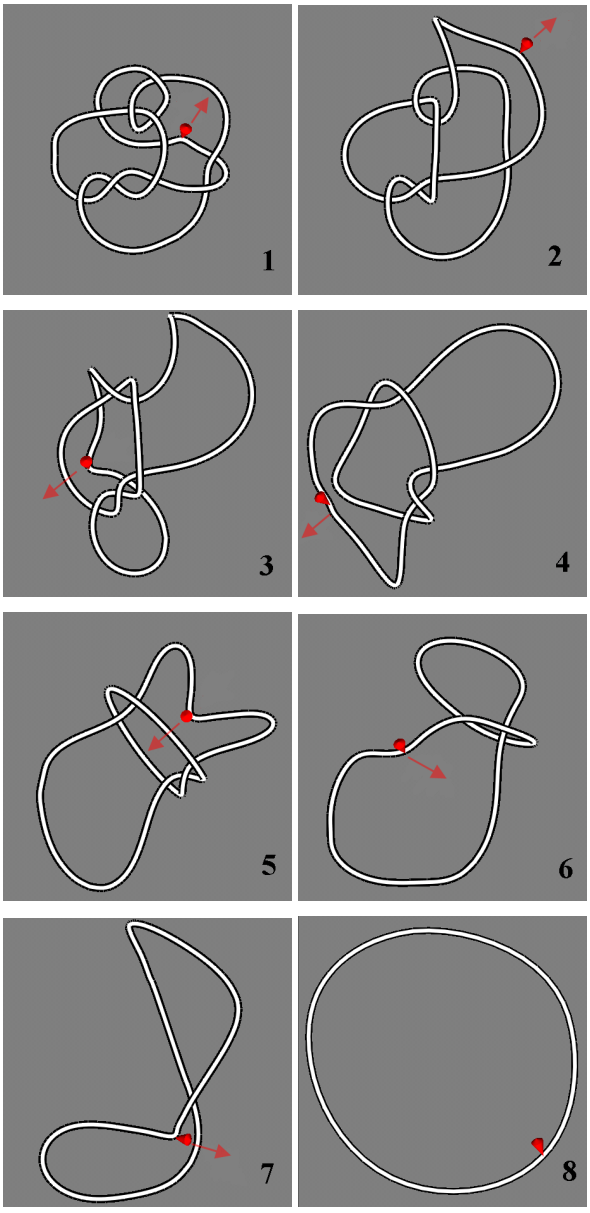


Figure 3.24: Selectively applying local deformations to the 3D curves exposes the starting object as isotopic to the unknot.

the shadows. In this chapter we introduce a teaching tool that uses 2D knot diagrams to manipulate the geometry of 3D mathematical knots via their projections; our unique 2D haptic interface allows the user to become familiar with sketching, editing, exploration, and manipulation of 3D knots rendered as images projected on a 2D shadow space. By combining graphics and collision-sensing haptics, we can enhance the 2D shadow-driven editing protocol to successfully leverage 2D pen-and-paper or blackboard skills. Building on the reduced-dimension 2D editing tool for manipulating 3D shapes, we now proceed to develop the natural analogy to produce a reduced-dimension 3D tool for manipulating 4D shapes.

4

Touching the Fourth Dimension

4.1 Introduction

> "Words and language, whether written or spoken, do not seem to play any part in my thought processes. The psychological entities that serve as building blocks for my thought are certain signs or images, more or less clear, that I can reproduce and recombine at will. The elements that I have mentioned are, in my case, visual and sometimes motor."
>
> — Albert Einstein, in a letter to mathematician Jacques Hadamard [Hadamard(1996)].

We now possess interactive graphics tools incorporating touch-responsive features that make it possible for a computer interface to implement and even to improve upon the multisensory mental paradigm described by Einstein. By exploiting such tools, we feel that we can make a non-trivial contribution to building intuition about classes of geometric problems whose intuitive, non-symbolic comprehension lies beyond the reach of the unaided human intellect. The challenge is to find specific examples where, in fact, we have problems that are simple enough to formulate clearly,

yet complex enough that even Einstein would have had trouble working out the details in "images that [one] can reproduce and recombine at will."

In this chapter we take the first steps towards this goal by designing and implementing novel multimodal methods for exploring surfaces (two-manifolds) embedded in 4D Euclidean space.

The idea of cross-dimensional understanding has long been a subject of fascination, starting with *Flatland*'s conundrum of how two-dimensional creatures might attempt to understand three-dimensional space [Abbott(1952), Dewdney(1984)]. Banchoff's pioneering work on the corresponding question of how 3D computer-based projections can be used to study 4D objects [Banchoff(1986), Banchoff(1990)] has been particularly influential. Other representative efforts include a variety of ways to render 4D objects (see, e.g., Noll [Noll(1967)], Hollasch [Hollasch(1991)], Banks [Banks(1992)], Roseman [Roseman(1993)], and Egli, Petit, and Stewart [Egli et al.(1996)Egli, Petit, and Stewart]), and to extend lighting model techniques to 4D (see, e.g., [Carey et al.(1987)Carey, Burton, and Campbell, Steiner and Burton(1987)] and [Hanson and Heng(1992)]).

Typically, 4D visualization methods employ a projection to 3D as a fundamental step; this helps the viewer to identify salient global features of the four-dimensional object, and provides structural continuity when rotating either the object's (rigid) 3D projection, or, with more difficulty, the 4D orientation matrix, which causes non-rigid deformations in the 3D projection.

While the resulting imagery supplies a sense of the object's overall shape, structural continuity is often difficult to discern. Many of the methods for providing access to the structure hidden by the 3D projection rely on modifying the rendering, e.g., by implementing a screen door view cutting

viewing holes in each facet to make "windows" into the interior, or applying 4D depth-coded color on the 3D projected surface (see Chapter 5).

Meshview, the 4D viewer program by Andrew J. Hanson and Hui Ma, implements an interactive parametric space "picker" (or point locator) for any 4D MESH file or list of MESH files. By sliding a 2D point locator in the parameter space, the user can pass through visual interruptions caused by the 3D projection and navigate on the true 4D structure [Hanson et al.(1999)Hanson, Ishkov, and Ma]. Figure 4.1 shows typical screen images from Meshview in the "picking" mode.

Although such a "4D point locator" helps to build a better mental model of a 4D surface, the perception of the continuous 4D structure is still limited. 3D projections of 4D topological surfaces often contain massive visual interruptions that can easily bury the "4D space picker." Therefore what is really needed is an enhancement of the 3D visual representation that allows intuition-building exploration of the shape itself (potentially vision-free). Thus the question arises, "How can we touch the fourth dimension?"

4.2 Motivation

People learn about the everyday world by combining sensory modalities, and knowledge of shape comes from a combination of sight, touch, and exploration. By combining computer graphics with computer haptics, which imitates the 3D sense of touch, we can provide multimodal exploration tools that can in principle improve on real life. This improvement is possible because, for example, the image of a knotted rope is interrupted where one part crosses another, and if we try to trace a *real* knotted rope with a fingertip, we will eventually collide with some part of the rope and have

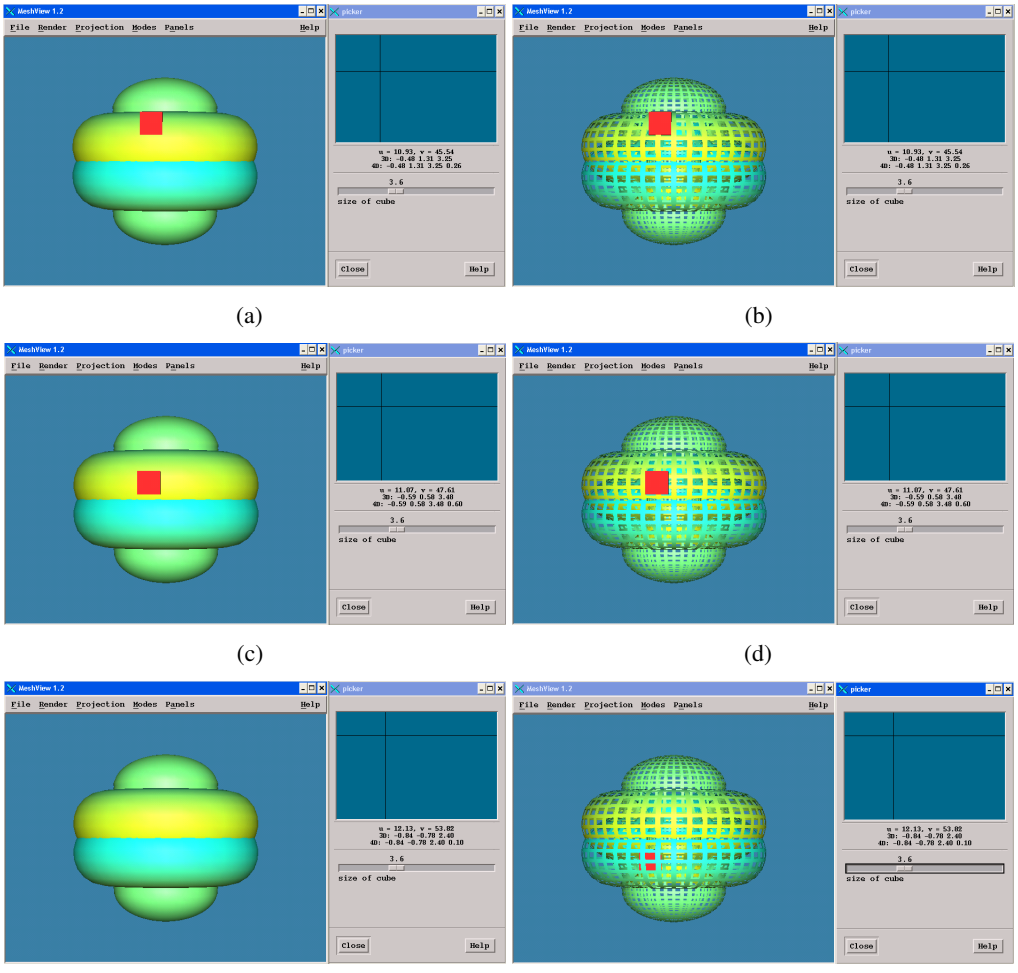


Figure 4.1: The Meshview interactive parameter space "picker" traces any 4D MESH surface. (*left*: 4D surface rendered in depth-coded color. *right*: 4D surface rendered in screen door mode.)

to interrupt our smooth passage. With a touch-based computer interface, there are no physical obstructions to the motion of the pictured "computer hand," and the entire object can be traced without the interruptions imposed by sight or a real physical model. We can therefore use the same methods to help us understand and manipulate the much more complicated case of shapes with self-intersecting *surfaces* (which arise naturally when projecting 4D to 3D) using the touch-based multimodal paradigm. If sound cues are added to describe the passage of the computer hand across a visual obstruction, one can explore a knot or a surface without necessarily having to use vision at all; when used in combination with a visual representation, the auditory cues provide additional redundant feedback enabling improved intuitive perception of spatial structure.

3D Shadow Images of 4D Topological Surfaces. To create a visual representation of a 4D-embedded surface, we typically project the entire object to 3D, construct a standard 3D computer graphics representation of the result, and render that to a 2D screen image, possibly as a stereo pair. We may then, in effect, depend on the second order effects of our practical experience with 3D structures to reconstruct a 3D shape in our minds. Full 3D information can be obtained from stereography or motion parallax, or, given the resources, an actual physical model. However, while the physical models for the types of problems we are considering can be very interesting, they may not be as useful as one might think. In Figure 4.2, for example, we see images of an actual surface corresponding to a particular cubic polynomial projected from 4D to 3D [Hanson(1994a)]; although the shape of the surface is completely without self-intersections in 4D, the model of the 3D projection has numerous very complex intersections, and, even with the physical model in hand, the self-intersections make it very difficult to trace and comprehend the intrinsic shape of the surface.

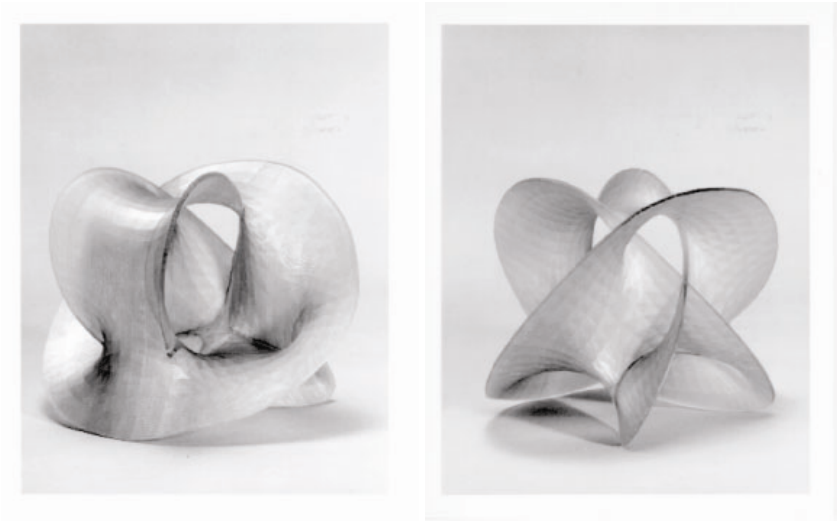
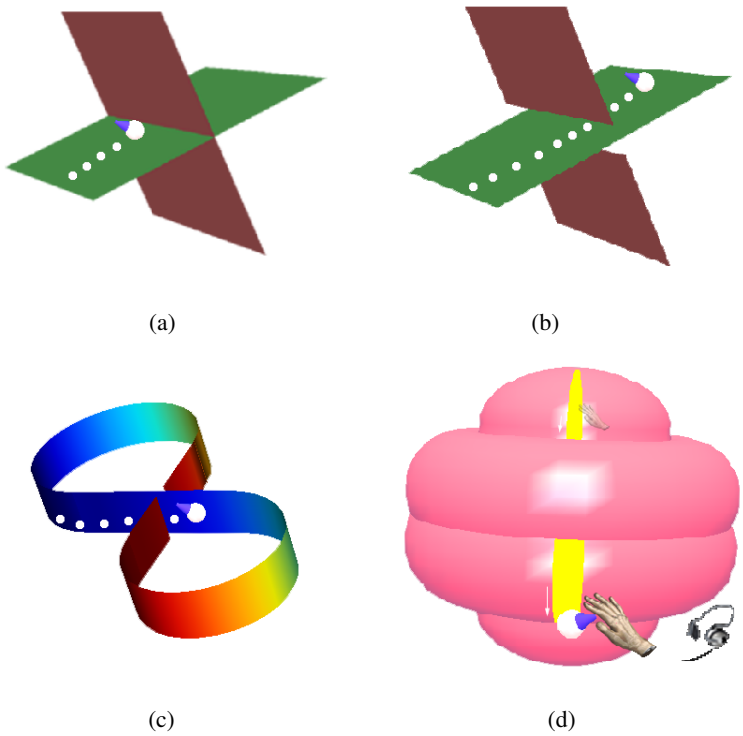


Figure 4.2: Two views of a physical model representing a highly self-intersecting surface, constructed by 3D projection from the four-dimensional mathematical description. It is nearly impossible to trace the shape continuously and with clear comprehension even when you are holding the physical object in your hands. (Model courtesy of Stewart Dickson.)

We are thus motivated to consider how 4D shapes projected to 3D might be explored to advantage with a 3D haptic probe. Figure 4.3(a) shows a pair of intersecting surfaces that apparently pass right through one another, so that we would naturally think of a path on the green surface as blocked by the brown "wall." However, if these two surface patches were distinct in 4D and the 3D intersection only an artifact of projection, one surface might be "above" and the other "below;" using either an analogy to the cutaway method used to draw a knot on a blackboard, or a corresponding analogy to depth-buffered rendering, we could erase the clashing section of the brown wall as in Figure 4.3(b), and continue our path along the *4D* green surface without interruption. Another alternative is to add an additional visual cue by assigning a surface color keyed to 4D depth relative to the projection center, as in Figure 4.3(c).

Finally, as suggested in Figure 4.3(d), we can explore a more complex continuous 4D surface using

a 3D haptic probe that avoids all physical entanglement: this is the main innovation that we shall present in this chapter — a 3D probe that respects a surface's local 4D continuity despite the self-intersections of its 3D projection. By creating auditory cues that mark crossing events we achieve a full multisensory modality that can be used with or without supplementary visual representations.



(a) (b)

(c) (d)

Figure 4.3: (a) Two intersecting surface segments, typically resulting from the projection of a 4D embedded object. (b) Crossing diagram of the surfaces corresponding to the green surface being closer to the 4D projection point than the brown surface. (c) A continuous 4D surface, where the pieces of the single surface do not touch in reality, but appear to intersect in the projection shown; depth is color-coded to show the absence of intersection. (d) Schematic summary of a tactile path with auditory cues enabling the meaningful 3D exploration of the spun trefoil, a knotted sphere embedded in 4D.

Overriding Conflicting Evidence. The key ideas of the overall scenario should now be clear. The logical series of modeling steps, the problems they induce, and the ultimate resolution of the

problems are as follows:

- *Create a model of a smoothly embedded object.* Examples are knotted curves embedded in 3D and knotted surfaces embedded in 4D, though there are obviously many other shapes that could be used.

- *Project to one lower dimension.* Various parts of the shape appear to touch each other when the smooth (non-intersecting) original shape is projected. What is seen is essentially the $(N-1)$-dimensional "shadow" of the original N-dimensional object.

- *The "shadow" object is visually discontinuous.* The projected image has self-intersections.

- *Tracing the "shadow" object is discontinuous.* If you attach a haptic probe to the projection or "shadow" object, you must detach the probe wherever self-intersections occur, pull away, and re-attach on the other side of the intersection in order to trace the logically continuous surface. Alternatively, you can allow the probe to stay attached at the intersections and choose a direction, but this does not maintain consistent local continuity.

- *Create a crossing-diagram object.* By performing a depth-buffer operation in $(N-1)$ dimensions, you can do slightly better than the shadow; the part of the self-intersecting collision that is "in front" (nearer the projection point) can be made continuous (as in a depth-buffered rendering), and the part that is farther from the projection point can be visually interrupted (though it remains physically continuous).

- *Tracing the crossing-diagram object is discontinuous.* A haptic probe attached to the crossing-diagram object is continuous as long as we cross the self-intersection on the near segment, but

once again we hit a discontinuity when we come around from the other direction and try to cross at the far segment. At this point, if the visible model interrupts the physical appearance of continuity, one again must detach the probe and move away to find the matching piece of the continuous object; however, we really want to experience the physical continuity of *both* the front and the back segments.

- *Solution: Haptically override the conflicting evidence.* To answer the question "How do we touch the fourth dimension?" we *override* the apparent visual collisions, conflicts, and gaps in the rendered image of the projection, and keep the haptic probe anchored to the *higher-dimensional* continuity that underlies the whole structure, regardless of whether it is above or below another conflicting part relative to the projection point.

- *Exploit and manipulate the "phantom effect" to advantage.* As noted, e.g., by Massie and Salisbury [Massie and Salisbary(1994)], users may initially find it disturbing to have a single point on a haptic probe constrained to a surface, while the rest of the virtual hand is passing, ghost-like, through other parts of the object. Our paradigm focuses on learning to *take advantage* of the lack of physical obstruction, which, when present, makes even the most elaborate physical model unexplorable. In addition, we have experimented with one additional feature, a *dynamic cutaway* algorithm that opens a path from the viewpoint to the haptic probe contact point, thus making the tip of the haptic probe visible to the eye even when buried behind additional layers of the surface.

In summary, we facilitate perception of the intuitive structure of an object such as a 4D-embedded surface projected to 3D by emphasizing the *contrast* between apparent discontinuities in the *visual*

representation; the interface enforces continuity in the haptic representation that emphasizes the true topology, even if this conflicts with what we see, while at the same time keeping visual contact with what is being touched as much as possible.

Overview of Interface Elements. Our interface design includes a variety of methods for providing redundant representation of 4D shape information, including the following:

- Local continuity of the haptic freedom of movement throughout the entire higher dimensional shape; visual conflicts that are artifacts of the projection have no haptic effects.
- Intelligent force-guided mechanisms for assisting the user in the haptic exploration. This assists the user in concentrating on navigating the shape itself, rather than being distracted by snags and discontinuities in the haptic path.
- 4D visual cues, including crossing diagram depth ordering methods and depth color coding.
- Attention-driven cutaways to "see through" occluding layers intersecting the line of sight between the viewpoint and the probe contact point.
- "Rolling manifold" methods that enable the user to optimize the local projection of the current local segment of the shape being touched by the cursor, providing accurate local metric information in the screen plane.
- Auditory cues that reinforce the occlusion information available from visual cues, and can potentially enable the exploration of an entire space without depending on visual representations.

4.3 Haptic Methods

Haptic interfaces can be effectively exploited to improve the sense of realism and to enhance the manipulation of virtual objects (see, e.g., [Baxter et al.(2001)Baxter, Scheib, and Lin]). The haptic exploration of unknown objects by robotic fingers (see, e.g., [Okamura(2000), Okamura and Cutkosky(2001)], [L.Kim et al.(2004)L.Kim, Sukhatme, and M.Desbrun] and [Yu et al.(2000)Yu, Ramloll, and Brewster]) is another variant that has requirements similar to ours. While we of course exploit many techniques of force-feedback and haptic user assistance that have been widely used in other interfaces (see, e.g., the work of [Zahariev and MacKenzie(2003)], [Kennedy(2002)], [Forsyth(2004)], [Park and Niemeyer(2004)], and [C.W.Reynolds(1999)]), we have found that many of the problems we encounter have been fairly unique, and thus have required customized hybrid approaches.

In this section, we introduce a "predictive navigational assistance" model for traversing a 4D polygonal object using a virtual proxy. To be effective, the navigation assistance must meet two goals: on the one hand, it must constrain the probe to the surface of the 4D object to give the impression of a "magnetic object" to which the haptic probe is drawn; on the other hand, the force should be applied in a way that facilitates exploring the local continuity of 4D objects such as knotted spheres, whose visual pictures are *interrupted* by massive self-intersections.

4.3.1 Simulated Sticky Stylus

Our basic force model implements a "sticky" stylus using a conventional damped-spring force to attach the stylus to the object surface. The proxy, a graphics point that closely follows the position of

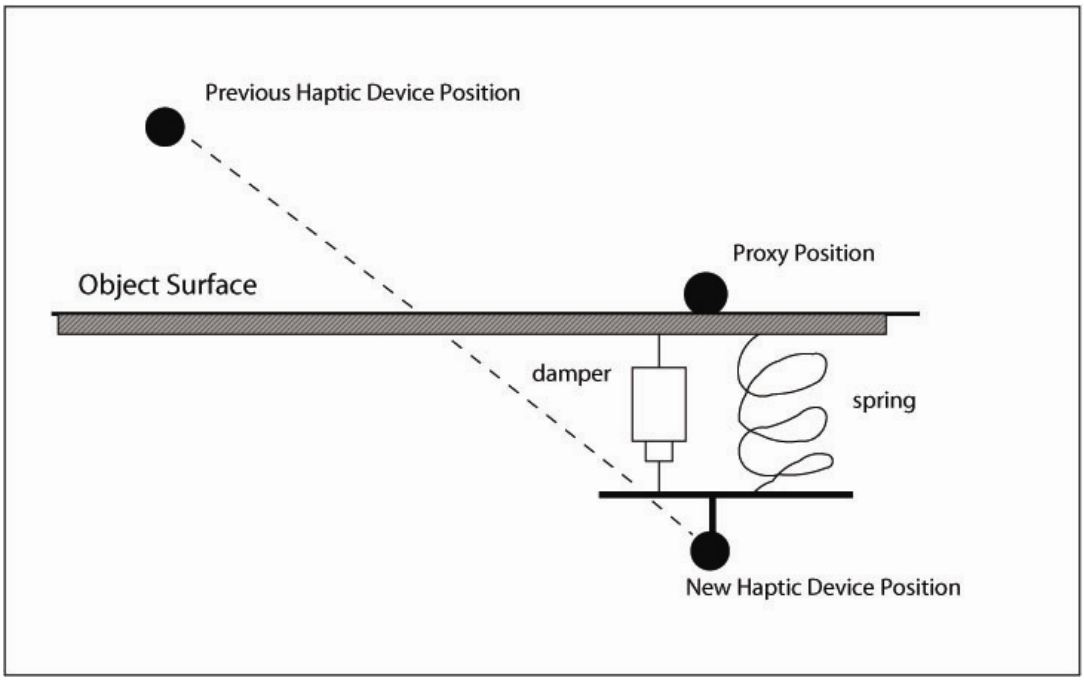


Figure 4.4: Surface constraint force model.

the haptic device, is constrained to the surface in question. The haptic rendering engine [Sen(2004)] continually updates the position of the proxy, attempting to move it to match the haptic device position, and applying the damped-spring force between the haptic device position and the proxy position.

We use the following model to compute the force in the normal direction constraining the 3D cursor or proxy to the neighborhood of the object surface,

$$\|\mathbf{f}_n\| = \alpha r^{1+\beta} \ , \tag{4.1}$$

where α is a constant and $\beta = 0$ for the standard case of an ideal (linear) spring. No force is applied in the direction tangent to the surface, thus allowing free motion and facilitating exploration of the

structure. The damping force

$$\mathbf{f}_d = -K_d\mathbf{V} \ , \tag{4.2}$$

where V is the radial velocity, is used to smooth the force feedback.

4.3.2 Constrained Navigation on the Local Surface Model

The force model presented so far in fact extends trivially to arbitrary dimensions of the vertex coordinates. The only essential difference is that in a 4D projection to 3D, each vertex has a single 4D "eye-coordinate," or depth w, in addition to the coordinates (x,y,z) of the 3D projection. To constrain the proxy to a continuous surface embedded in 4D, we introduce local submodels of the geometry that cover a neighborhood of the current contact point in the 3D projection and serve as the current active haptic contact domain.

The following steps describe the haptic servo loop model:

1. *Get coordinates.* Let the current proxy coordinate data be denoted as $(x_{\text{cur}}, y_{\text{cur}}, z_{\text{cur}}, w_{\text{cur}})$, and the previous proxy coordinate as $(x_{\text{pre}}, y_{\text{pre}}, z_{\text{pre}}, w_{\text{pre}})$.

2. *Estimate local continuity.* In order to construct a continuous local structure to guide the user's movements, we first compute the depth range for a single local update period as

$$\Delta_w = w_{\text{cur}} - w_{\text{pre}} \ . \tag{4.3}$$

The estimated local continuity of 4D depth is then given by:

$$w_{\text{cur}} - \Delta_w \leq w \leq w_{\text{cur}} + \Delta_w \tag{4.4}$$

3. *Construct local model.* The search for candidate facets begins at the current proxy location and ends when Eq. (4.4) no longer holds. These facets form the local model for computing forces. An alternate implementation could directly exploit mesh continuity in parameter space, while the current method uses only 4D proximity.

4. *Haptically override the conflicting evidence.* Now the sticky stylus can be applied trivially to the local model, as shown in Figure 4.5. The virtual proxy is constrained to a locally continuous domain to fully exploit the model's topological information, and haptic operations on this local model are fast and efficient.

5. *Update.* Set $(x_{\text{pre}}, y_{\text{pre}}, z_{\text{pre}}, w_{\text{pre}})$ to the values of $(x_{\text{cur}}, y_{\text{cur}}, z_{\text{cur}}, w_{\text{cur}})$.

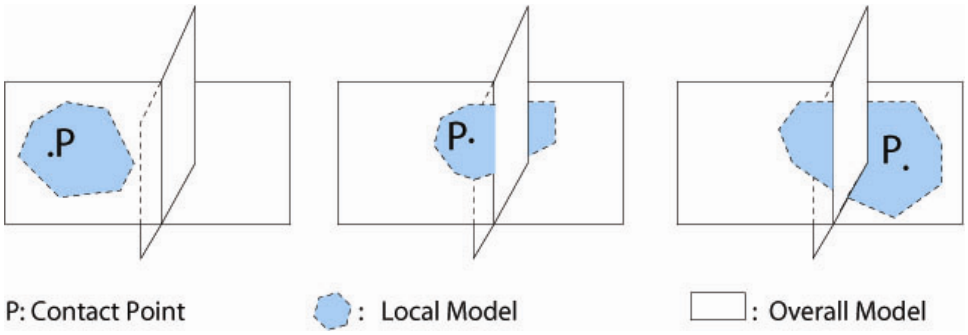
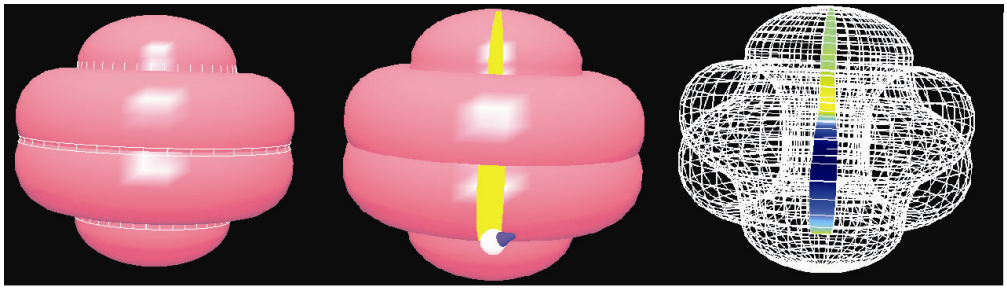


Figure 4.5: Despite apparent surface conflicts, only the local model with 4D continuity constraints is actively involved in the haptic process.

This rendering rule allows us to slide through the visual intersections as though they were ghost images while constraining motion to the smooth surface that exists in 4D but cannot be seen without

interruption in 3D. As shown in Figures 4.6(a) and (b), we can feel the four-dimensional continuity even though it is not apparent in the visual representation. Figure 4.6(c) depicts the continuous depth-encoded 4D path on the 3D projection with the shaded surface removed. The path inherits the 4D depth cues from the original 4D shape and reveals the fact that the user is navigating the true continuous underlying surface and exploring a "feelable" structure.



(a) (b) (c)

Figure 4.6: (a) This 4D object contains massive self-intersections in its 3D projection. (b) The proxy slides through the visual intersections and explores the continuous 4D structure. (c) The projected 3D path with color-coded 4D depth is exposed in wireframe mode.

4.3.3 Adding Force Suggestions

The sticky stylus *in principle* is sufficient to allow the viewer to explore the full continuity of a given surface. However, in practice, many interesting geometric objects such as 4D knotted surfaces produce complex images that snag the stylus in sharp bends and obscure the structural continuity and local surface features (see, e.g., Figure 4.2). This presents a significant challenge to the user if only the bare attractive force interface is supplied. Our purpose in this section is to present a predictive navigation interface that assists and guides the user towards the neighboring surface features without depending on visual feedback concerning the direction the hand should move.

Assuming we know the desired position and desired velocity for the stylus, we propose the following supplementary force

$$\mathbf{f}_{\text{pd}} = K_p\,(\mathbf{P}_{\text{des}} - \mathbf{P}) - K_v(\mathbf{V}_{\text{des}} - \mathbf{V}) \tag{4.5}$$

where $\mathbf{f}_{\text{pd}}$ is the force due to the predictive navigation assistance, $\mathbf{P}$ and $\mathbf{V}$ are the actual (current) contact point positions and velocities, and $\mathbf{P}_{\text{des}}$ and $\mathbf{V}_{\text{des}}$ are the desired contact point positions specified by the user if the default coordinates are to be overridden. K_p and K_v serve as stiffness and damping matrices. Starting from this point, we next investigate how Kalman Filters can be used as *predictors* to assist navigation control.

▷ **Noise Analysis.** The position and velocity data obtained from haptic sensors are typically noisy due to both the motion of the human hand and the construction of the sensors themselves, as illustrated by the data in Figure 4.7; the measured velocity data (blue) are obtained by sliding the probe over a curved surface. The noise present in the measured data makes position and velocity prediction difficult.

▷ **Kalman Filtering.** In order to predict the desired position and velocity from noisy data, we need to apply an appropriate filter. A logical candidate is the Kalman Filter (KF), which is essentially a recursive solution of the least-squares problem [Welch and Bishop(1995), Maybeck(1979)]. Among applications similar to ours, we note Simon's [Simon(2001)] application of Kalman filtering to vehicle navigation, and Negenborn's [Negenborn(2003)] study of the robot localization problem.

The KF and discrete stochastic dynamical equations for our system can be written as

State equation: $\hat{\mathbf{x}}_{k+1} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \hat{\mathbf{x}}_k + \begin{bmatrix} T^2/2 \\ T \end{bmatrix} \mathbf{u}_k + \mathbf{w}_k$

Output equation: $\hat{\mathbf{y}}_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \hat{\mathbf{x}}_k + \mathbf{z}_k$

where $\hat{\mathbf{x}}_k = \begin{bmatrix} \mathbf{P}_k \\ \mathbf{V}_k \end{bmatrix}$, and $\mathbf{u}_k$ is the random, time-varying acceleration and T is the time between step k and step $k+1$. We assume that the process noise $\mathbf{w}_k$ is white Gaussian noise with noise covariance $\mathbf{S}_w = \mathbf{E}(\mathbf{w}_k \mathbf{w}_k^T)$, and the measurement noise $\mathbf{z}_k$ is white Gaussian noise with noise covariance matrix $\mathbf{S}_z = \mathbf{E}(\mathbf{z}_k \mathbf{z}_k^T)$, and that it is not correlated with the process noise. $\mathbf{Q}$ is called the estimation error covariance, which is initialized as $\mathbf{S}_w$. The KF estimation is then formulated as follows:

Gain: $\mathbf{K}_k = \mathbf{A}\mathbf{Q}_k\mathbf{C}^T(\mathbf{C}\mathbf{Q}_k\mathbf{C}^T + \mathbf{S}_z)^{-1}$

Prediction: $\hat{\mathbf{x}}_{k+1} = (\mathbf{A}\hat{\mathbf{x}}_k + \mathbf{B}\mathbf{u}_k) + \mathbf{K}_k(\hat{\mathbf{y}}_{k+1} - \mathbf{C}\hat{\mathbf{x}}_k)$

Update: $\mathbf{Q}_{k+1} = \mathbf{A}\mathbf{Q}_k\mathbf{A}^T + \mathbf{S}_w - \mathbf{A}\mathbf{Q}_k\mathbf{C}^T\mathbf{S}_z^{-1}\mathbf{C}\mathbf{Q}_k\mathbf{A}^T$

$$\left(\mathbf{A} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} T^2/2 \\ T \end{bmatrix} \quad \mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix} \right) .$$

The KF provides optimal (minimum variance, unbiased) estimation of the state $\hat{\mathbf{x}}_{k+1}$ with the given observed data. In Figure 4.7, we have plotted 30 time units of true velocity (red), measured velocity (blue), and the estimated velocity (green). We notice that the measured data introduce significant jitter (human and mechanical), and thus cause difficulty determining the desired position and velocity for predictive force rendering. We see explicitly that the noisy observed signal can be improved using the Kalman Filter to accurately predict the velocity for the haptic stylus (green).

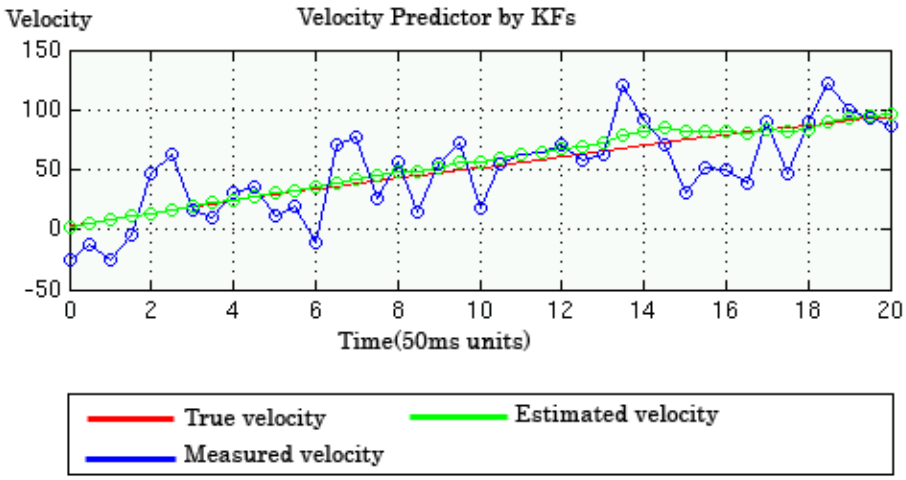


Figure 4.7: The KF helps to predict the desired velocity in the presence of noise. The desired position is calculated based on the desired velocity and provides an accurate prediction of the user's behavior.

▷ **Steering towards desired position** With the desired position $\mathbf{P}_{k+1}$ and velocity $\mathbf{V}_{k+1}$, the force suggestion can be computed using Eq. (4.5) at each servo loop. This force is then projected to local facet of the contact point, and applied to the haptic probe.

4.3.4 Experiments and Results

The proposed predictive "power assisted" force model was tested on a 2.8 GHz Pentium 4 PC running Windows XP with NVIDIA's GeForce FX 5200 graphics card. The graphics process routines were updated at about 20 Hz, whereas the haptics process ran at a 1 kHz update rate. As illustrated in Figure 4.6, the motion conforms to the local continuity constraints so that the user's probe effectively follows the shape of the projected 4D object.

Next, a series of tests were carried out to verify the effectiveness of our proposed method for constructing the local model. Haptic rendering of large and detailed 4D objects requires a significant

computational load; by choosing to render only a local subset of the polygons in the haptic process, we can significantly reduce this load.

Table 4.1 illustrates the typical saving in polygon count that we are able to achieve without any degradation of the user's perception of the interface; the local model strategy thus improves the system response at no observable cost.

4D Model	Polygon Number in Overall Model	Polygon Number in Local Model
4-Torus	900	220
Spun Trefoil Knot	2800	600
Steiner's Roman Surface	1600	360
Twist Spun Knot	2800	600

Table 4.1: Local Model vs Overall Model

On the other hand, our algorithm exploits Kalman Filtering for more accurate estimations of a *noisy dynamic system*. The KF estimates the state of a noisy system using noisy measurements. In this way, our algorithm significantly reduces the effect of noise in obtaining clean path prediction, and this facilitates smooth navigation on the surface.

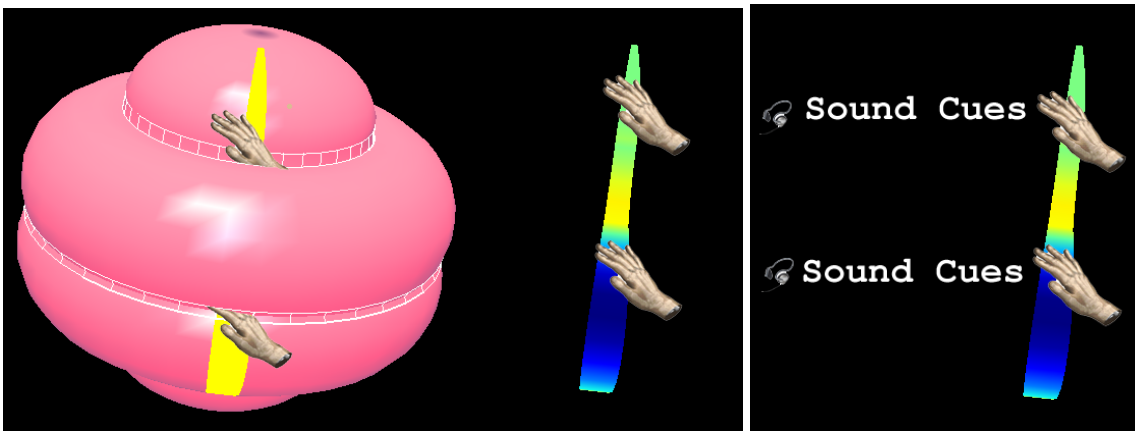
4.4 Auditory Cues

The addition of sound cues to an environment is well-known to improve presence and realism (see, e.g., [Zahariev and MacKenzie(2003)]). Audible annotation of a user's exploration of a 4D object can be used to coordinate and accent particular events. Creating an auditory "advice" to supplement the haptic exploration is significant because it makes it possible for users to build mental models of 4D geometry without necessarily depending on sight, though the visual feedback can be very useful as a redundant information source. We remark that using the multimodal interface with one's eyes

closed creates an intense experience that could conceivably improve one's internal mental model because, paradoxically, it removes potentially confounding visual distractors.

4.4.1 The Problem

Traditional techniques for perceiving visual pictures of 4D surfaces intersecting in a 3D projection rely on associating 4D depth with visual cues such as color coding, texture density, etc. Images are undoubtedly important for understanding complex spatial relationships and structures, but they may also serve to confuse the user when the visual evidence is difficult to interpret. There are currently only limited methods for presenting information non-visually, e.g., haptics and 3D hardcopy methods such as those used to produce the models in Figure 4.2. Physical models obstruct efforts at tactile exploration of continuous but self-intersecting surfaces by definition, leaving us with haptic methods; however, haptics methods alone do not necessarily convey the *context* of an exploration such as the location of a crossing or intersection. (Note that in 3D knot theory, the marking of over/under crossings in the 2D projection is an essential step.) While the location of a crossing can be flagged by the visual interruption itself, or by adding a slight jitter or jump to the haptic interface at the transition point, extending the modality to include auditory cues for these events is perhaps more versatile, like the narration of a "teacher" who is guiding us through a touch-based world. We are thus led to exploit sound, using word labels such as "over" and "under" to perceptually mark those points at which the continuous topological motion of the haptic device passes a visual disruption in the graphics image, as illustrated in Figure 4.8(a).



(a) (b)

Figure 4.8: (a) With visual feedback, the user is acutely aware of sliding through visual interruptions caused by the 3D projection. Without using visuals, however, one would be totally unaware of encountering a significant interruption unless supplementary cues such as sound feedback are supplied. (b) Sound cues supplement or replace visual cues to assist in building a clear mental model of a 4D surface.

4.4.2 The Solution

We can thus choose to supplement our design by allowing the user to explore the 4D surfaces visually and haptically, but accompanied in addition by appropriate sounds triggered, e.g., by passing the locations of illusory 3D surface intersections. Using these multiple sensory modalities to present information can overcome contradictions in the visual feedback, or even make the visual feedback superfluous.

Example. When performing a haptic exploration revealing local 4D continuity, users can feel the shape of the object via the haptic interface as described above. They can trace out the shape of a path with their computer-idealized finger, hear, for example, a change in pitch to indicate 4D depth, and have specific auditory token sounds or spoken narration triggered at special locations such as visual

obstruction transitions. A schematic picture for the audition-based supplement to the mental-model building exploratory interface is shown in Figure 4.8(b).

As the user interacts with the 4D object in its 3D projection, the visual, haptic, and auditory stimuli seamlessly contribute to a new mental picture corresponding to 4D multimodal visualization. Thus our paradigm provides an interface that assimilates the three sensory modalities of our perceptual model and makes them all work together to achieve the goal of perceiving the nature of 4D shape.

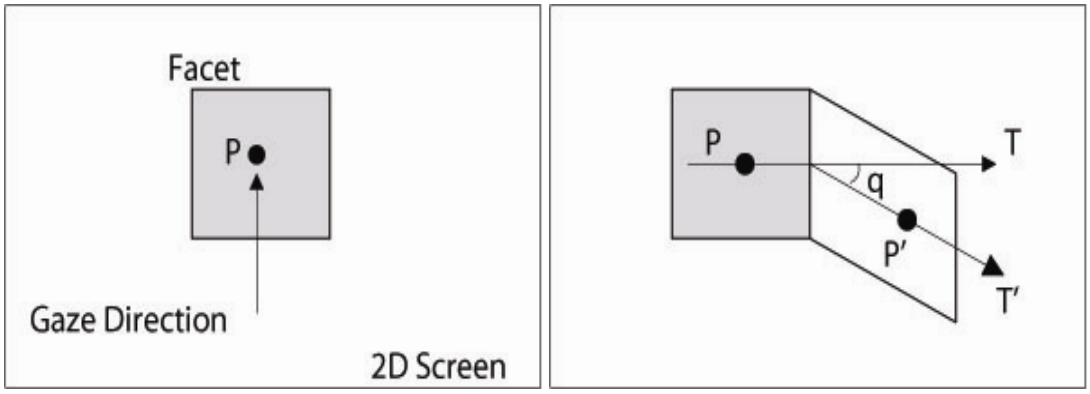
4.5 4D Haptic Rolling Manifold

There are some tasks that can be performed better by automated manipulation. For example, we can explore all points of a ball by rolling it around until the desired point comes up to face us; as advocated by Hanson and Ma[Hanson and Ma(1995a)], the presentation of arbitrary curves and surfaces to the user's view can be automatically optimized by "walking" along a local (typically geodesic) path. With the aid of the haptic probe and a 3D projection, we can adapt this family of methods to produce not only a maximal *viewable* but also a maximal *touchable* local region with each step of the walk; this permits the user with a haptic stylus to continuously sense a maximally presented portion of a neighborhood around the central point, and thus to get a richer tactile sense of the surface shape.

4.5.1 Maximizing Viewable and Touchable Aspects at Each Step

Consider a surface represented by flat polygonal facets in ordinary 3D space. Our basic maximal projection procedure then requires that the center of the facet of interest lie completely in the screen

plane, thus facilitating both viewing and touching. As shown in the Figure 4.9(a), we begin by choosing a point of interest **P** in the facet, and then use an algorithm such as shortest-path motion to transform the relation between the object and the viewpoint to maximize viewable area as we move to a new interest point $\mathbf{P}'$. If $\mathbf{P}'$ lies in the current face, no rotation is needed. If $\mathbf{P}'$ lies in an adjacent face, we rotate as prescribed below and then translate $\mathbf{P}'$ to the screen center. Translation and rotation are applied in tandem in the haptics and graphics rendering threads.



(a) (b)

Figure 4.9: (a) Surface facet with interest point **P**, maximally projected onto the 2D screen to facilitate viewing and touching. (b) 3D facet rotation.

4.5.2 Rotating between Facets

We now compute the transition across a given edge between two facets, assuming we know the unit vectors **T** (the probe's current incremental motion vector), and $\mathbf{T}'$, the projection of **T** straight down onto the target facet.

With $\hat{\mathbf{N}} = \mathbf{T} \times \mathbf{T}' / |\mathbf{T} \times \mathbf{T}'|$ giving the rotation axis, and $\cos\theta = \mathbf{T} \cdot \mathbf{T}'$ giving the rotation angle, the axis-angle rotation matrix $\text{Rotate}(\theta, \hat{\mathbf{N}})$ re-orients the user to match the projected geodesic direction,

as shown in Figure 4.9(b).

4.5.3 Tunneling the Probe to Target the Surface

During the rolling process, visual perception of the interest point can be further assisted by opening a visual window into the interior of the 3D projection; this permits the viewer to keep the haptic probe continuously in view as it traverses any touchable part of the object, thus enabling both *touching* and *seeing* the true continuous topology.

The method of tunneling the probe to target the surface is depicted in Figure 4.10. Each facet on the polygonal model of the 4D object being explored is examined for possible occlusion of the haptic probe, and thus assigned the corresponding level of sparseness for rendering. To tunnel the probe, we can apply a direct control of optical properties (e.g., the opacity value) to those occluding facets. Sometimes smooth transitions between different levels of sparseness can be applied to avoid visual misinterpretations (see, e.g., Figure 4.11(a)).

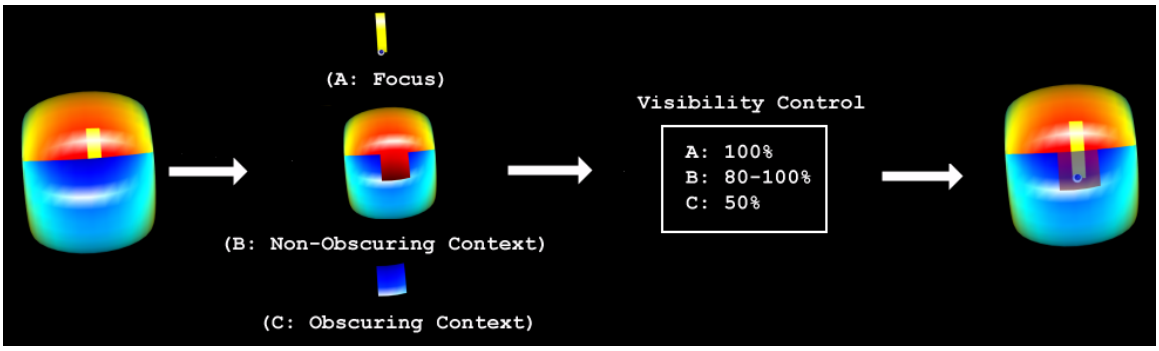
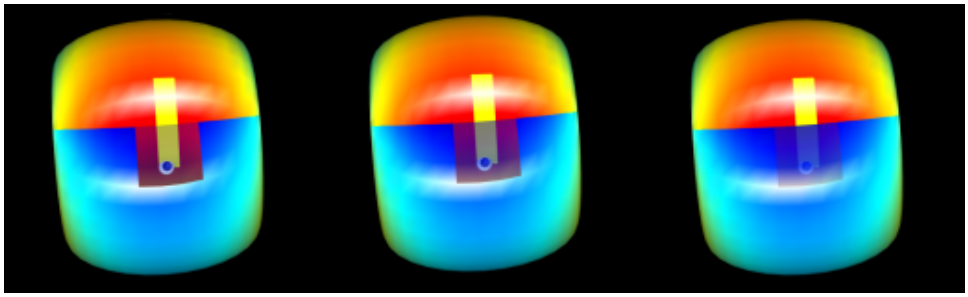


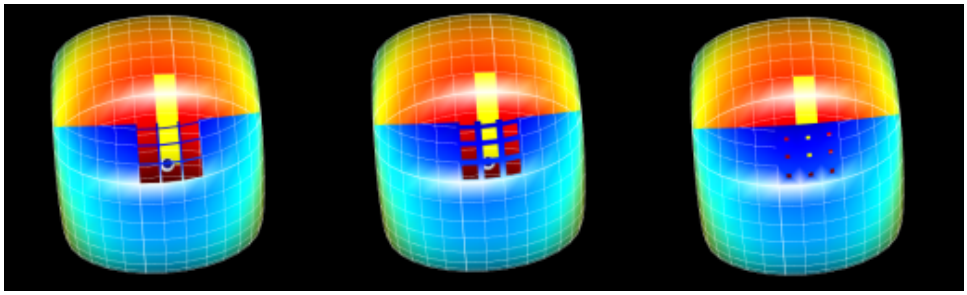
Figure 4.10: Tunneling the Probe to Target the Surface.

Sceen-door transparency is also a strategy to simulate transparency. The impact of increasing sparseness using screen-door transparency is shown in Figure 4.11(b). More examples of tunneling the

probe to target the surface are shown in Figure 4.12.



(a)

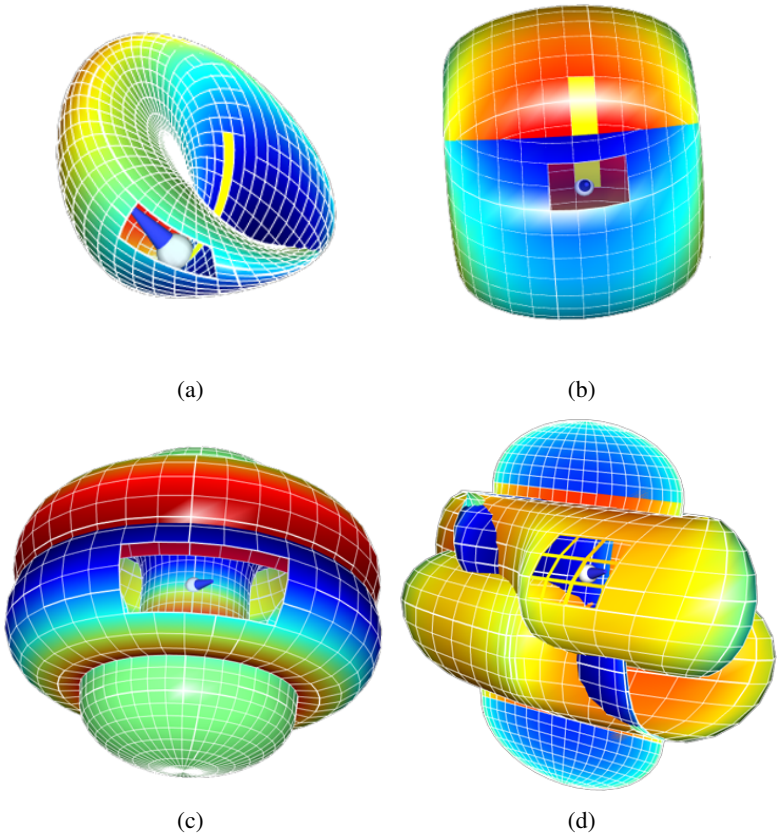


(b)

Figure 4.11: (left to right: increasing level of sparseness) (a) Opacity modulation. (b) Screen-door transparency.

4.5.4 Fixing the Stylus using a "Rubber Band Line"

Since, in the paradigm we are describing, the facet of interest is fixed to the screen center both graphically and haptically, one would in principle imagine fixing the haptic probe to the screen center where the interest facet is projected; however, this is not necessarily a useful interface choice, since the user gets no tactile feedback. Instead, we implement a supplementary interface feature based on a simulated "rubber band line" from the current haptic position to the screen center, and then compute a haptic force rendered using Eq. (4.1) with $\beta = 3$, and r the distance from proxy position to screen center. This pulls the displaced haptic probe gently back to the screen center (like



(a) (b)

(c) (d)

Figure 4.12: (a) RP^2. (b) 4D torus. (c) Spun trefoil knot. (d) Twisted spun trefoil knot.

a weak rubber band tether), and attaches it to the newly projected interest facet. In this way, we can keep the facet of interest maximally projected, and yet allow the user to feel a limited continuous neighborhood of the central focus of attention, thus getting a tactile sense of the surface shape. Figure 4.13 illustrates a 4D "rolling manifold" with the proxy constrained to the maximal viewable and touchable region with each step of the walk.

4.6 User Environment and Further Examples

Our implementation is based on a standard OpenGL graphics system with a high-performance graphics card and sound card, combined with SensAble Technology's Omni PHANToM force-feedback haptic device. Our user interface is based on OpenGL, SensAble's OpenHaptics toolkit, and a locally customized GLUI API. Various options provide haptic forces and selected auditory signals to enhance the interactive feedback. Figure 4.14 shows a representative view of the interface.

The user can load or create a variety of geometric objects for investigation, and is also presented with a wide selection of options to help maintain an optimal presentation of the object being studied.

Figure 4.15 shows the user exploring a 2-torus (technically the two-manifold $S^1 \times S^1$) embedded in Euclidean 4-space and projected orthographically to 3D. We can see that the user can effectively slide through the visual interruption and *walk* on the continuous 4D structure.

In Figure 4.16, we show a geodesic path followed by the haptics-based walking algorithm on an ordinary torus embedded in 3D (which happens to coincide with a particular perspective projection

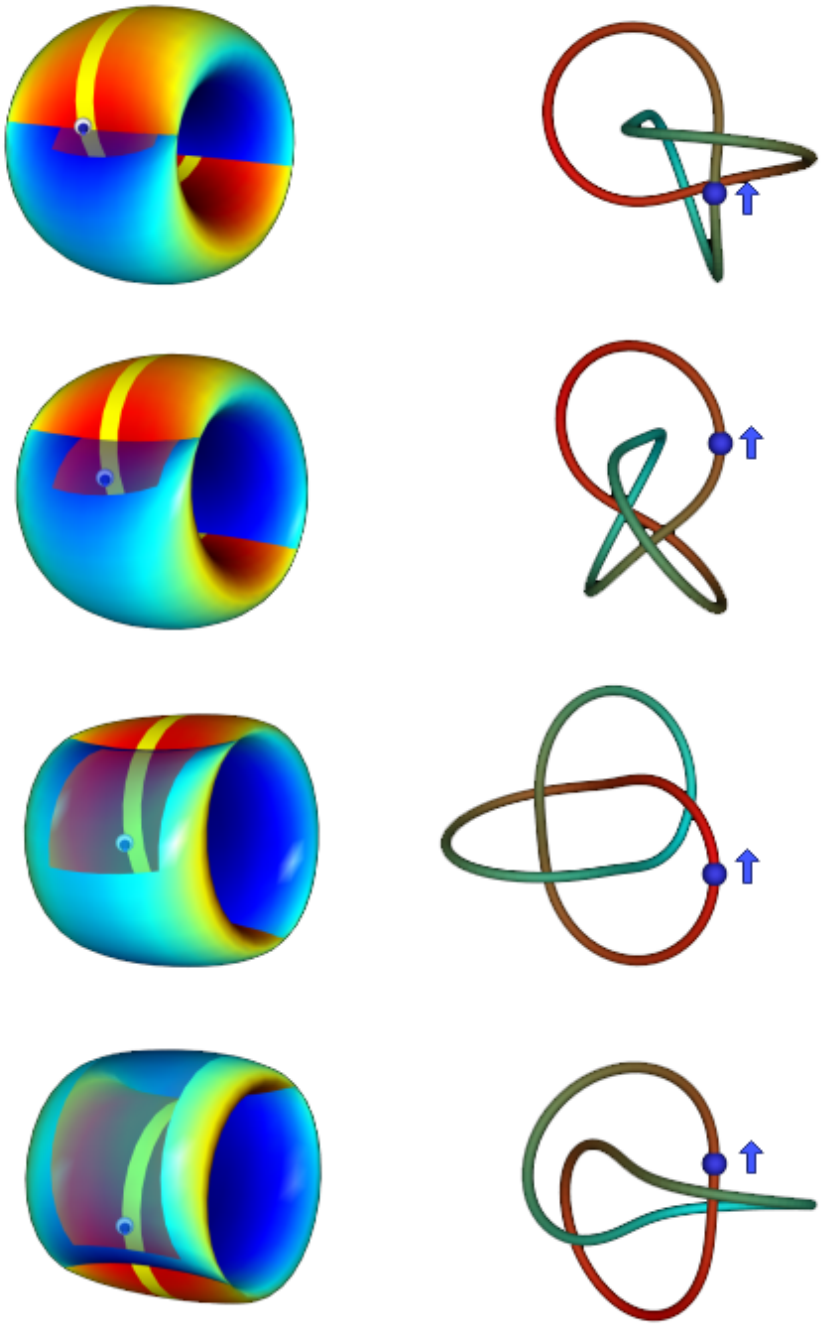


Figure 4.13: A haptic interface for constrained motion keyed to automatically align the manifold/knot orientation maximally toward the viewer. The viewer can then *see* and *touch* the surface feature of interest locally. Perceiving the touched interest point is furthermore assisted for surfaces using the automated cutaway algorithm to suppress obscuring polygons.

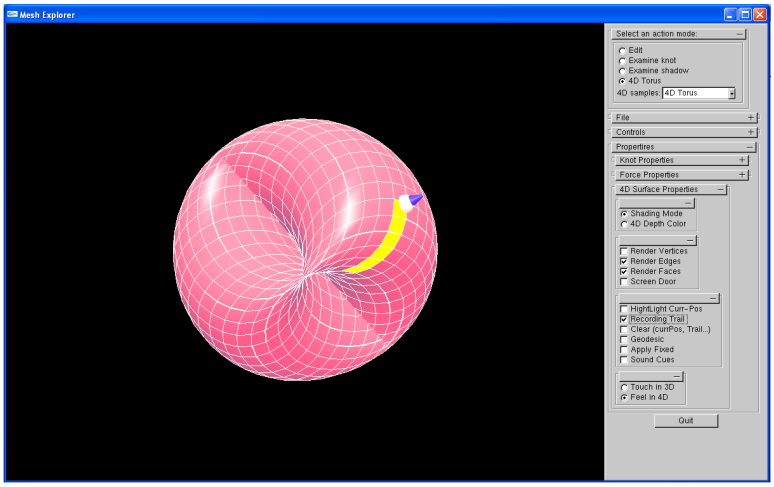


Figure 4.14: Overview of the multimodal 4D exploration interface.

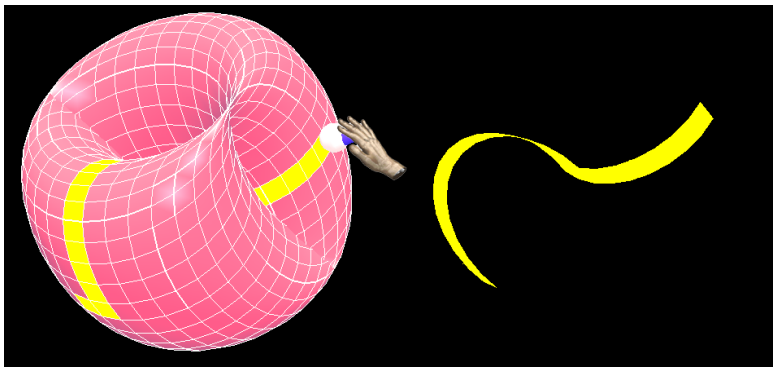


Figure 4.15: Exploration by following the continuous structure of the torus embedded in 4D, ignoring or overriding the disruptive self-intersections of the 3D projection.

of the 2-torus from 4D).

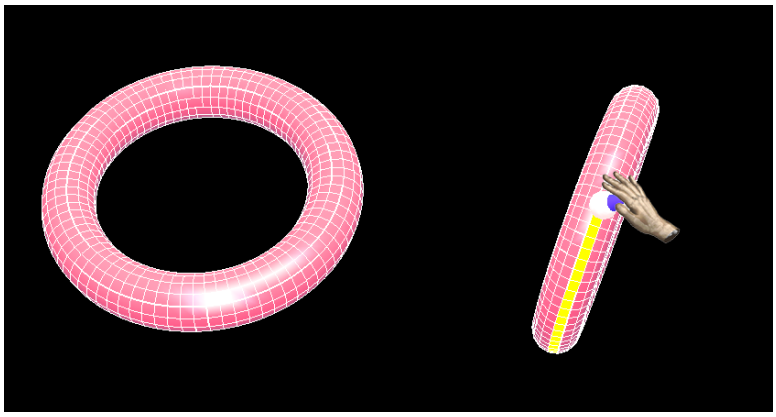
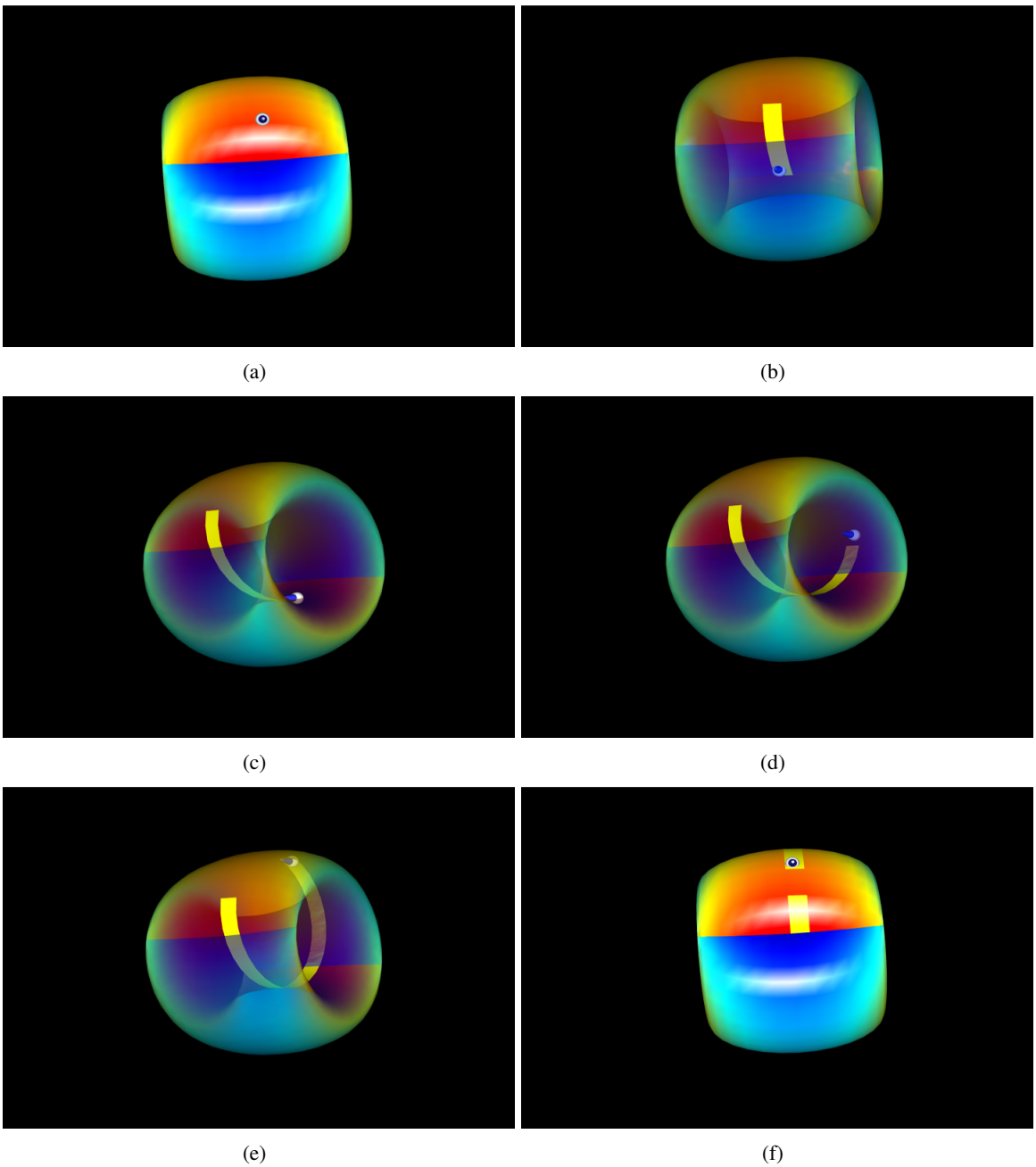


Figure 4.16: Following an approximate geodesic path, illustrating the enhanced navigation mode on the ordinary torus.

4.7 Summary

Current computer interfaces can support multimodal representations that integrate visual information with haptic feedback and interaction. Exploiting these capabilities permits us to build a particular type of kinesthetic intuition about continuous complex geometric surfaces, and is especially suited to the families of such surfaces that result naturally from projecting 4D-embedded surfaces to 3D. In accordance with the best visualization science practice, different features can be applied to the overall images, using attention-driven cutaway algorithms to reduce the cognitive load still further.

In this chapter, we discussed the design and implementation of a set of methods to *touch* and *see* geometric structures with a combination of haptic and visual sensing. We can begin to imagine the way that Einstein might have thought about the description of higher spaces by combining “visual



(a) (b) (c) (d) (e) (f)

Figure 4.17: Multimodal exploration of the 4D torus with transparent view.

and sometimes motor" sensations. With the multimodal approach advocated here, many facets of our intuitive real-life experiences can in principle be transferred to the fourth dimension.

5

Collisions in Four Dimensions

5.1 Intersecting Shadows from Higher Dimensions

5.1.1 2D Example

To help understand shadows from higher dimensions, we again begin with a family of images corresponding to a 2D projection of the neighborhoods of the crossing points of two 3D curve segments. This could in principle be a pair of 2D curve segments, as though drawn with a pen on 2D paper. If all we can see is the pen strokes or the shadow of the 3D crossing, we find the result in Figure 5.1(a), which is devoid of 3D information, and shows collisions. This problem is typically overcome by employing the "crossing diagram" method illustrated in Figure 5.1(b); this corresponds essentially to a depth-buffered rendering with some embellishments to emphasize discontinuities in depth. By thickening the curve to give it geometric structure and shading, we can get the additional improvement shown in Figure 5.1(c). Now we can begin to see that collisions in higher dimensions take place only if there is also a collision in the shadow, but that collisions in the shadow dimensions may not necessarily imply collisions in the source dimension.

(a) (b) (c)

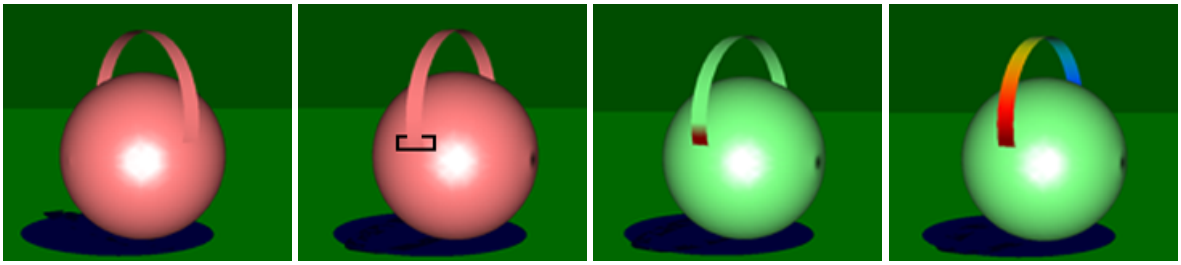
Figure 5.1: (a) 2D shadow diagram of crossing curves for two 3D rings. (b) Knot diagram representation of the linked rings provides 3D depth information. (c) Rendering with light and material adds apparent 3D geometry and shape to the 2D image.

5.1.2 3D Example

The explanation we just used for projecting surfaces from 3D to 2D has an exact analog that applies to projection from 4D to 3D. We find that the metric properties are easier to understand if we use an orthogonal projection, but in selected circumstances, perspective 4D projection from the focal point of a 4D pinhole camera can also serve to reveal essential structures. The typical side-effect of projection is that the resulting surfaces can intersect in the 3D projection, even when there are no intersections or singularities of any kind in 4D space.

We are thus motivated to extend the 2D arguments of Figure 5.1 to one dimension higher, and to show how 4D shapes projected to 3D can be created and edited with a 3D haptic probe. The first step is to note that the analog of Figure 5.1(a) is just a pair of intersecting surfaces that apparently pass through one another, as shown in Figure 5.2(a). This example is instructive even in 3D, as there is no particular reason that we cannot make (at least in computer graphics) two surfaces that pass through one another. Proceeding to the exact analog of Figure 5.1(b), we need to assume that

one surface is "above" the other (this time in 4D instead of 3D), and that a 3D volumetric depth-buffered rendering would render one surface as occluding the other, as shown (again with slight embellishment) in Figure 5.2(b). We can of course use additional visual information such as color-coded 4D depth relative to the projection center to denote 4D depth, distinguishing the "above" and "below" parts as in Figure 5.2(c). Finally, if we smooth the 4D depth, we can obtain a more uniform representation whose depth-coded projection is shown in Figure 5.2(d).



(a) (b) (c) (d)

Figure 5.2: (a) Two intersecting surfaces in 3D, typically resulting from the projection of a 4D scene to its 3D shadow. (b) Crossing diagram of the surfaces; the front portion of the ribbon surface is closer to the 4D projection point than the spherical surface. (c) Above-below crossing markings using 4D depth coding. (d) Color depth coding of ribbon modified by smoothing its 4D depth; just the top of ribbon is now at same 4D distance from the projection point as the entire sphere.

5.1.3 Various Modes for Studying Intersecting Shadow Images

5.1.3.1 Visual Modes

We adopt the fundamental philosophy that we wish to have a *holistic viewpoint* of a 4D surface, so we can see the whole object (in principle) at once, without using slicing or animation. Since humans cannot actually see in 4D, the only way we can create a holistic depiction of a 4D-embedded surface is by projecting it from a 4D projection point into a 3D volume that effectively implements a volumetric retina. Once we have done this, we have access to a wide variety of methods that

can be used to attach explanatory visual notation to the 3D projected structure. Although there are indeed methods, mentioned earlier, for creating renderings based on 4D light, we will focus here on a subset of basic approaches that exploit 3D visual methods and lighting, believing that this level of simplicity facilitates clarity in the implementation, and well as interaction speed.

Direct 3D Projection. The most straightforward image of a 4D object results from projecting it to 3D, much as the structure of a generic 3D knot was projected to 2D. Orthogonal projection is easy to implement and usually sufficient, although we can also get some additional relative depth insight from perspective projection using a 4D pinhole camera model. Figure 5.3 is a representative image of the 3D orthogonal projection of the spun trefoil, one of the most elementary knotted spheres in 4D. The bare projected image loses the important spatial ordering of the 4D object, revealing neither a continuous 4D structure nor the fact that in 4D nothing actually touches anything else. The 3D projection has many parts "buried" in the interior and contains many self-intersecting surfaces that cannot be removed from the representation we *see*; these characteristics of direct 3D projection interrupt the perception of continuous 4D structure.

Screen Door View. Our first attempt to get at the structure hidden by the direct 3D projection keeps the opaque basic surface geometry, but systematically cuts viewing holes in each facet to make "windows" into the interior. This "Screen Door" drawing method is a standard approach for revealing the interior of a solid opaque object: it is used when structures lying inside or passing through an opaque object are of as much interest as the exterior. Instead of rendering the polygons through pixel masks as in, e.g., [Mulder et al.(1998)Mulder, Groen, and van Wijk], we prefer

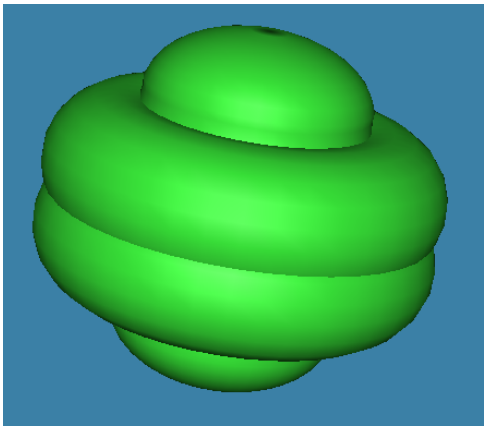


Figure 5.3: Massive self-intersections can result from the projection of a 4D surface with no self-intersections to its 3D image.

a faster screen door cutout algorithm, which explicitly removes parts of the original facet object to expose the interior structure. The screen door view, shown in Figure 5.4, exploits the standard depth buffer to avoid ambiguities with respect to spatial ordering, provides a sharp contrast between foreground and background objects, and facilitates the understanding of spatial relationships. Although screen-door rendering makes much of the interior structure visible, it still obscures the fact that in 4D nothing touches anything else.

A brief synopsis of the algorithm we use is as follows: If we consider a surface represented by quadrilateral facets in ordinary 3D space, then for each facet f, we calculate the midpoint $\overrightarrow{m}$ and the normal $\overrightarrow{mn}$ at the midpoint, computed from the normals attached to each vertex of the facet. As shown in Table 1, the facet is then replaced by a quad-strip circuit around the outer vertices and the set of inner vertices computed from a weighted average of the midpoint and each outer vertex. This is very fast and efficient to implement in OpenGL, for example.

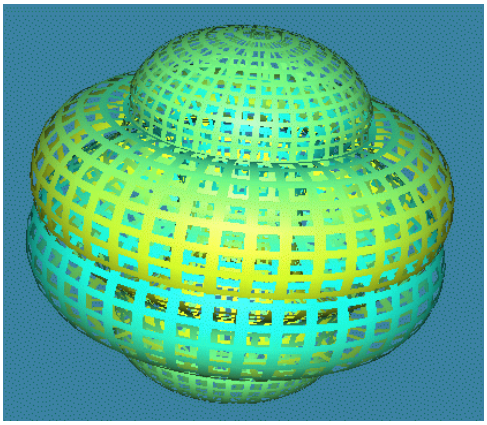


Figure 5.4: Screen door rendering of the spun trefoil knotted sphere, including 4D depth-coded color.

Transparent Surface Rendering. Another possible strategy is to reveal internal structure using transparency, and this option is shown in Figure 5.5. By turning an opaque object into a semi-transparent object, we can easily see through the surface and study some features of the interior spatial structure. However, the more complex an object is, the less useful the transparent rendering becomes.

The 3D projections of 4D objects usually have complex internal properties and extensive self-intersections; the standard OpenGL blending support thus requires back-to-front polygon sorting in order to obtain anomaly-free alpha blending, and this is typically accomplished by one of several textbook methods such as the BSP tree; we note that there is a slight additional performance expense for interactive systems, since the BSP tree must be computed with each 4D rotation that changes the polygons in the 3D projection.

Depth Coloring and Cutting Away. Color encoding is an effective way of communicating relative 4D surface depth in a 3D projection. By identifying each color table element with a different

Table 1. Algorithm: Screen Door Cutout for each Facet *f*

$\overrightarrow{m}$ ⟵ midpoint of facet *f*;
$\overrightarrow{mn}$ ⟵ approximate normal for $\overrightarrow{m}$;
t ⟵ radial proportion of cutout;
glBegin(GL_QUAD_STRIP);
  for each vertex $\overrightarrow{v}$ with normal $\overrightarrow{vn}$ on the facet *f*
    $\overrightarrow{wv}$ = (1 - t) × $\overrightarrow{v}$ + t × $\overrightarrow{m}$;
    $\overrightarrow{wvn}$ = (1 - t) × $\overrightarrow{vn}$ + t × $\overrightarrow{mn}$;
    Process weighted 4D color, texture for $\overrightarrow{wv}$;
    glNormal3dv($\overrightarrow{wvn}$);
    glVertex3dv($\overrightarrow{wv}$);
    Retrieve 4D color, texture for $\overrightarrow{v}$;
    glNormal3dv($\overrightarrow{vn}$);
    glVertex3dv($\overrightarrow{v}$);
  end;
glEnd();

---

height, we can create a visualization in which a shape will truly intersect itself only if it contains two points that have both the same 3D position and the same color. Figure 5.4 in fact contains a color 4D depth encoding in addition to the screen-door transparency, and careful inspection would show that no two colors are exactly alike at the 3D surface-intersection curves.

A different depth encoding strategy exploits the discrete above/below structure, and employs crossing markings or physical removal of the self-intersections in the 3D projection. The marking method may be thought of as a special "local" case of the general 4D depth color-encoding method: In Figure 5.6(a), we mark in blue the neighborhood of the surface intersections that are *nearer* the projection point in 4D, and in red the portions that are *farther* from the projection point in 4D. The result is precisely analogous to our crossing diagram in Figure 4.3, but without an actual cutaway; to get the exact analog, we can implement a complete cutaway of the more distant surface elements, as shown in Figure 5.6(b). Clearly there are many variants of this, with different choices of markings,

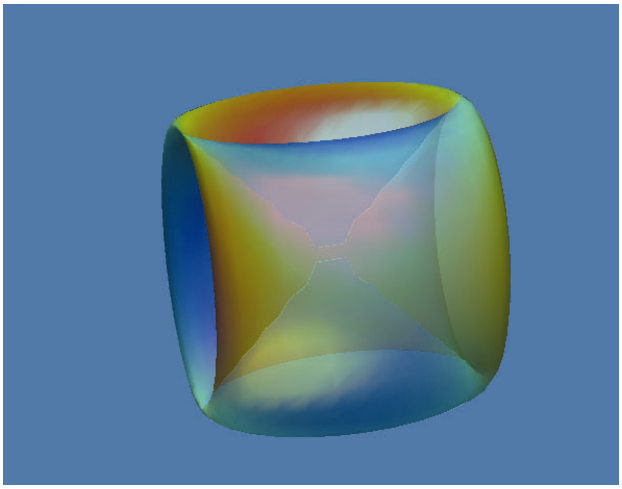


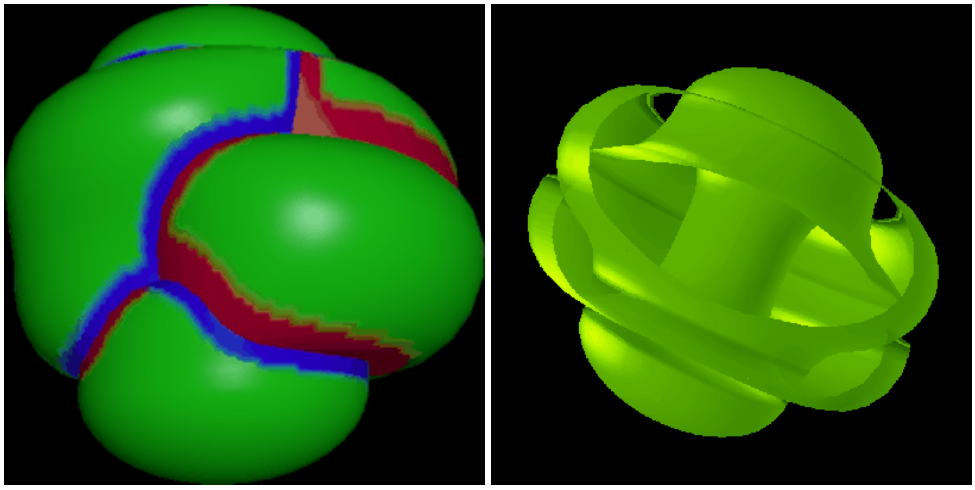
Figure 5.5: Transparent rendering as an alternate way to expose the internal structure of a self-intersecting surface.

cutaways, and even transparent blended combinations of the two.

5.1.3.2 Applying General Rotations

We are now familiar with the fact that images of 3D curves intersect in the 2D projection, and that 4D surfaces can intersect in 3D, exhibiting interruptions where one sheet crosses the other. Comprehension of a 3D crossing from its 2D diagram can be assisted by various visual modes, e.g., rendering the knot-crossing diagram. Above-below crossing markings on the intersecting 3D projected images of 4D surfaces are similarly helpful.

Another approach to understand shadow images from higher dimensions is to apply general rotations to objects projected from higher dimensions; such a rotation will smoothly alter the shadow images. For example, if shadows of 3D rings falsely appear to be linked, a properly chosen 3D rotation will detach the two shadow images from each other. The analogous observation holds when



(a) (b)

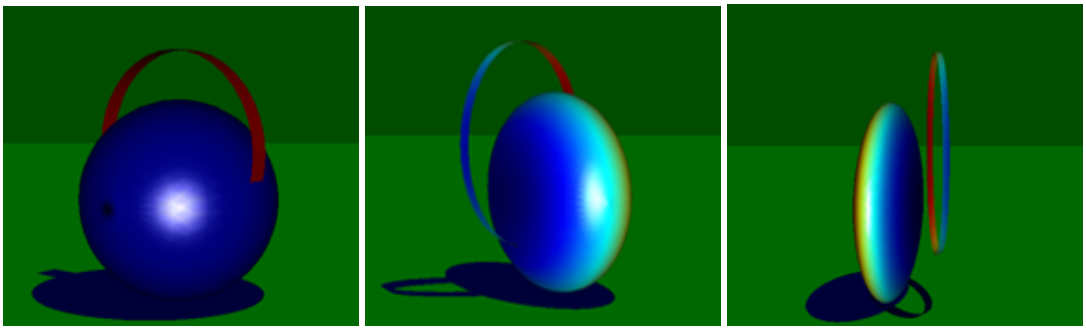
Figure 5.6: (a) Illustration of above-below crossing markings for the twist-spun trefoil, a surface embedded in 4D that, surprisingly, is actually not knotted. (b) 3D window into 4D depth cutaway of the spun trefoil knot.

we apply 4D rotations to 4D objects represented by their 3D shadows, as shown in Figure 5.7.

Figure 5.8 shows that even though we can apply various 4D rotations to a real 4D link structure, the two shadow images always intersect each other in 3D, very much like the 2D shadow diagram of crossing curves for two linked 3D rings (see Figure 5.1).

5.1.3.3 Haptic Method

In Chapter 4, we presented a paradigm for the multimodal exploration of shape that exploits haptic methods to overcome the limitations inherent in graphics images and physical models. We exploited the free motion of a computer-based haptic probe, a "computer-simulated hand," to support a continuous motion that follows the *local continuity* of the object being explored; in our principal test case of 4D-embedded surfaces projected to 3D, this permitted us to override the visual conflicts and



(a) (b) (c)

Figure 5.7: Applying general rotation to intersecting 3D shadow images. (a) The projection of two unlinked 4D embedded objects can appear linked. (b) 4D rotation changes the projected images. (c) A particular 4D rotation detaches the 3D projected images.

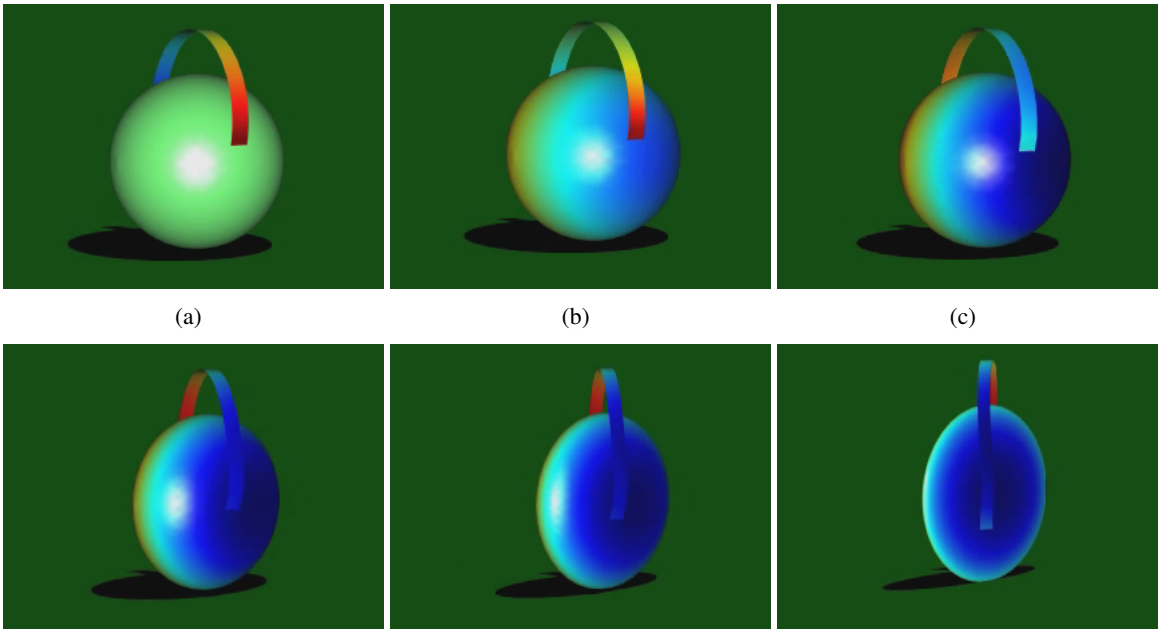
perceive the full continuity of the surface as though in fact we were *perceiving* an actual 4D object (see Figure 5.9).

5.2 Geometry in Higher Dimensions

In this section, we introduce a set of methods for detecting and sensing collisions in four dimensions to go beyond what can be done with crossing diagram methods or by applying general rotations alone listed in the above section.

5.2.1 Vector Operations and Points in 4D

For the most part, vector operations in four space are simple extensions of their three-space counterparts. For example, computing the addition of two four-vectors is a matter of forming a resultant vector whose components are the pairwise sums of the coordinates of the two operand vectors. In the same fashion, subtraction, scaling, and dot-products are all simple extensions of their more



(a) (b) (c)

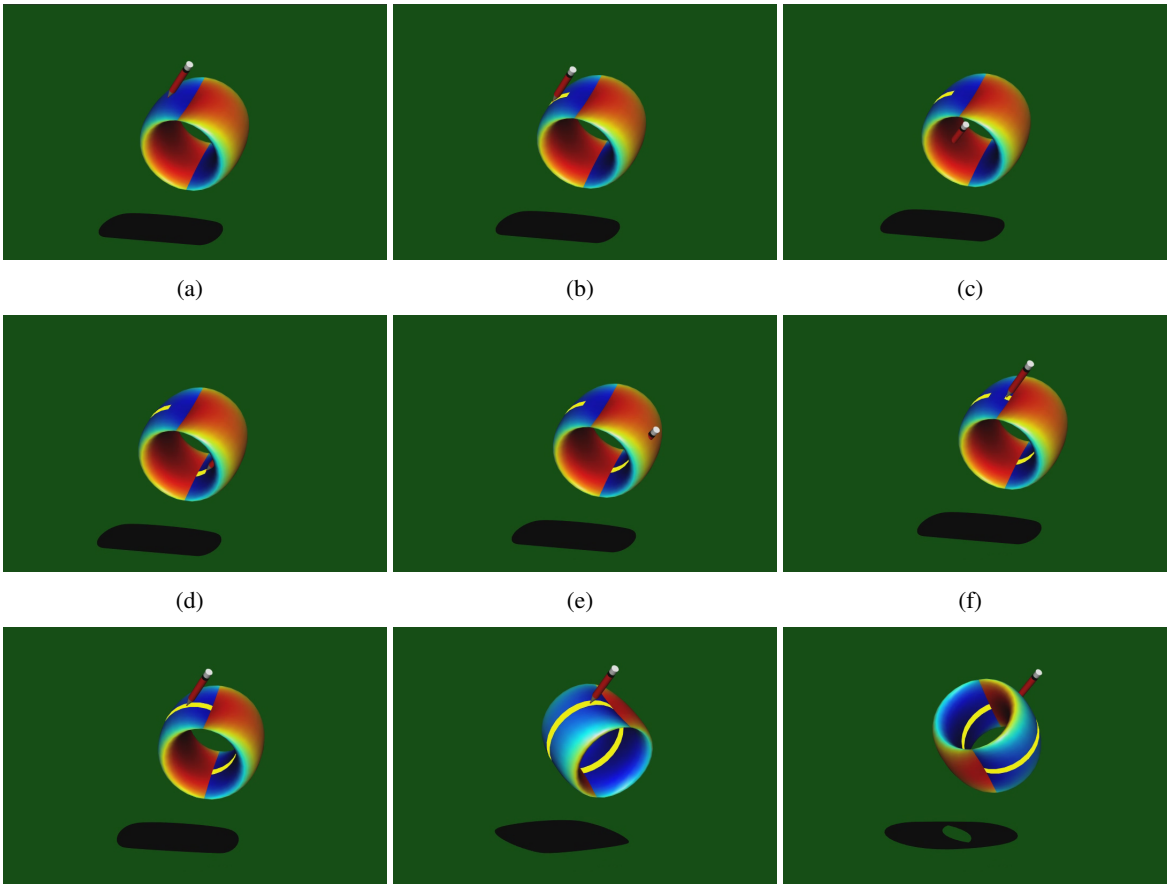
(d) (e) (f)

Figure 5.8: (a)-(f): Applying 4D rotation to change the projected images of a real 4D link structure does not detach the 3D projected images.

common three-vector counterparts.

In addition, operations between four-space points and vectors are also simple extensions of the more common three-space points and vectors. For example, computing the four-vector difference of four-space points is a simple matter of subtracting pairwise coordinates of the two points to yield the four coordinates of the resulting four-vector.

For completeness, the equations of the more common four-space vector operations follow. In these equations, $U = [U_0, U_1, U_2, U_3]$ and $V = [V_0, V_1, V_2, V_3]$ are two source four-vectors and k is a scalar value:



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 5.9: (a)-(i): We can exploit the free motion of a computer-based haptic probe to support a continuous motion that follows the local continuity of the object being explored.

$$U+V=[U_0+V_0,U_1+V_1,U_2+V_2,U_3+V_3]$$

$$U-V=[U_0-V_0,U_1-V_1,U_2-V_2,U_3-V_3]$$

$$kV=[kU_0,kU_1,kU_2,kU_3]$$

$$U\cdot V=U_0V_0+U_1V_1+U_2V_2+U_3V_3\ .$$

The main vector operation that does not extend trivially to four-space is the cross product. A three-dimensional space is spanned by three basis vectors, so the cross-product in three-space computes an orthogonal three-vector from two linearly independent three-vectors. Hence, the three-space cross product is a binary operation.

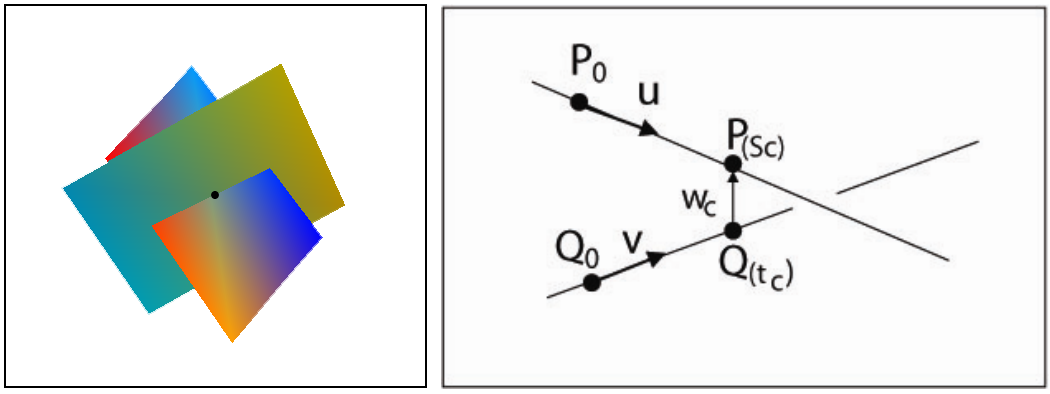
In N-space, the resulting vector must be orthogonal to the remaining $N-1$ basis vectors. Since a four-dimensional space requires four basis vectors, the four-space cross product requires three linearly independently four-vectors to determine the remaining orthogonal vector. Hence, the four-space cross product is a trinary operation; it requires three operand vectors and yields a single resultant vector [Hollasch(1991), Hanson and Heng(1992), Hanson(1994b)]:

$$X_4(U,V,W) = \begin{vmatrix} U_0 & V_0 & W_0 & \hat{\mathbf{e}}_0 \\ U_1 & V_1 & W_1 & \hat{\mathbf{e}}_1 \\ U_2 & V_2 & W_2 & \hat{\mathbf{e}}_2 \\ U_3 & V_3 & W_3 & \hat{\mathbf{e}}_3 \end{vmatrix} .$$

5.2.2 Distances in 4D

To understand the non-intuitive mechanisms of 4D collision, let us start with a pair of two-dimensional planes through the origin in four-dimensional space (see Figure 5.10). The two squares intersect in a single 4D point at the origin. In the three-dimensional projection, although the planes appear to intersect along an entire line due to the annihilation of the w dimension, when the surfaces

are 4D depth color-coded, we can see that there is just one pair of points with the same fourth coordinates as well as the same coordinates in the 3D projection. *Note that 4D collisions take place only if there is also a 3D collision in the shadow, but that 3D shadow collisions may not imply 4D collisions.*



(a) (b)

Figure 5.10: (a) 4D collision: surfaces are color-coded to indicate their depth in four-space relative to the projection point, so we observe that there is just one pair of points with the same fourth coordinate as well as the same first three coordinates. (b) Closest points between 4D lines.

4D Distance Based on Projection. Figure 5.10(b) illustrates the basic case for 4D collision detection; a 4D depth collision test must be performed along the intersecting lines of the projected images of 4D surfaces. 4D collision occurs if, and only if, one or more pairs of points are located with same fourth coordinate along the intersecting line.

Now let 4D surfaces $\mathbf{S}_A$ and $\mathbf{S}_B$ intersect in the 3D projected image along the line segment $\mathbf{L}_{\text{se}}$, from point $\mathbf{P}_s = (x_s, y_s, z_s)$ to point $\mathbf{P}_e = (x_e, y_e, z_e)$. Suppose the pair of 4D points sharing $\mathbf{P}_s$ as shadow points are $\mathbf{P}_0$ on $\mathbf{S}_A$, and $\mathbf{Q}_0$ on $\mathbf{S}_B$; likewise, we have $\mathbf{P}_1$ on $\mathbf{S}_A$, and $\mathbf{Q}_1$ on $\mathbf{S}_B$ sharing $\mathbf{P}_e$ as shadow

point. Obviously, $\mathbf{P}_0$, $\mathbf{P}_1$, $\mathbf{Q}_0$, and $\mathbf{Q}_1$ can be represented as:

$$\begin{aligned} \mathbf{P}_0 &= (x_s, y_s, z_s, w_{sA}) \\ \mathbf{Q}_0 &= (x_s, y_s, z_s, w_{sB}) \\ \mathbf{P}_1 &= (x_e, y_e, z_e, w_{eA}) \\ \mathbf{Q}_1 &= (x_e, y_e, z_e, w_{eB}) \ . \end{aligned} \tag{5.1}$$

Now we can detect the closest approach of the intersection lines contained in each plane, and prevent actual 4D collisions from occurring before they happen based on keeping the closest approach greater than some suitable small number ε.

Closest Points between 4D Line Segments. We first consider two infinite lines $\mathbf{L}_1$: $\mathbf{P}(s) = \mathbf{P}_0 + s(\mathbf{P}_1 - \mathbf{P}_0) = \mathbf{P}_0 + s\mathbf{u}$ and $\mathbf{L}_2$: $\mathbf{Q}(t) = \mathbf{Q}_0 + t(\mathbf{Q}_1 - \mathbf{Q}_0) = \mathbf{Q}_0 + t\mathbf{v}$. Let $\mathbf{w}(s,t) = \mathbf{P}(s) - \mathbf{Q}(t)$ be a vector between points on the two lines. We want to find the $\mathbf{w}(s,t)$ that has a minimum length for all s and t. In any N-dimensional space, the two lines $\mathbf{L}_1$ and $\mathbf{L}_2$ are closest at unique points $\mathbf{P}(s_c)$ and $\mathbf{Q}(t_c)$ for which $\mathbf{w}(s_c, t_c)$ attains its minimum length. Also, if $\mathbf{L}_1$ and $\mathbf{L}_2$ are not parallel, then the line segment $\mathbf{P}(s_c) \leftrightarrow \mathbf{Q}(t_c)$ joining the closest points is uniquely perpendicular to both lines at the same time. No other segment between $\mathbf{L}_1$ and $\mathbf{L}_2$ has this property (see Figure 5.10(b)). That is, the vector $\mathbf{w}_c = \mathbf{w}(s_c, t_c)$ is uniquely perpendicular to the line direction vectors $\mathbf{u}$ and $\mathbf{v}$, and thus it satisfies the equations:

$$\begin{aligned} \mathbf{u} \cdot \mathbf{w}_c &= 0 \\ \mathbf{v} \cdot \mathbf{w}_c &= 0 \ . \end{aligned} \tag{5.2}$$

We can solve these two equations by substituting $\mathbf{w}_c = \mathbf{P}(s_c) - \mathbf{Q}(t_c) = \mathbf{w}_0 + s_c\mathbf{u} - t_c\mathbf{v}$, where $\mathbf{w}_0 = \mathbf{P}_0 - \mathbf{Q}_0$, into each part of Eq. (5.2) to get two simultaneous linear equations. Then, letting $a = \mathbf{u}\cdot\mathbf{u}$, $b = \mathbf{u}\cdot\mathbf{v}$, $c = \mathbf{v}\cdot\mathbf{v}$, $d = \mathbf{u}\cdot\mathbf{w}_0$, and $e = \mathbf{v}\cdot\mathbf{w}_0$, we solve for s_c and t_c as:

$$s_c = \frac{be - cd}{ac - b^2}, \quad t_c = \frac{ae - bd}{ac - b^2} \ . \tag{5.3}$$

Having solved for s_c and t_c, we have the points $\mathbf{P}(s_c)$ and $\mathbf{Q}(t_c)$ where the two lines $\mathbf{L}_1$ and $\mathbf{L}_2$ are closest. Then the distance between them is given by:

$$d(\mathbf{L}_1, \mathbf{L}_2) = \left| (\mathbf{P}_0 - \mathbf{Q}_0) + \frac{(be - cd)\mathbf{u} - (ae - bd)\mathbf{v}}{ac - b^2} \right| \ . \tag{5.4}$$

Now we represent a segment $\mathbf{S}_1$ (between endpoints $\mathbf{P}_0$ and $\mathbf{P}_1$) as the points on $\mathbf{L}_1 : \mathbf{P}(s) = \mathbf{P}_0 + s(\mathbf{P}_1 - \mathbf{P}_0) = \mathbf{P}_0 + s\mathbf{u}$ with $0 \le s \le 1$. Similarly, the segment $\mathbf{S}_2$ on $\mathbf{L}_2$ from $\mathbf{Q}_0$ to $\mathbf{Q}_1$ is given by the points $\mathbf{Q}(t)$ with $0 \le t \le 1$. The distance between segments $\mathbf{S}_1$ and $\mathbf{S}_2$ may not be the same as the distance between their extended lines $\mathbf{L}_1$ and $\mathbf{L}_2$. The first step in computing a distance involving segments is to get the closest points for the lines they lie on. So, we first compute s_c and t_c for $\mathbf{L}_1$ and $\mathbf{L}_2$, and if these are in the range of the involved segment, then they are also the closest points for them. But if they lie outside the range, then they are not and we have to determine new points that minimize $\mathbf{W}(s,t) = \mathbf{P}(s) - \mathbf{Q}(t)$ over the ranges of interest.

To do this, we first note that minimizing the length of $\mathbf{w}$ is the same as minimizing $|\mathbf{w}|^2 = \mathbf{w}\cdot\mathbf{w} = (\mathbf{w_0} + s\mathbf{u} - t\mathbf{v})\cdot(\mathbf{w_0} + s\mathbf{u} - t\mathbf{v})$ which is a quadratic function of s and t. In fact, this expression defines a paraboloid over the (s,t)-plane with a minimum at $C = (s_c, t_c)$, and which is strictly increasing

along rays in the (s,t)-plane that start from C and go in any direction. However, when segments are involved, we need the minimum over a subregion $\mathbf{G}$ of the (s,t)-plane, and the global minimum at C may lie outside of $\mathbf{G}$. An approach is given by [Eberly(2001)], suggesting that the minimum always occurs on the boundary of $\mathbf{G}$, and in particular, on the part of $\mathbf{G}$'s boundary that is visible to C. Thus by testing all candidate boundaries, we can compute the closest points between 4D line segments.

Closest Points between 4D Point and Triangle. The problem now is to compute the minimum distance between a point $\mathbf{P}$ and a triangle $\mathbf{T}(s,t) = \mathbf{B} + s\mathbf{E}_0 + t\mathbf{E}_1$ for $(s,t) \in D = (s,t) : s \in [0,1], t \in [0,1], s+t \leq 1$. The minimum distance is computed by locating the value $(\bar{s},\bar{t}) \in D$ corresponding to the point on the triangle closest to $\mathbf{P}$. Note that our algorithm should work also for a triangle and a point embedded in arbitrary dimensions.

The squared-distance function for any point on the triangle to $\mathbf{P}$ is $Q(s,t) = |\mathbf{T}(s,t) - \mathbf{P}|^2$ for $(s,t) \in D$. The function is quadratic in s and t,

$$Q(s,t) = as^2 + 2bst + ct^2 + 2ds + 2et + f, \tag{5.5}$$

where $a = \mathbf{E}_0 \cdot \mathbf{E}_0$, $b = \mathbf{E}_0 \cdot \mathbf{E}_1$, $c = \mathbf{E}_1 \cdot \mathbf{E}_1$, $d = \mathbf{E}_0 \cdot (\mathbf{B} - \mathbf{P})$, $e = \mathbf{E}_1 \cdot (\mathbf{B} - \mathbf{P})$, and $f = (\mathbf{B} - \mathbf{P}) \cdot (\mathbf{B} - \mathbf{P})$. Quadratics are classified by the sign of $ac - b^2$, which here becomes

$$ac - b^2 = (\mathbf{E}_0 \cdot \mathbf{E}_0)(\mathbf{E}_1 \cdot \mathbf{E}_1) - (\mathbf{E}_0 \cdot \mathbf{E}_1)^2 = |\mathbf{E}_0 \times \mathbf{E}_1|^2 > 0. \tag{5.6}$$

The positivity is based on the assumption that the two edges $\mathbf{E}_0$ and $\mathbf{E}_1$ of the triangle are linearly independent, so their cross product is a nonzero vector.

In calculus terms, the goal is to minimize $Q(s,t)$ over D. Since Q is a continuously differentiable function, the minimum occurs either at an interior point of D where the gradient $\nabla Q = 2(as+bt+d, bs+ct+e) = (0,0)$, or at a point on the boundary of D.

The Algorithm. The gradient of Q is zero only when $\bar{s} = (be-cd)/(ac-b^2)$ and $\bar{t} = (bd-ae)/(ac-b^2)$. If $(\bar{s},\bar{t}) \in D$, then we have found the minimum of Q. Otherwise, the minimum must occur on the boundary of the triangle. To find the correct boundary, consider Figure 5.11,

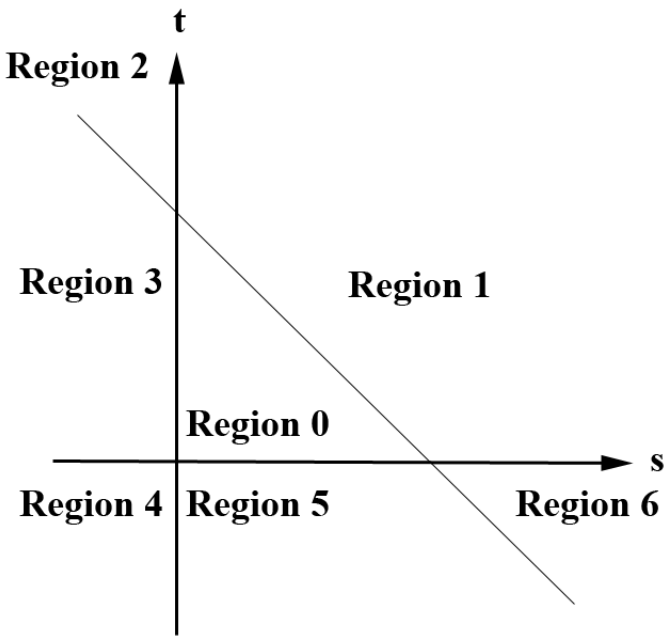


Figure 5.11: Partitioning of the st-plane by triangle domain D.

The central triangle labeled region 0 is the domain of $Q, (s,t) \in D$. If $(\bar{s},\bar{t})$ is in region 0, then the point on the triangle closest to **P** is interior to the triangle.

Suppose $(\mathbf{s},\mathbf{t})$ is in region 1. The level curves of Q are those curves in the st-plane for which Q is a constant. Since the graph of Q is a paraboloid, the level curves are ellipses. At the point where $\nabla Q = (0,0)$, the level curve degenerates to a single point $(\bar{s},\bar{t})$. The global minimum of Q occurs there, call it V_{min}. As the level values V increase from V_{min}, the corresponding ellipses are increasingly further away from $(\bar{s},\bar{t})$. There is a smallest level value V_0 for which the corresponding ellipses are increasingly further away from $(\bar{s},\bar{t})$. There is a smallest level value V_0 for which the corresponding ellipse (implicitly defined by $Q = V_0$ just touches the triangle domain edge $s+t=1$ at a value $s = s_0 \in [0,1]$, $t_0 = 1 - s_0$. For level values $V < V_0$, the corresponding ellipses do not intersect D. For level values $V > V_0$, portions of D lie inside the corresponding ellipses. In particular, any points of intersection of such an ellipse with the edge must have a level value $V > V_0$. Therefore, $Q(s,1-s) > Q(s_0,t_0)$ for $s \in [0,1]$ and $s \neq s_0$. The point (s_0,t_0) provides the minimum squared-distance between **P** and the triangle. The triangle point is an edge point. Figure 5.12 illustrates the idea by showing various level curves.

5.3 Lifting in Four Dimensions

Experiencing a physical sense of a 4D collision is the key to creating a bridge to four-dimensional reality. 3D physics-based simulation has had broad success in many modeling domains, and the work closest to ours involves chains (see, e.g., [Barzel and Barr(1988)]) and deformable objects (see, e.g., [Terzopoulos et al.(1987)Terzopoulos, Platt, Barr, and Fleischer], [Terzopoulos and Witkin(1988)]

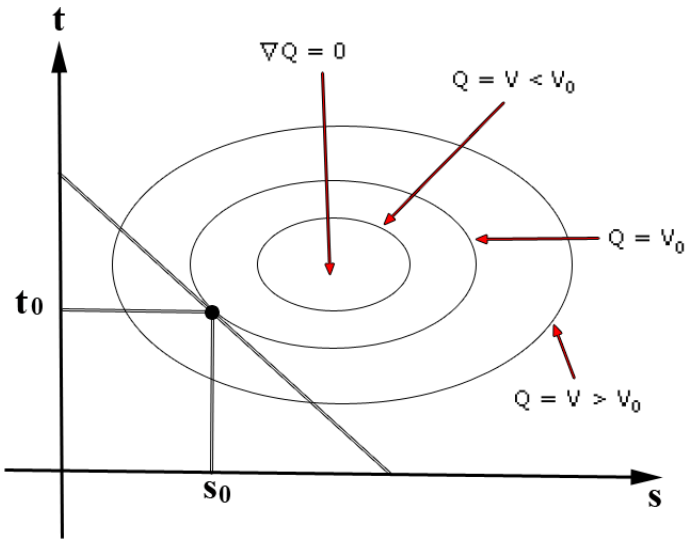


Figure 5.12: Various level curves $Q(s,t) = V$.

[Brown et al.(2004)Brown, Latombe, and Montgomery], and [Wejchert and Haumann(1991)]). Haptic interfaces have also been used to implement realistic 3D physics-based simulations (see [Baxter et al.(2001)Baxter, Scheib, and Lin] and [Kim et al.(2003)Kim, Sukhatme, and Desbrun]). By extending these approaches to 4D physics-based haptic simulation, we can establish intuitive interactions exposing the true nature of higher dimensions, even though our controls are restricted to the physical world, which is the 3D domain of 4D shadows.

5.3.1 Constrained Motions and Special Effects

Among the particular effects we have implemented, we note the following:

- Constrained motions: During the interaction, the haptic proxy attaches a control point to a user-specified location. The 4D control point moves in the 3D projected coordinates, leaving the hidden dimension unchanged.
- Sensing collisions: When a 4D collision is detected, friction forces are rendered to the haptic device enhance the sense of physical reality.
- Sensing gravity: The pull of gravity can be rendered to the haptic device to help provide a sense of lifting a real object in four dimensions.

5.3.2 4D Collision Detection between Rigid Objects

The general case of collision handling involves both self-collisions and collisions between distinct objects. A particular case of interest involves collisions between rigid objects, i.e., collisions of distinct mathematical objects embedded in 4D. Therefore, in this section we will only deal with collisions between objects represented by a set of triangles (embedded in 4D).

Now let t_0 be an instant when there is no interpenetration between the objects, which is typical for most of our initial embeddings. Consider a time interval $[t_0, t_0 + \Delta t]$. Knowing the positions and velocities of each node of our model at time t_0, it is possible to compute its position at time $t_0 + \Delta t$. Collision detection then consists of finding out if one or more collisions occurred during this interval.

Only two types of collisions need to be considered:

- either a node of one of the mesh went through a triangle of the other object ("point-triangle"

collision);

- or the edge of a triangle of one object went through an edge of the other object ("edge-edge" collision).

Note that the since we are studying collisions between rigid 4D objects, we don't need to take care of "point-triangle" collisions or "edge-edge" collisions within one object (as long as we are sure the initial embedding of each object is safe). This feature is very important for our collision detection method.

5.3.2.1 Collision Response

4D collisions between distinct 4D rigid objects are handled in a similar way to 3D cases. For example, since the classic 4D "chain" consists of linked spheres and circular ribbons, we can exploit the haptic interface to thread ribbons among spheres to create an arbitrary chain. Then we can apply an external force by pulling on the top ribbon via the haptic device, and thus lift the chain in four dimensions (see, e.g., Figure 5.13(c,d)). Just as in ordinary space, when the ribbon approaches collision with a linked sphere, we apply a force at the collision point, which lifts the sphere along with the ribbon against the external force of gravity, combined with damping so that the body of the sphere swings about the ribbon contact point in a familiar way.

5.3.2.2 Pair Reduction

In practice, interesting 4D surface models may consist of thousands of polygons, and manipulating these objects may require very costly searches to perform collision detection. A typical complexity-reduction strategy is to use a four-dimensional bounding box to eliminate pairs that have no possible 4D depth overlap. In addition, as noted earlier, although nearest approaches of 4D objects may not coincide in the 3D projection, any actual 4D collision must occur also in the 3D projection. This fact actually helps to reduce the collision detection problem to one lower dimension, and to accelerate the identification of starting points for the nearest-approach computation.

5.3.3 Example: Lifting a Chain in Four Dimensions

A classic 4D structure is a "chain" consisting of linked spheres and circular ribbons. The forces to be considered include:

- Forces applied to a body by the haptic probe. The force is constrained to the shadow domain in which it is applied, with the unseen projection-direction coordinate held fixed. Just as one can lift 2D shadow images of 3D rings with a control constrained to the 2D projection plane, we can lift 3D projected images of 4D object configurations in an analogous way.
- Force of gravity and damping. We allow objects to hang from one another realistically as they are lifted against the force of gravity by contact with one another.

The mechanism of 4D lifting of the chain can be implemented via the following cycles (see Figure 5.13):

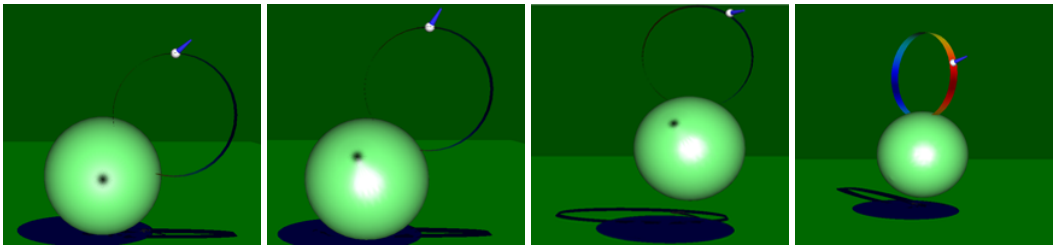
↪ Haptic proxy, attached to a ribbon at a user-specified location, moves in shadow space.

↪ Continue until a potential collision is detected between the ribbon and an adjacent sphere.

↪ Locate closest points, prepare forces for application to 4D objects.

↪ Activate contact forces and gravity to adjust positions of objects.

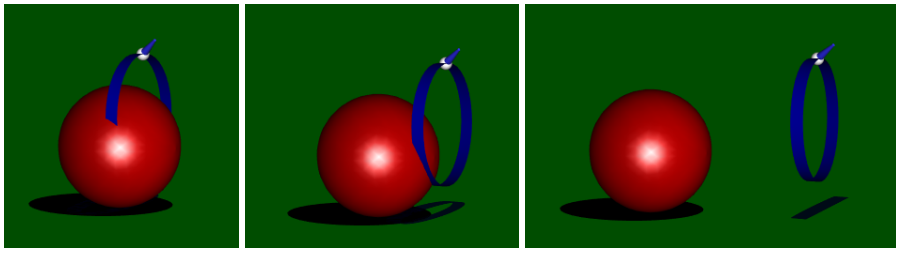
↪ Haptic proxy is free again to move the ribbon with constrained motions.



(a) (b) (c) (d)

Figure 5.13: The haptic interface for lifting a chain element in four dimensions, sensing collisions and weight.

In Figure 5.14, we show the result of a construction that produces two unlinked 4D embedded objects that superficially appear linked; we can disassociate the 3D projected images by pulling on the circular ribbon until the projections no longer conflict.



(a) (b) (c)

Figure 5.14: The haptic interface for physically disassociating the 3D projected images of two unlinked 4D embedded objects.

5.4 Conclusion

Going beyond the physics of three-dimensional space suggests many exciting applications such as those we have described in this section. We have extended many aspects of 3D physics simulation to 4D, including the construction of 4D knots and links. The 4D collision mechanism yields interesting perceptual insights into the 4D world, and can potentially make the 4D world as real as the 3D world by exploiting the sense of touch for colliding objects.

6

Editing 4D Cloth-like Objects

6.1 Motivation

The idea of cross-dimensional understanding has developed in many directions. Abbott's *Flatland* asked how two-dimensional creatures might attempt to understand three-dimensional space [Abbott(1952), Dewdney(1984)], while Banchoff's pioneering work suggested how 3D computer-based projections could be used to study 4D objects [Banchoff(1986), Banchoff(1990)]. Other representative efforts include a variety of ways to render 4D objects (see, e.g., Noll [Noll(1967)], Hollasch [Hollasch(1991)], Banks [Banks(1992)], Roseman [Roseman(1993)], and Egli, Petit, and Stewart [Egli et al.(1996)Egli, Petit, and Stewart]), and to extend lighting model techniques to 4D (see, e.g., [Carey et al.(1987)Carey, Burton, and Campbell, Steiner and Burton(1987), Hanson and Heng(1992)]). Our own previous efforts suggested how haptic exploration of 4D objects can exploit topological continuity by ignoring illusory 3D surface intersections and focusing on the intrinsic 4D geometry [Hanson and Zhang(2005)].

In parallel to the above-cited work studying 4D mathematical objects via 3D projected images, there

are also many approaches describing 3D curves and knots in terms of their shadows, largely motivated, one suspects, by the constraints of blackboard lectures and mouse-based computer interfaces. For example, sketching a knot with Scharein's Knotplot is done in a plane (basically the shadow world of the 3D embeddings) using a mouse-based control system [Scharein(1998)], while Cohen et al. create 3D curves by correlating the curve with its sketched shadow to compute the curve's 3D shape [Cohen et al.(1999)Cohen, L.Markosian, Zeleznik, Hughes, and R.Barzel].

Typical visualization methods for understanding higher dimensions employ a projection to 2D, or perhaps 3D, as a fundamental step; this helps the viewer to identify salient global features of the whole object and provides structural continuity when rotating the object's projection, or performing higher-dimensional rotations to change the projected image. Haptic exploration, being intrinsically limited to the physical world, also must project higher dimensional objects to three dimensions or less for interaction; within this context, knotted curves embedded in 3D can be projected to 2D and edited topologically within a consistent projected context to reveal complex topological relationships and structure. With the advent of modern interactive graphics technology we can begin to appreciate the challenge of sensing the phenomena from higher dimensions that baffled the two-dimensional shadow man of Fechner's story. In this chapter we begin to answer the question: "How can we use touchable shadows and perceivable shadow-space forces to manipulate all the degrees of freedom of a four-dimensional world?"

6.2 Physically Interacting with 3D Shadows of 4D surfaces

We now turn to our main objective, which is to create unique physical experience for the 4D world that can begin to make the strange more familiar.

The technique described in Chapter 3 for controlling 3D curves projected to 2D has an exact analog that applies to projection from 4D to 3D. 4D shapes can be created, edited, and explored in their 3D projections by using the full three degrees of freedom of the haptic probe instead of artificially restricting ourselves to a 2D subspace. We typically want to investigate 4D surfaces instead of curves, since surfaces in 4D play many roles analogous to those of curves in 3D; in particular, spheres are the analogs of closed curves, and knotted curves are replaced by knotted spheres in 4D. Pushing and pulling on the projection of a 4D surface in the 3D projection manipulates its shape in all but the shadow-ray direction. By using alternate states or modifiers, the controller can activate shadow-ray-aligned motion, and for models or sketch results with very small shadow-ray components, we can handle collisions by forcing the modeled shape component to go "above" or "below" its colliding neighbor in the ray direction. Rigid 4D rotation controls can in turn expose any component in the 3D projection for selective editing.

Just as we can edit projections of 3D curves, we can analogously sketch the 3D shadows of 4D surfaces. We depend on the 3D collision mechanisms to locate possible crossings, and use a modifier to make over and under choices in 4D. Figure 6.1 shows the mental model of a user working with a 4D flat torus, while Figure 5.2 shows the first step of the construction of a 4D "chain" consisting of linked spheres and circular ribbons.

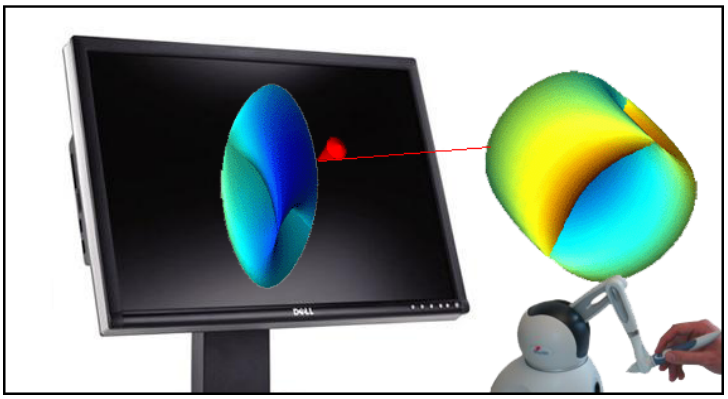


Figure 6.1: Editing the 3D shadow image of 4D object embedded in a simulated 4D mathematical world.

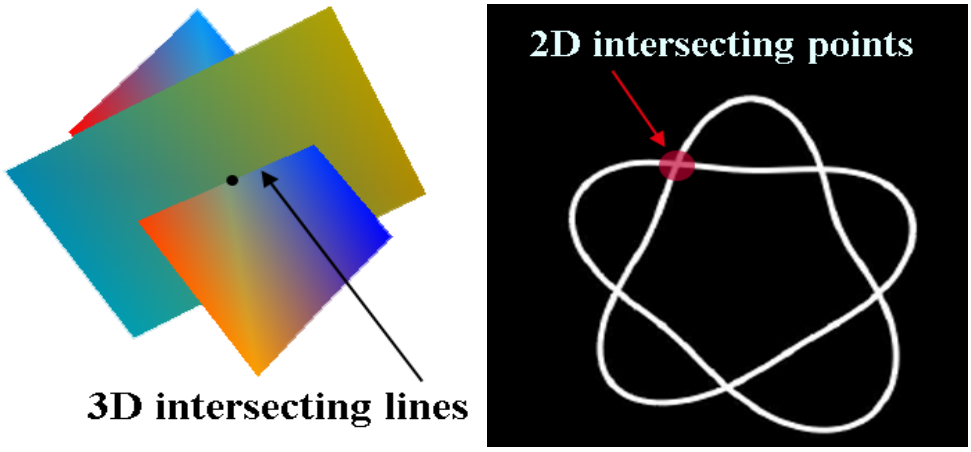
6.2.1 Collision Mechanism for Non-rigid 4D Surfaces

The haptic device can freely edit a 4D shape (typically a surface) in the 3D shadow space using our basic control philosophy augmented by 4D physical simulation. The key to editing topological surfaces embedded in four dimensions is a proper collision handling mechanism that realizes material forces and prevents collisions from occurring in 4D.

6.2.1.1 Collision in Four Dimensions

To understand the nonintuitive mechanisms of 4D collision, let us start with a pair of two-dimensional planes through the origin in four-dimensional space (see Figure 6.2(a)). The two squares intersect in a single 4D point at the origin. In the three-dimensional projection, however, the planes appear to intersect along an entire line due to the projective collapse of the w dimension. When the surfaces are 4D depth color-coded, we can see that there is just one pair of points with the same fourth coordinate along the intersection line in the 3D projection. *Note that 4D collisions*

take place only if there is also a 3D collision in the shadow, but that 3D shadow collisions may not imply 4D collisions.



(a) (b)

Figure 6.2: (a) 4D Collision: surfaces are color-coded to show distance from the 4D projection point, so we see that only one point on the intersection line of the two planes has the same 4D coordinate. (b) A pair of 3D points that share same 2D projected coordinates.

We use Figure 6.2(b) to illustrate the algorithm for projection based collision detection: although 2D projections of 3D curves intersect with each other at the same projected coordinate, a 3D depth test must be performed on the pair of 2D intersecting points to determine if they actually collide in 3D. Similarly, a 4D depth collision test must be performed along the 3D lines where the projected images of the 4D surfaces intersect. The closest points on the 4D line segment pair are identified to compute the distance between them [Zhang and Hanson(2006)]. *4D collision occurs if, and only if, one or more pairs of points are located with the same fourth coordinate along the intersecting line.*

6.2.1.2 4D Collision Avoidance

Just as for 2D or 3D collision handling mechanisms, there are two possibilities for handling 4D collisions: resolve them after they happen and are detected using methods employed in [Zhang and Hanson(2006)], or never let them occur. Since the haptic probe on a colliding point pair is physical, one cannot simply shift both positions to undo the collision; thus, we typically choose to prevent 4D collisions before they happen, i.e., with 4D collision avoidance.

Let t_0 be an instant when there is no interpenetration between the two polygons embedded in 4D. Consider a time interval $[t_0, t_0 + \Delta t]$. Knowing the positions and velocities of each node of the 4D surface model at time t_0, it is possible to compute its positions at time $t_0 + \Delta t$ (we predefine a threshold value δ for the thickness of the 4D surface and any motion between two frames is clamped to be no greater than δ). Collision avoidance then consists of determining if one is heading towards a potential collision. Two cases occur:

- a vertex of a triangle in a 4D mesh is going towards a triangle of the mesh and the distance is less than δ (point-triangle collision)
- the edge of a triangle of a 4D mesh is going towards another triangle and their distance is less than δ (edge-edge collision)

If either of these two states exists, the pair of closest points on the colliding components are identified and l is defined as the 4D vector passing through them. Then an equal (but opposite) displacement along l is applied to each component along l, where the displacement is just enough to take the component out of collision range. The basic 4D computations for point-triangle distance

and edge-edge collision are extended from the 3D cases [Zhang and Hanson(2006), Provot(1997)], and in principle can be extended and applied to 2D surfaces embedded in arbitrary N-dimensional space.

6.2.2 4D Cloth-Environment Simulation

Visualizing 4D topological surfaces has long been a subject of fascination; however, most of the previous work has focused on correctly rendering images of such surfaces in the projected space. Four-dimensional mathematical and physical behavior (such as 4D collisions) cannot exist at all except in the "mind" of the computer's mathematical imagination. We are thus led to investigate an enhanced visualization environment where we model 4D surfaces with a 4D mass-spring system supporting physical interaction within the "shadow-driven editing" method.

6.2.2.1 4D Mass-Spring System

There have been a number of cloth models proposed in the last decade. Most of them aim to reproduce the physical behavior of cloth [Provot(1997), Desbrun et al.(2000)Desbrun, Meyer, and Barr], and others are able to generate approximate, but plausible animations in the context of interactive virtual reality gaming applications (see, e.g., [J.Lander(1999)]). By extending the mass-spring system to four dimensions, we can in principle model the 4D mathematical and physical behavior of 4D topological surfaces. For the sake of generality, we will assume that each mass point i is linked to all the others with (linear) springs of rest length $l^0_{i,j}$ and stiffness $k_{i,j}$. This stiffness is set to zero if the actual model does not contain a spring between mass i and j. In the remainder of this chapter,

we will focus on 2-manifold deformable objects embedded in 4D. We will use the following three types of linear springs in our 4D interactive system:

- springs linking masses $[i, j]$ and $[i+1, j]$, and masses $[i, j]$ and $[i, j+1]$, will be referred to as "structural springs"
- springs linking masses $[i, j]$ and $[i+1, j+1]$, and masses $[i+1, j]$ and $[i, j+1]$, will be referred to as "shear springs"
- springs linking masses $[i, j]$ and $[i+2, j+1]$, and masses $[i, j]$ and $[i, j+2]$, will be referred to as "bending springs."

Figure 6.3 shows a cloth-like surface patch embedded in four dimensions. The 3D projection of the cloth has geometric information only in 3D $(x, y, z, 0)$, and could in principle be seen and touched in our everyday life. However, this manifold is *flattened* in the fourth dimension initially, i.e., each vertex has a 4D "eye coordinate" or depth $w = 0$; therefore it behaves initially just like a piece of 3D cloth (see Figure 6.3(b)) when exposed to 3D external forces.

To help the reader better understand this, we give a 3D analog in Figure 6.4: the point and the circle have non-zero geometric information only in 2D $(x, y, 0)$, and the point will be trapped inside the ring when exposed to 2D external forces only.

Now the question is "How can we take the point out of the ring if we are only able to apply 2D forces?" Starting with the 3D example in Figure 6.3(b), we uncover a simple rule to achieve this: if we apply a general 3D rotation to the ring and point, we will be able to move the point out of the 2D ring by applying a simple 2D translation on the point (see Figure 6.3(a)). This can be trivially

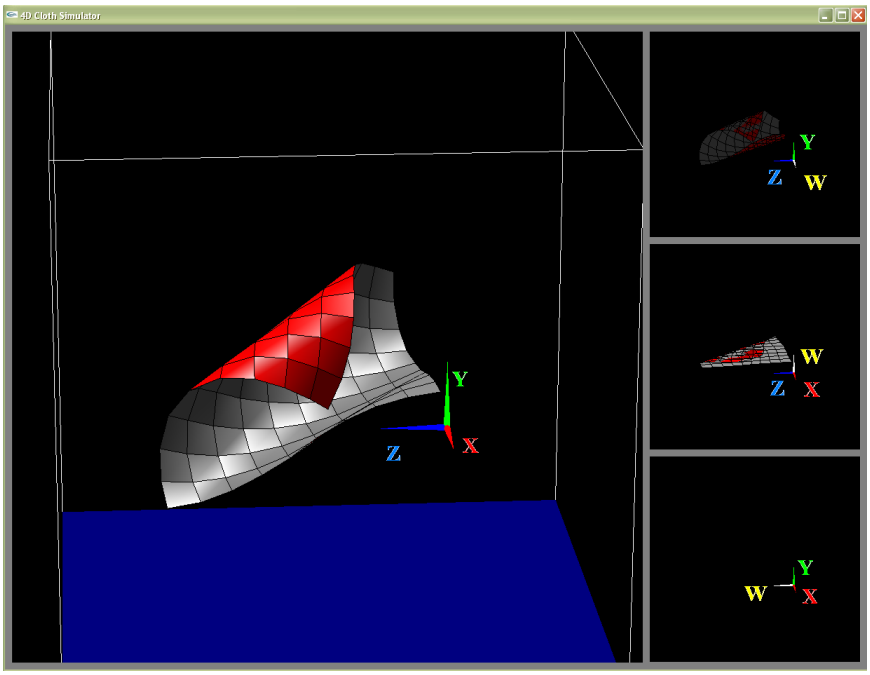
extended to our 4D example, if we apply a general 4D rotation to the cloth, each vertex will acquire a non-vanishing 4D "eye coordinate" or depth w; if we again apply 3D forces to this piece of the 4D-embedded cloth, we see that the cloth's apparent intersections with itself in the 3D projection are an illusion, and it actually does not have self-intersections in four dimensions (see Figure 6.5(b)).

6.2.2.2 Choosing the Shadow Plane

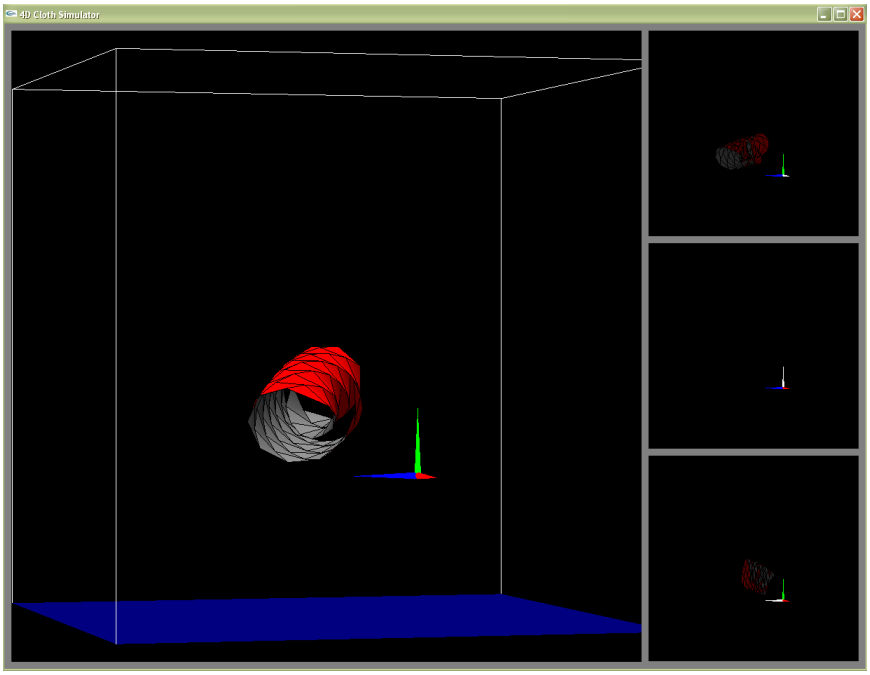
The orientation of the projection plane is an important factor in the interface. When sketching higher-dimensional objects in the projection, users typically will choose an informative projection to work on, much as we draw a knot diagram with pen and paper (see, e.g., Figure 3.11 and Figure 3.12). In the editing interface, a separate mouse-driven orientation control is provided to adjust the 4D rotation matrix whose first two columns define the projection plane; this allows sufficient control to achieve a good projection for each particular editing subtask (see Figure 6.7).

6.2.2.3 Forces

The internal force $F_{int}[i,j]$ is the resultant of the tensions of the springs linking mass $[i,j]$ to its neighbors, where the calculation is based on Eq. 3.1 in four dimensions. The external force $F_{ext}[i,j]$ is applied when the mass $[i,j]$ is exposed to haptic manipulation. We note that our mass-spring system is configured in 4D, so the internal forces exerted by 4D springs are 4D vectors. The external forces applied by the 3D haptic device are still 3D vectors, and the 3D shadow space is the human-accessible dimension when interacting with 4D objects (this is the heart and soul of our "shadow editing" method). All these considerations allow us to compute the force $F_{i,j}(t)$ applied to the mass

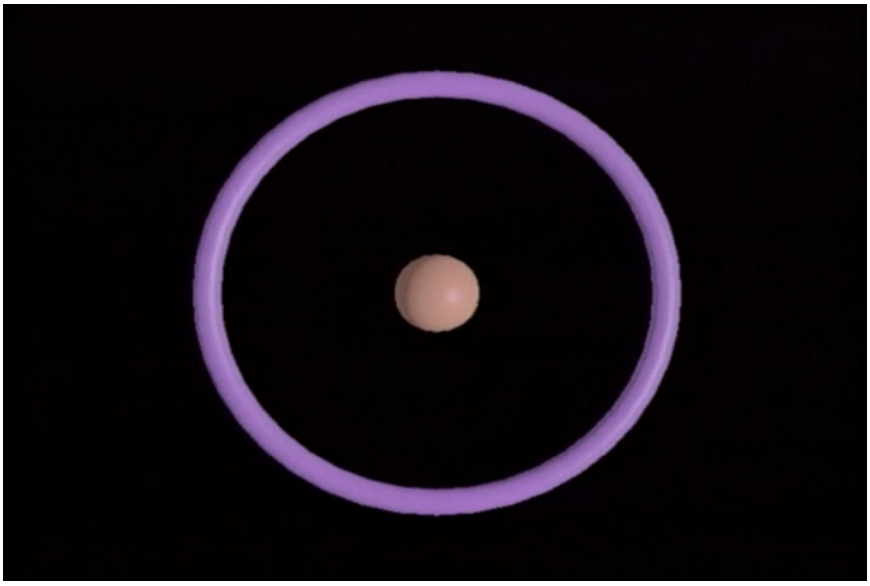


(a)

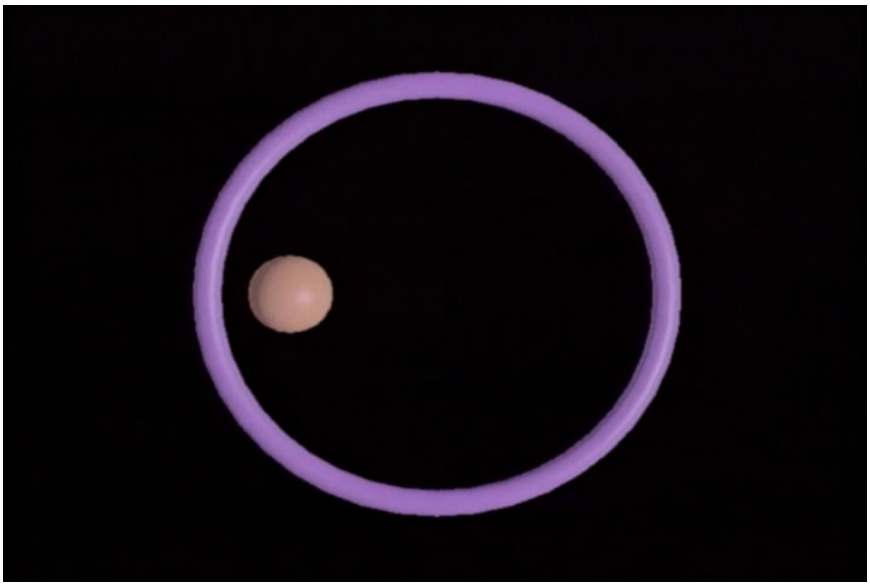


(b)

Figure 6.3: Flattened 4D cloth behaves like 3D cloth.

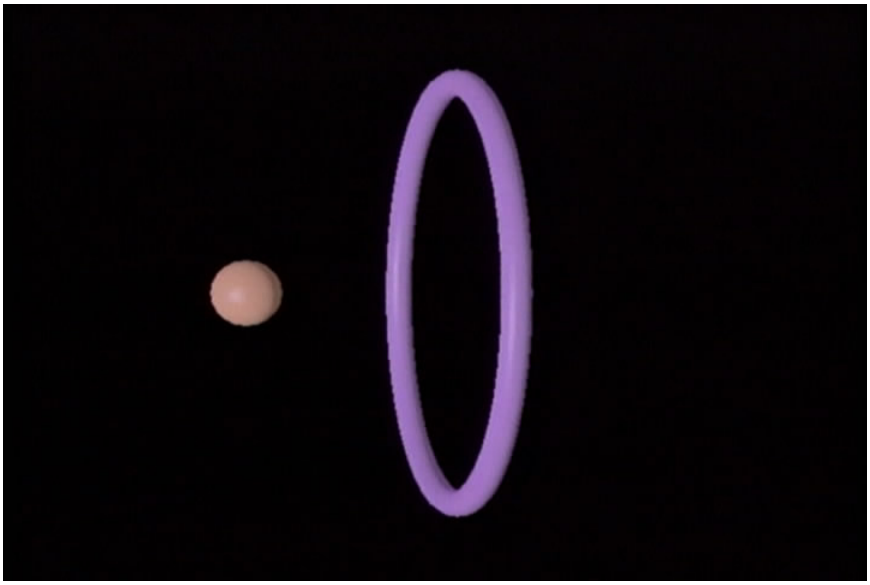


(a)

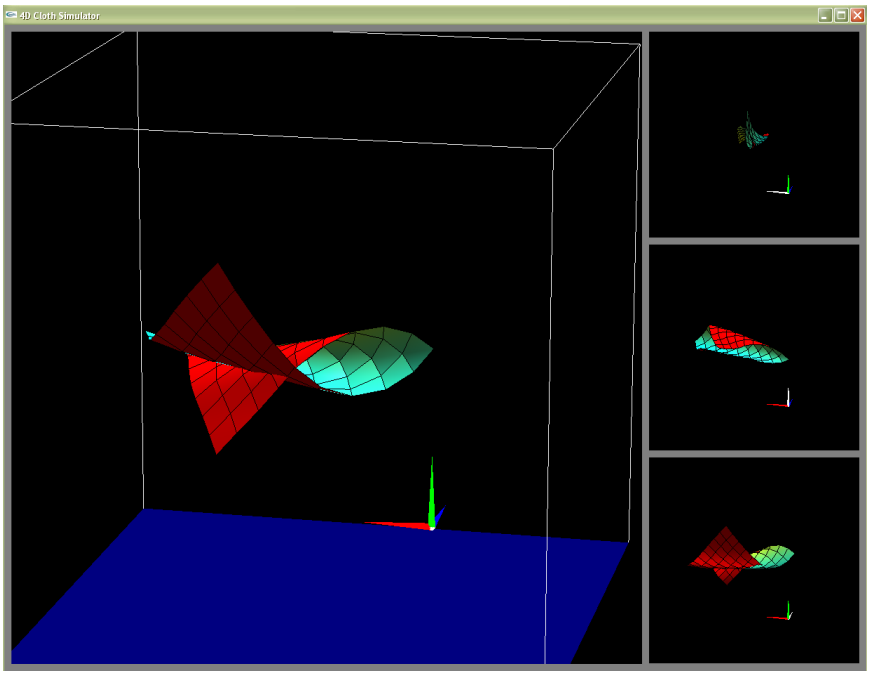


(b)

Figure 6.4: A point embedded in 3D is trapped inside a 2D ring when exposed to 2D external forces only.



(a)



(b)

Figure 6.5: (a) A point can escape from the surrounding 2D ring after 3D rotation. (b) Non-intersecting 4D cloth may appear to intersect itself in the 3D projection.

$[i, j]$ at any time.

6.2.2.4 Integration

The fundamental dynamical equation can be explicitly integrated across time by the Euler method:

$$\begin{cases} \alpha_{i,j}(t+\Delta t) = \frac{1}{m_{i,j}} F_{i,j}(t) \\ V_{i,j}(t+\Delta t) = V_{i,j}(t) + \Delta t\, \alpha_{i,j}(t+\Delta t) \\ P_{i,j}(t+\Delta t) = P_{i,j}(t) + \Delta t V_{i,j}(t+\Delta t) \end{cases} \tag{6.1}$$

6.2.2.5 Haptic 4D Cloth Rendering

When we combine haptics with 4D cloth rendering, we support interaction with a 4D cloth simulation in the 3D shadow space, incorporating projected 4D collision forces. Selected 3D haptic cloth rendering methods [Raymaekers et al.(2005)Raymaekers, Vanacken, Cuppens, and Coninx] are adapted to our 4D environment. Our system uses constraint-based methods [Zilles and Salisbury(1995)], haptically rendering a "rubber band line" from the haptic pointer to the picked vertex. Users can thus develop a correct interactive experience with the intuitive nature of unfamiliar 4D geometry by editing parts of the object in assorted 3D projections using 3D force feedback.

6.2.3 Examples

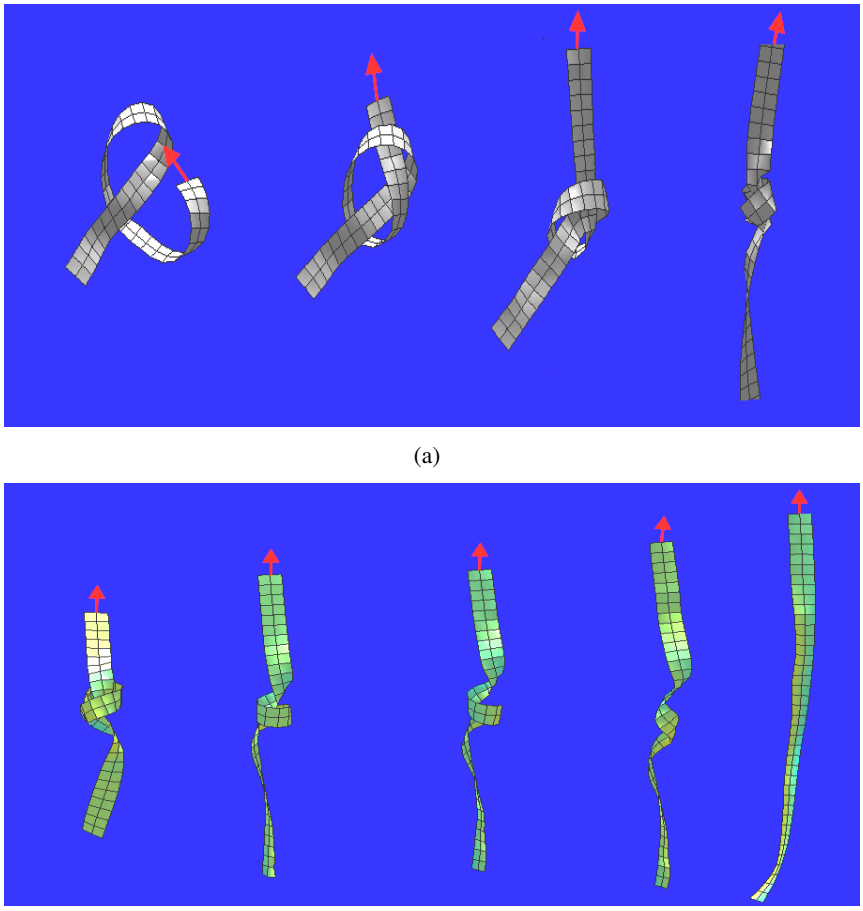
Interacting with 4D cloth-like objects using "shadow editing" methods can help users to develop a correct interactive experience with the intuitive nature of unfamiliar 4D geometry.

6.2.3.1 Visualizing 4D Collisions.

One interesting 4D mathematical phenomenon is that in four dimensions loops and strings can always be untied although they appear knotted in the projected space (this can be confirmed by applying a rigid 4D rotation). With our accurate 4D mathematical and physical modeling and "shadow editing" method, users can interactively untie a 4D string that appears knotted (see Figure 6.6).

6.2.3.2 Tightening the 4D Spun Trefoil

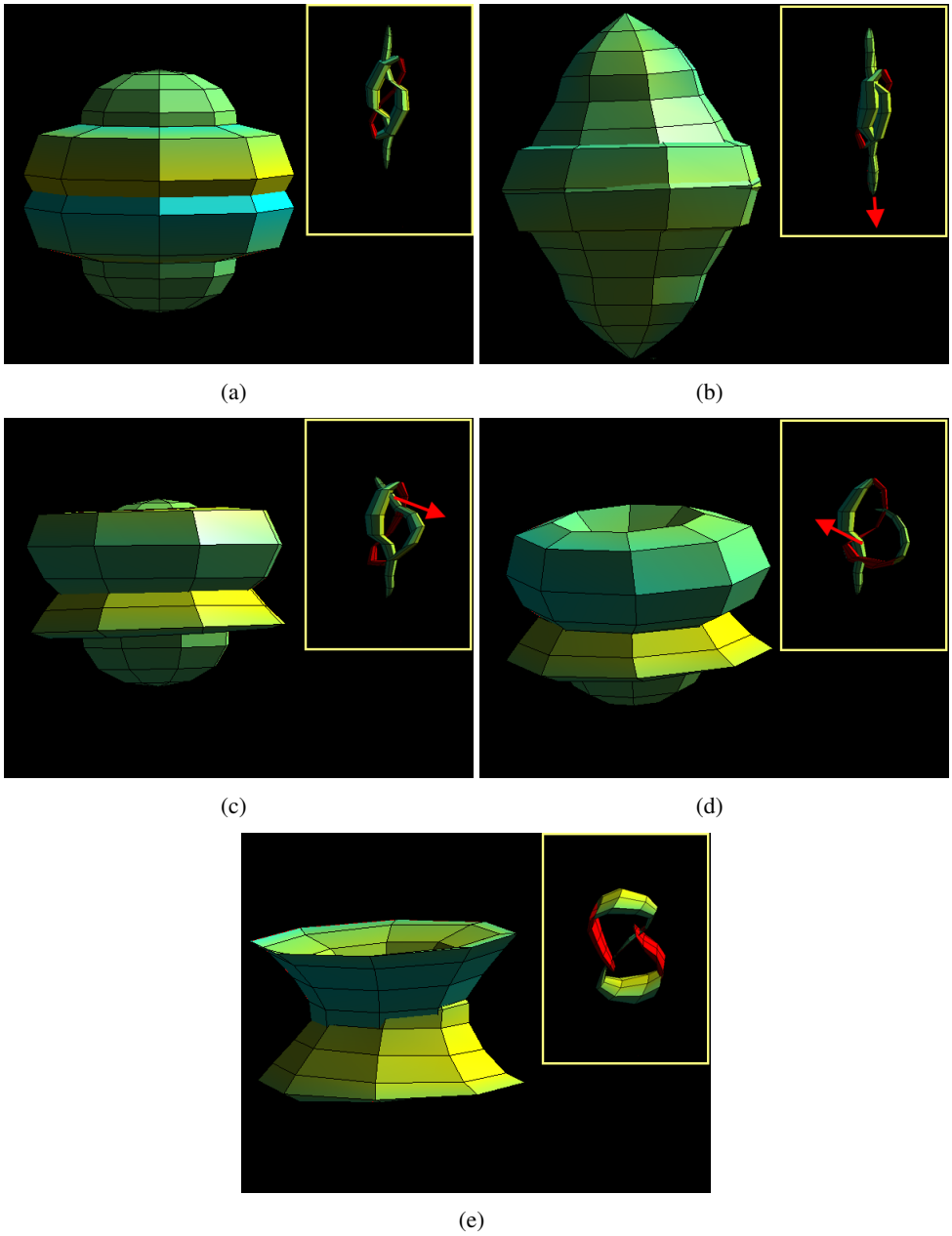
In four dimensions, even though loops and strings can always be untied, surfaces can be knotted. One family of 4D knots consists of knotted spheres that are formed by spinning a knotted line segment in the fourth dimension to sweep out the surface; an example is the 4D spun trefoil knot. This 4D knotted sphere, just like a truly knotted 3D trefoil knot, cannot be untied. As illustrated in Figure 6.7(a-b), the user can try to pull apart the 4D knotted sphere (the 4D analog of a knotted piece of string). The effort fails, since in the rigorous computer simulation of the 4D physical world, the object is a true knot and cannot be untied by pulling on it. Neither can it be loosened, as illustrated in Figure 6.7(c-e). In this way, users can develop a correct interactive experience with the intuitive nature of 4D space.



(a)

(b)

Figure 6.6: (a) A knot tied in a 3D ribbon cannot be undone. (b) An apparent knot in the 3D projection of a 4D ribbon falls apart.



(a) (b)

(c) (d)

(e)

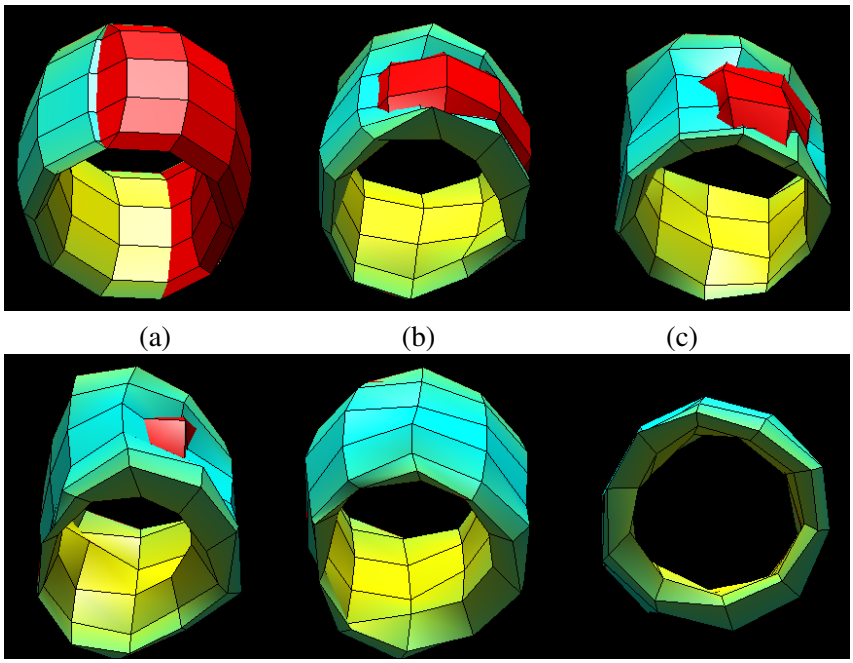
Figure 6.7: (a,b) Tightening a 4D knotted sphere by pulling on the 3D shadow. (c,d,e) Pushing on the knotted sphere to loosen it.

6.2.3.3 Deforming the 4D Torus

The 4D embedded torus (the product of two circles, technically written as T^2) is an object of fundamental interest, with a standard model given by $\mathbf{x}(u,v) = (\cos u, \sin u, \cos v, \sin v)$. Although the orthogonal 3D graphics projection of this torus has two lines of self-intersection, the true surface is actually a smooth topological manifold in four dimensions. Earlier, we presented a multimodal paradigm for exploring the smooth intrinsic features of topological surfaces like this embedded in four dimensions (see [Hanson and Zhang(2005)]). We can now obtain an alternative mental model of the structure of this surface by interactively deforming the object in the four dimensions. Figure 6.8 shows how to use local deformation to manipulate the projected image of the torus to actually eliminate the self-intersections in the 3D projection without altering or damaging the 4D surface in any significant way. Essentially, the pinch-points, where the X-shaped fold-over occurs, can be moved towards one another until they cancel both on the near and on the far side. The figure is color-coded for 4D depth, so the slight red bulge that remains in the middle is at a completely different depth from the blue surface; thus we can just push the red bulge through the blue surface in the 3D projection, yielding the final toroidal shape.

6.2.3.4 Physical Lifting in Four Dimensions

In Figure 6.9, we illustrate how materials interact in 4D by threading a ribbon into a piece of cloth, first going "above" in 4D, pulling under, and coming out the other side "below." If the cloth and ribbon are slippery, gravity will cause the cloth square to slide off the ribbon, much like two pieces of spaghetti in our 3D experience. In Figure 6.10, we perform the same operation except on a



(a) (b) (c)

(d) (e) (f)

Figure 6.8: (a)-(f): Selectively applying local deformations to the 4D embedding of the torus helps expose the underlying topological structure of the surface.

closed 2-sphere embedded in 4D; as long as we keep the two ends of the ribbon above the sphere, the captured point can no longer escape, and the sphere is lifted without being able to slide off; this is analogous to threading a string into a 3D ring. Just as we can link together rings in 3D to form a chain, we can in fact form 4D chains of alternating ribbons and spheres.

6.2.4 Multimodal Exploration of the Fourth Dimension

Haptic exploration of topological surfaces embedded in four dimensions can be supported by constraining the probe to conform to the local 4D continuity of the objects being examined [Hanson and Zhang(2005)]. The multimodal exploration in the 3D shadow image of the 4D embedded object passes continuously through the visually disruptive self-intersections of the 3D projection, revealing the full richness of the complex spatial relationships of the target shape.

6.3 Implementation Environment and Preliminary User Studies

Our implementation uses a SensAble Technology Omni PHANToM force-feedback haptic device combined with a high-performance graphics card supporting OpenGL. Our user interface is based on OpenGL and SensAble's OpenHaptics API. The software runs on a Dell PC desktop with 3.2GHz Intel Pentium 4 CPU. The haptic frame rate remains above 1000Hz for most tasks we have encountered (the haptic device requires a refresh rate of about 1000Hz in order to give the kinesthetic sense of stiff contact).

The 2D/3D system matches the shadow plane with the 2D computer screen, and a 2D cursor represents the haptic proxy in the scene. Haptic functions such as sketching, editing, and exploration of

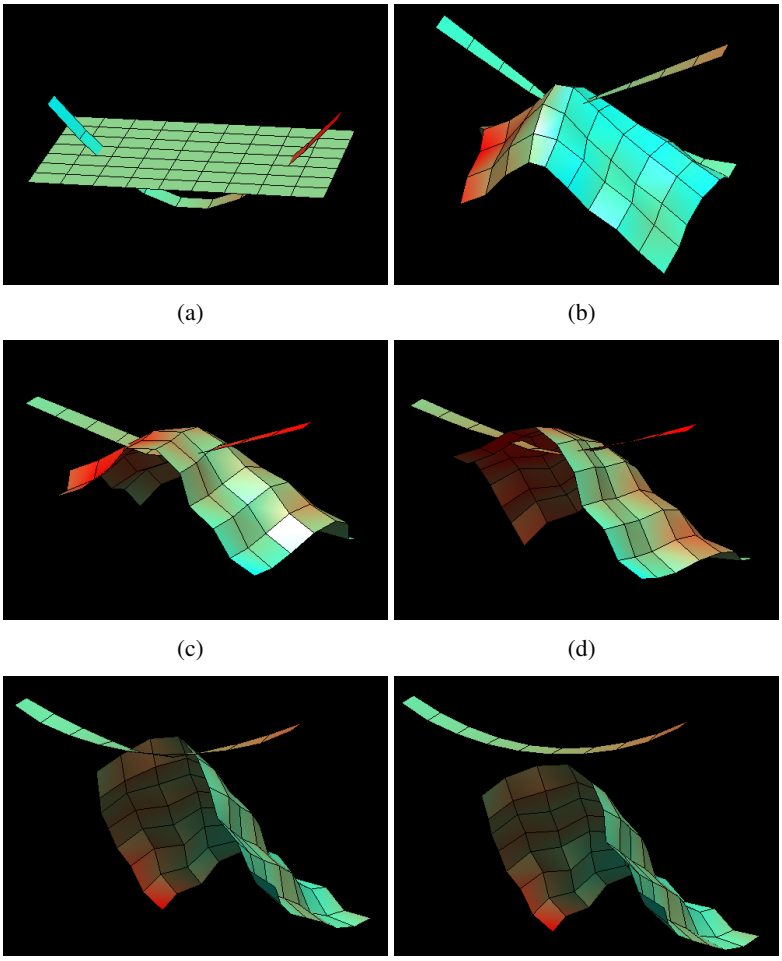
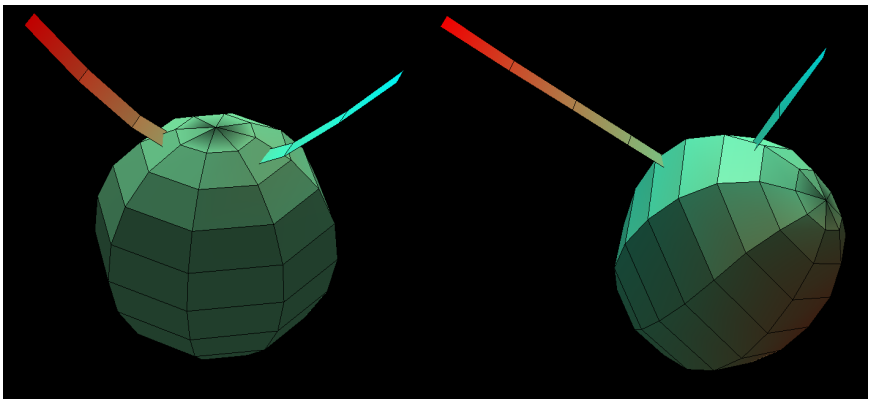


Figure 6.9: (a) Threading a 4D ribbon into a piece of 4D cloth (blue is under, red is over). (b,c) The 4D materials collide and exert lifting forces. (d,e,f) The 4D cloth slides off the ribbon due to gravity.



(a) (b)

Figure 6.10: (a) Threading a piece of 4D ribbon into a closed 4D-embedded 2-sphere. (b) When lifted against gravity, the 4D linking cannot fall off.

3D curves projected to the 2D plane are enhanced by real-time force-feedback. The haptic control of 2D projected images of 3D knots is smooth and natural, and the user senses the collisions in the higher dimension when taking control of the shadow figures.

Eight students from a topology class used the multimodal exploration tool for 3D mathematical knots before they were exposed to knot theory in the classroom. Haptic navigation on both 2D knot diagrams (see Figure 3.6(b)) and 3D rendered images (see Figure 3.6(c)) were supported to assist the participants' understanding of the relation between 2D knot diagrams and the actual 3D mathematical knot being represented. A set of 10 questions was used in our questionnaire. These questions focus on different aspects such as participants' acceptance of the "shadow-driven" exploration tool, haptic methods for aiding understanding of 3D knot structure, and auditory cues as supplementary information (measured using a 5-point Likert scale). Tasks such as determining whether two given knot diagrams represent the same knot and how to convert a true knot into an

	Show Driven Exploration Tool	Haptic Method	Auditory Cues
Evaluation (1 to 5)	M=4.0 SD=1.44	M=3.9 SD=1.25	M=3.7% SD=1.03%

Table 6.1: Participants' evaluation of knot exploration system

unknot by changing an individual crossing from "over" to "under" are also included in the questionnaire. Seven students (87.5%) were able to finish all the tasks in the questionnaire. Their response to the system was quite positive, as shown in Table 6.1 (scale 1 corresponds to "strongly disagree," 5 corresponds to "strongly agree"). Suggestions for improving the user interface were solicited; for example, students suggested constructing the Gauss code of the knot being explored in parallel with the crossing-triggered sound cues. We plan to incorporate some of this useful feedback into our 2D/3D and 3D/4D interfaces.

6.4 Summary

We have discussed a family of haptic methods for intuitively controlling the representations of higher-dimensional phenomena in their shadow spaces. By adding force-feedback to the concept of reduced-dimension object manipulation controllers, we have created a unique way of building intuition about 3D knots and curves in a 2D touchable space; this approach is naturally extensible to a full 3D haptic controller that provides touchable intuition and force feedback through simulated representations of shapes in a 4D world. Starting from this basic framework, we can attack families of significant problems in 4D intuitive visualization such as the interactive manipulation of apparently knotted, but actually unknotted, spheres in 4D. Other possible future work could extend the range of objects for which we can support 4D physical modeling to include more complex knots,

links, and Riemann surfaces, as well possibly supporting three-manifolds in addition to curves and surfaces.

7

Example: A Framework for Physically-Based Sphere Eversion

7.1 Introduction

We have now described our design and implementation of a simulation framework that can support physical simulation of colliding surfaces embedded in 4D that self-intersect without interference in the 3D projection; this framework can exploit either a normal desktop mouse interface or a 3D haptic interface for manipulation and control. There is an interesting set of problems in 3D geometry for which this particular framework is remarkably well-suited: the deformations of 3D surfaces in 3D for which non-interfering self-intersection is allowed. One of the most interesting of these, and one that has been widely studied, is the problem of smoothly turning a sphere inside out, which cannot be done without permitting self-intersections in 3D.

The problem of turning a sphere inside out without tearing or creasing, known in mathematics as a regular homotopy that *everts the sphere*, is a classic puzzle that has been solved in many ways since

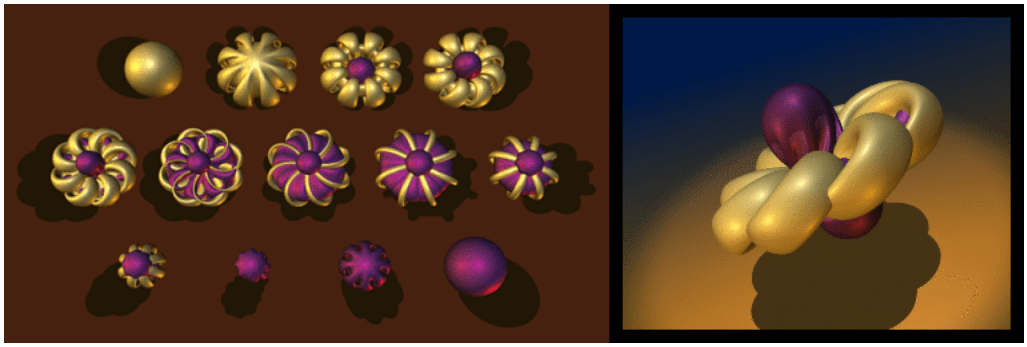


Figure 7.1: Thurston's method for everting the sphere. ©The Geometry Center

Smale first proved it must be possible [Smale(1958)]. For our purposes, we may simply assume that it is possible to evert the sphere, and seek a physical model that allows the eversion to be achieved in a natural and physically intuitive manner.

Figure 7.1 shows an example of a particular sphere eversion solution due to Thurston, implemented in computer graphics at the Geometry Center, and discussed in detail in the film *Outside In* [Center()]. Curiously, as explained in the film, an ordinary circle (S^1 cannot be everted in the Euclidean plane, while the ordinary sphere (S^2) can be everted in 3D Euclidean space. In Appendix B, we provide a summary of the mathematical background.

7.2 Circles Cannot Be Everted

To "turn a unit circle inside-out" would be to construct a regular homotopy between the standard embedding of the circle in the plane, $A(\theta) = (\cos\theta, \sin\theta)$, and its reflected parametrization, $B(\theta) = (-\cos\theta, -\sin\theta)$. B has its normal and velocity vectors pointing in the opposite direction of A's. Even though both A and B are regular curves, the circle cannot be everted. Intuitively, it seems

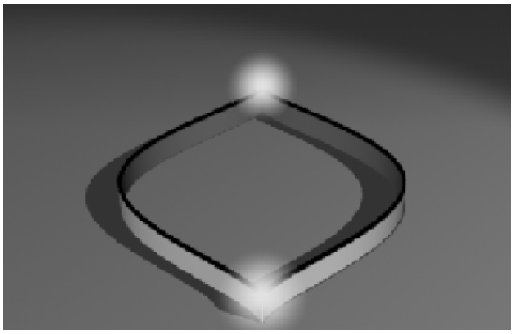


Figure 7.2: When we pull the two poles of the circle through one another, we get cusps. ©The Geometry Center

simple enough to pull two halves of the circle through one another, but this results in cusps where the derivative becomes singular.

See Figure 7.2 for an illustration of cusps. To understand why the circle cannot be everted, we introduce the concept of turning number or winding number. "The winding number of a curve is the total number of counterclockwise turns the curve makes." It has been shown that [Phillips(1966), Gardberg(1996)]:

Theorem 7.2.1. *The turning number of a regular curve does not change in the course of a regular homotopy, so two curves that are regularly homotopic must have the same turning number.*

The converse, due to Whitney [Whitney(1937)], states that:

Theorem 7.2.2. *Any two regular curves with the same winding number are regularly homotopic.*

These insights let us understand why the circle cannot be everted: the standard and inverted maps have different winding numbers. A, which winds counter-clockwise, has turning number 1. B, though, winds clockwise, so we say that its turning number is -1. Looking at Figure 7.3, note that an alternative way to consider calculating turning number is to think of a monorail traversing a curve (with walls raised and colored for clarity) that is red on one side and green on the other, with

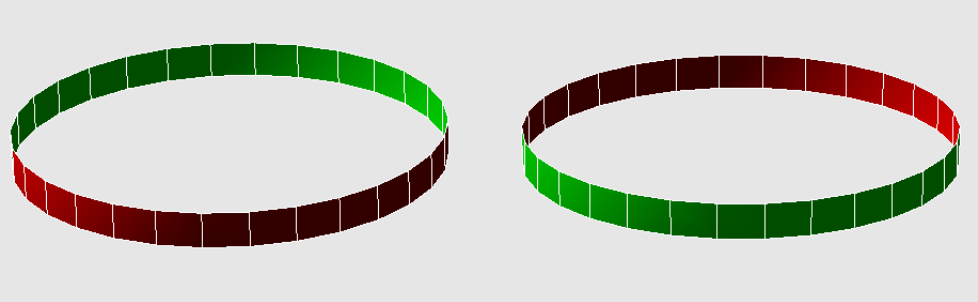


Figure 7.3: Standard and inverted circles, with sides raised for clarity.

red always on the car's right as it moves ever forward. If it were to go around the left-hand circle once, that would be one counter-clockwise rotation. Now, instead of actually pulling this monorail car around the entire curve, look instead at places where the car is traveling in a certain direction (that is, where the unit velocity vector takes a certain value). For the movie *Outside In*, created at The Geometry Center, the velocity $(1,0)$ was selected. This point will in general correspond to either a left, counterclockwise turn (a "smile"), or to a right, clockwise turn (a "frown"); subtracting the number of right turns from the number of left turns with this velocity value gives the turning number.

7.2.1 Physically Modeling the 2D Circle

In this section, we describe the families of models used to implement the interaction with a 2D circle. We begin by showing the motion of a manipulated 2D circle passing through itself in 2D space, without tearing or creasing. The circle cannot be turned inside out in 2D space, as illustrated, e.g., in the movie *Outside In*. Our task here is to show how we can fully exploit interactive graphics methods to resolve the possible confusion regarding singularities and self-intersection, and transform this complex task to a new level of interactive physical reality.

7.2.1.1 Raising the Sides of a 2D Circle

We begin by raising the sides of the 2D circle so that we can distinguish a standard circle from an inverted one. Let $A_+ = (\cos\theta, \sin\theta, +\kappa)$ and $A_- = (\cos\theta, \sin\theta, -\kappa)$. By constructing the surface A connecting A_+ and A_-, we can visualize a 2D circle with raised sides. Still better, we can distinctively color the inside and outside of the extruded 2D circle (see, e.g., Figure 7.3).

7.2.1.2 The Mesh

Visualizing topological surfaces using physical models is a long-standing problem. There have been a number of fabric models proposed, most with the goal of reproducing the physical behavior of cloth (see, e.g., Provot or Desbrun et al. [Provot(1997), Desbrun et al.(2000)Desbrun, Meyer, and Barr], or the approximate but plausible animations supporting interactive game virtual reality applications by Lander [J.Lander(1999)]).

To study the sphere eversion modeling problem, we are led to investigate an enhanced visualization environment where we model cloth-like surfaces with a 3D mass-spring system supporting physical interaction using self-intersection and smooth motions (i.e., no pinch-points). By adapting the mass-spring system to our eversion case, we can in principle model the eversion by exploiting the physical behavior of the intersection-permitting topological surface. For the sake of generality, we will assume that each mass point i is linked to all the others (j) with (linear) springs of rest length $l^0_{i,j}$ and stiffness $k_{i,j}$. This stiffness is set to zero if the actual model does not contain a spring between masses i and j. We will use the following three types of linear springs in our interactive system (see, e.g., Figure 7.4):

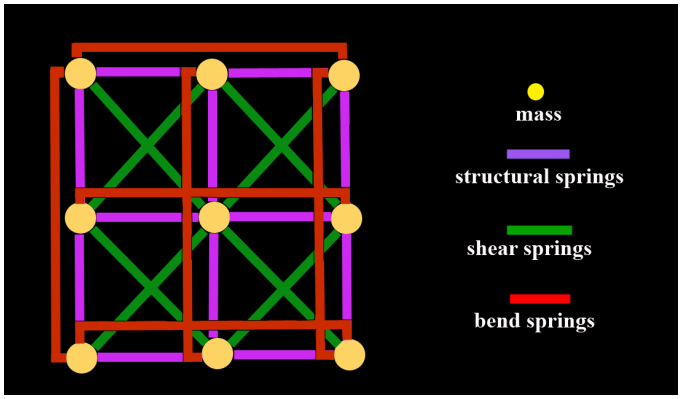


Figure 7.4: The interconnections of cloth springs.

- Springs linking mass $[i, j]$ to mass $[i+1, j]$ and $[i, j]$ to $[i, j+1]$ will be referred to as "structural springs."
- Springs linking masses $[i, j]$ and $[i+1, j+1]$ and linking masses $[i+1, j]$ and $[i, j+1]$ will be referred to as "shear springs."
- Springs linking masses $[i, j]$ and $[i+2, j+1]$ and liking masses $[i, j]$ and $[i, j+2]$ will be referred to as "bending springs."

7.2.2 Physically Interacting with a 2D Circle

7.2.2.1 Dynamics and Forces

The system under study is a mesh of $m \times n$ masses, each mass being positioned at time t at the point $P_{i,j}(t)$, where $i = 1,...,m$ and $j = 1,...,n$. The evolution of the system is governed by the fundamental law of dynamics:

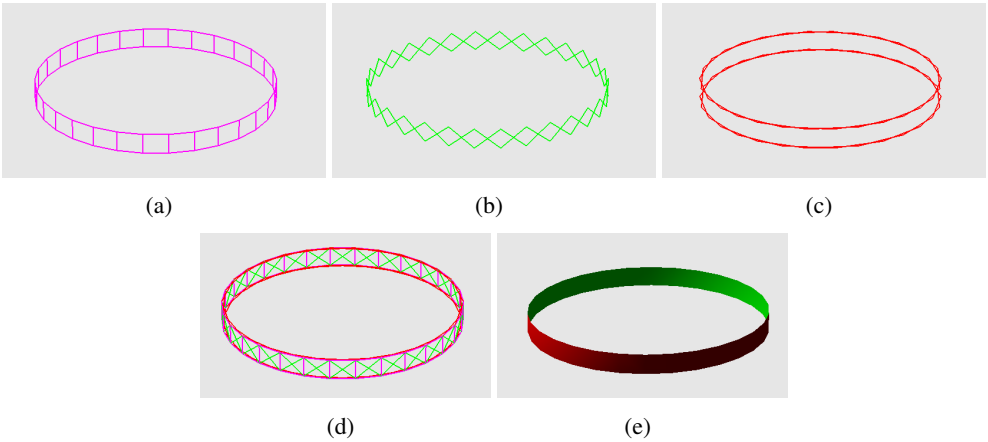


Figure 7.5: The interconnection of three types of springs for a 2D circle, with sides raised for clarity. (a) Structural springs. (b) Shear springs. (c) Bending springs. (d) The interconnection of all three types of springs. (e) Smooth representation of a 2D circle.

$$\mathbf{F_{i,j}} = \mu \alpha_{\mathbf{i,j}} \tag{7.1}$$

where μ is the mass of each point $\mathbf{P_{i,j}}$ and $\alpha_{\mathbf{i,j}} = \mathbf{d^2 P_{i,j}(t)/dt^2}$ is the acceleration caused by the force $\mathbf{F_{i,j}}$. $\mathbf{F_{i,j}}$ can be divided between the internal and external forces. The internal force is the resultant of the tensions of the springs linking $\mathbf{P_{i,j}}$ to its neighbors:

$$\mathbf{F}_{int}(P_{i,j}) = -\sum_{(k,l)\in\Re} K_{i,j,k,l}[l_{i,j,k,l} - l^0_{i,j,k,l}\frac{l_{i,j,k,l}}{\|l_{i,j,k,l}\|}] \,, \tag{7.2}$$

where:

- $\Re$ is the set of all labels (k,l) such that the point $P_{k,l}$ is linked by a spring to $P_{i,j}$,
- $l_{i,j,k,l} = \overrightarrow{P_{i,j}P_{k,l}}$,
- $l^0_{i,j,k,l}$ is the natural length of the spring linking $P_{i,j}$ and $P_{k,l}$,

- $K_{i,j,k,l}$ is the stiffness of the spring linking $P_{i,j}$ and $P_{k,l}$.

The external force varies according to the load to which the model is exposed. The viscous damping is given by:

$$\mathbf{F}_{dis}(P_{i,j}) = -C_{dis}\mathbf{v_{i,j}} \ , \tag{7.3}$$

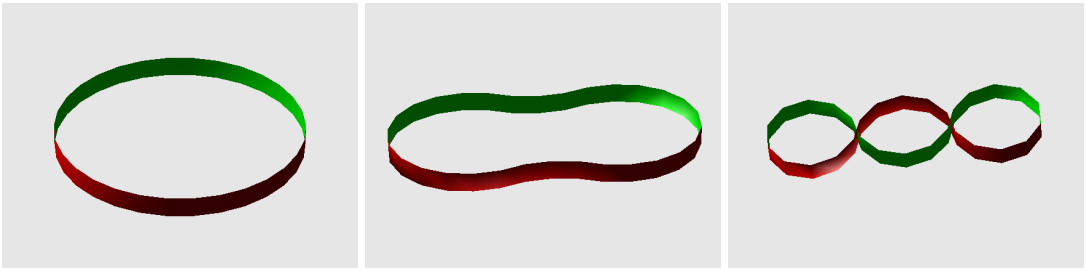
where C_{dis} is a damping coefficient, and $v_{i,j}$ is the velocity of point $P_{i,j}$. The role of this damping is in fact to provide a first order approximation to a model for the dissipation effects of the mechanical energy. It is introduced as an external force, but could actually be considered as an internal force as well.

7.2.2.2 Integration

All these considerations allow us to compute the force $F_{i,j}(t)$ applied on point $P_{i,j}$ at any time t. The fundamental equation of dynamics can therefore be explicitly integrated through time by a simple Euler method:

$$\begin{cases} \alpha_{i,j}(t+\Delta t) = \frac{1}{\mu} F_{i,j}(t) \\ v_{i,j}(t+\Delta t) = v_{i,j}(t) + \Delta t\, \alpha_{i,j}(t+\Delta t) \\ P_{i,j}(t+\Delta t) = P_{i,j}(t) + \Delta t\, v_{i,j}(t+\Delta t) \end{cases} \tag{7.4}$$

where Δt is a chosen time-step.



(a) (b) (c)

Figure 7.6: Self-intersection is legal in immersion. (a)-(c): The 2D circle undergoes self-intersection when exposed to external forces.

7.2.3 Rules

7.2.3.1 Self-Intersection is Legal

Returning to the rules of differential topology, we note that self-intersection is legal and allowed in an immersion (but not an embedding). Self-intersection can occur whenever a mapping happens to send two points of M^2 to the same point in $\mathbb{R}^3$. Therefore, unlike many other physically based simulations, whose intrinsic models prevent 3D collisions, we disable the collision handling mechanism and allow self-collision to support the simulation of eversion (see Figure 7.6).

7.2.3.2 Pinch-Points are Illegal

The key to immersion is motion without tearing or creasing, and thus we must make pinch-points illegal during the deformation (see Figure 7.2). We are particularly interested in finding how the shear springs and bending springs react when exposed to external forces, since the shear springs help to preserve the space between diagonal elements of the cloth-like object, and bending springs keep the model from folding like a handkerchief along the edges of the structural springs. We would

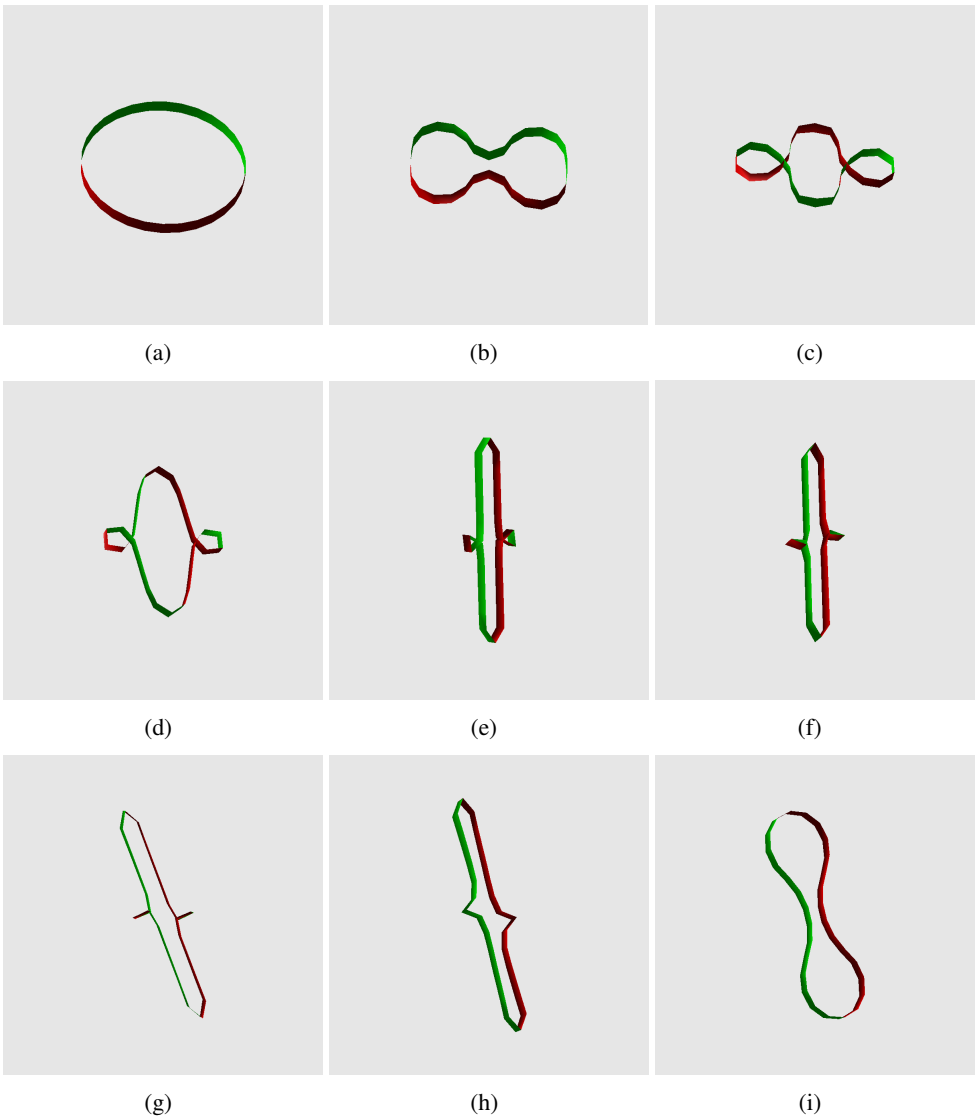


Figure 7.7: A wrong eversion of 2D circle due to the super-elastic bending spring effect.

like the fibers that compose our cloth-like material to run the length of the fabric and to resist folding and bending so as to prevent tearing or creasing during immersion.

The "Super-Elastic Bending Spring" Effect. We will study here the case of the 2D circle exposed to two opposite external forces, e.g., on the South pole and the North pole of the circle. Figure 7.7 (a)-(i) shows the deformation of a 2D circle: first, the top and bottom of the 2D circle are pushed part way through one another, leaving a red ring around the middle of a green circle; next, after an equatorial crease, the North part and South part of the circle exchange places. The 2D circle is turned inside out, but an equatorial crease (really just two points) has been produced.

This example clearly shows one of the problems: shrinkage of the bending springs is very high around the equator compared to all the other springs. The deformation of the circle is therefore locally concentrated around the equators, and the deformation rate decreases rapidly with the distance between the vertices and the equators. The value of the deformation rate of the most highly deformed bending springs exceeds 100%.

Increasing stiffness for bending springs To avoid the "super-elastic" effect, we have tried to adjust the parameters of the model. This can be accomplished partly by increasing the stiffness of the bending springs. For a similar level of constraints (same stiffness for structural springs in our case), the deformation rate should be lower for stiff bending springs. This result can indeed be attained, but not simply. Experience shows that, for a given time-step Δt and a given mass μ, there is a critical stiffness value K_c above which the numerical resolution of the system is divergent. In fact, this result is well known in the case of linear differential equations. The mathematical results

concerning such linear equations show that their numerical solution is ill-conditioned if Δt is greater than the natural period of the system [Bathe(1982)], given by:

$$T_0 \approx \pi \sqrt{\frac{\mu}{K}} \,. \tag{7.5}$$

Therefore, if we want to increase stiffness, we have to decrease Δt below the new decreased value of T_0. For a similar animation time, the number of iterations needed will then be greater, and the algorithm will be more costly.

In our model, all springs have a natural non-vanishing length, but they remain nevertheless intrinsically linear springs. However, when these springs are coupled, the cause the model to lose its linearity: it cannot be reduced to linear differential equations. However, experience shows that the result given by Eq. (7.5) is still (at least qualitatively) true in our case.

Dynamic Inverse Constraints on Deformation Rates. Another idea is to apply an ad hoc dynamic inverse procedure to the "super-shrunk" springs so as to reduce their shrinkage. At each given time-step, the numerical integration is achieved using Eq. (7.4). Then the deformation rates of all springs are computed. If, and only if, the deformation rate of a spring is greater than a critical deformation rate τ_c, then a dynamic inverse procedure is applied to the two ends of the spring so that its deformation rate exactly equals τ_c. This means that, if we choose $\tau_c = 0.1$, we want the length of the springs not to exceed their natural length by more than 10% (for many fabrics, it could even be less than that). The underlying reasoning that helped us build this procedure was the following: we assume that the position of the spring computed using Eq. (7.5) is correct regarding its direction, but not regarding the distance between the two ends of the spring; the only thing to do is then to reduce

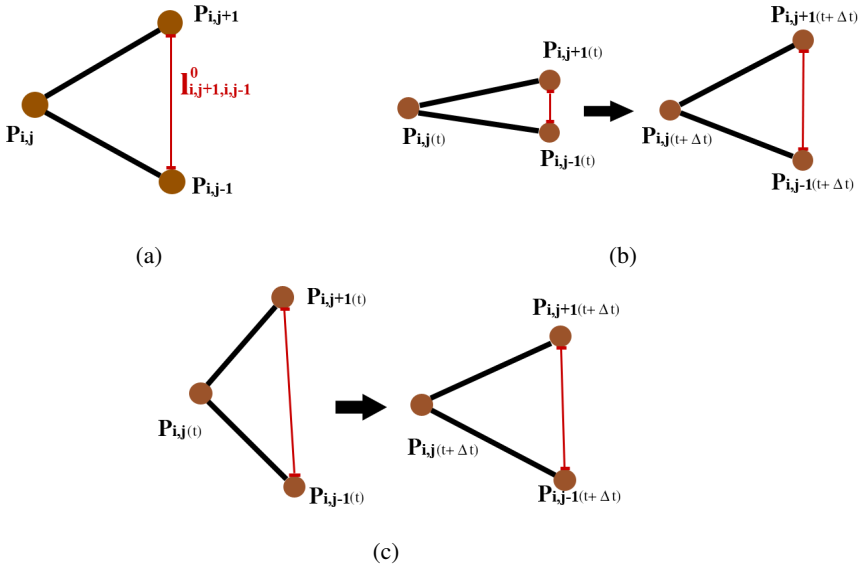


Figure 7.8: Principle of our ad hoc dynamic inverse procedure: adjust the "super-elongated" and the "super-shrunk" bending springs. (a) The natural length of the bending spring $\|P_{i,j+1}, P_{i,j-1}\| = l^0_{i,j+1,i,j-1}$. (b) Before adjustment: $\|P_{i,j+1}(t), P_{i,j-1}(t)\| \ll l^0_{i,j+1,i,j-1}$. After adjustment: $\|P_{i,j+1}(t+\Delta t), P_{i,j-1}(t+\Delta t)\| = l^0_{i,j+1,i,j-1} \times (1-\tau_c)$. (c) Before adjustment: $\|P_{i,j+1}(t), P_{i,j-1}(t)\| \gg l^0_{i,j+1,i,j-1}$. After adjustment: $\|P_{i,j+1}(t+\Delta t), P_{i,j-1}(t+\Delta t)\| = l^0_{i,j+1,i,j-1} \times (1+\tau_c)$.

this distance so that the deformation rate does not exceed τ_c while keeping the computed direction of the spring unchanged. The distance reduction is done differently, depending on whether the ends of the spring are loose or are fixed by a new dynamic inverse procedure. If both ends are loose, both are evenly "brought closer" to their middle so that the "shrunk" spring reaches τ_c. If only one end is loose, then it is "brought closer" to the fixed end so as to reach τ_c. If both are fixed, they are left unchanged.

Thus, in a single computation, all the springs with a deformation rate exceeding τ_c after the numerical integration are adjusted to a more "reasonable" deformation rate. Of course, at this point, this operation has modified the position of many vertices, and may have over-elongated other springs.

But, if the deformation is very locally concentrated, the springs affected by the operation should be less elongated than the ones that had been detected before the operation. One of the effects of the procedure is to help the deformation propagate through the structure, instead of remaining in a concentrated area. We did not take into account the order in which the super-elongated springs are adjusted at each step. In our procedure, this order depends entirely on our data structure. This is acceptable only because we restricted our method to situations in which constraints are locally distributed, that is situations in which only a few springs are super-elongated at each step. If large constraints were globally extended to the whole cloth object, then the adjustment order of the springs would probably have more importance, and should be studied.

7.2.4 Results

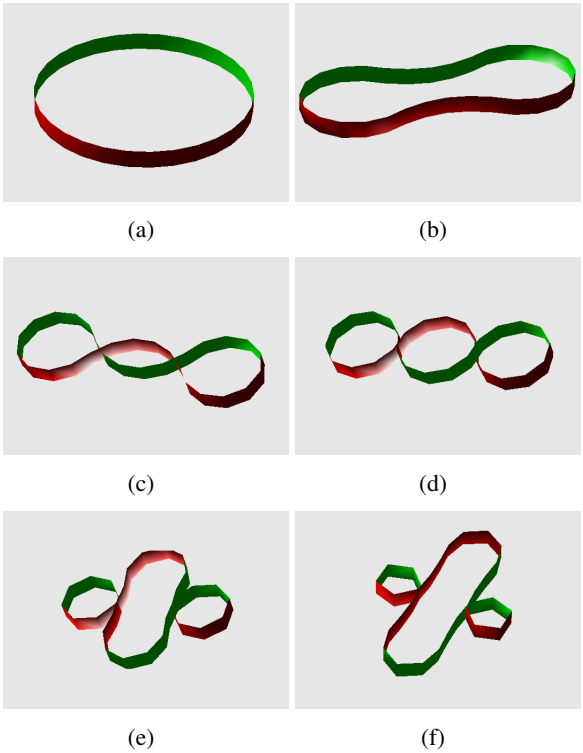
Figure 7.9 shows the screen images when we try to pull the two poles of the circle through one another. No cusps are allowed during the deformation since an ad hoc dynamic inverse procedure is applied to the "super-shrunk" springs.

7.3 Turning a 2D Circle Inside Out in 3D

7.3.1 Interacting with a 3D String

We begin by interacting with the *half-model* of 2D Circle: a piece of string in 3D.

Figure 7.10 provides a graphical illustration of the process. The process begins with a piece of string that is colored red on the outside and green on the inside. First, the top and bottom of the string



(a) (b) (c) (d) (e) (f)

Figure 7.9: Now when we pull the two poles of the circle through one another, we do not get cusps.

are pushed part of the way through one another (see Figure 7.10 (a)-(e)). This pushing effort will leave a red ring around the middle of the green string (see Figure 7.10(f)). We will not get cusps due to the Dynamic Inverse Constraints applied to the physical model. Now, if we keep pushing, the string will jitter and the ring will be twisted a bit (see Figure 7.10(g)). Continuing to push, the top and bottom of the string are then twisted 180 degrees (see Figure 7.10(h)-(o)). At this point, the original top and bottom of the string have been moved through one another. We see a different view in which the string seems both upside-down and inside-out. But the whole process is free of sharp creases and tearing.

Figure 7.11 depicts a different method to achieve the same effect as in Figure 7.10. This method first twists the ends of the 3D string a little bit (see Figure 7.11(a)-(d)), then again pushes the top and bottom portions of the string partially through one another, resulting in a ring in the middle of the string (see Figure 7.11(e)-(j)). Now, instead of continuing to push on the top and bottom portions of the string, we twist the poles in opposite directions. This has the effect of converting the ring in the middle into a twisting. Now continue to rotate the ends in opposite directions, until each has been rotated 180 degrees (see Figure 7.11(k)-(q). At this point we only need to drag on the ends (see Figure 7.11(r)-(t)), and we get what we achieved in Figure 7.10. This approach is very similar to Thurston's sphere eversion method.

7.3.2 A 2-ribbon Problem

Now we proceed to tackle the problem of turning a 2D circle inside out in 3D. We would like to think of a 2D circle as two pieces of connected 3D string, which we are now familiar with. Figure

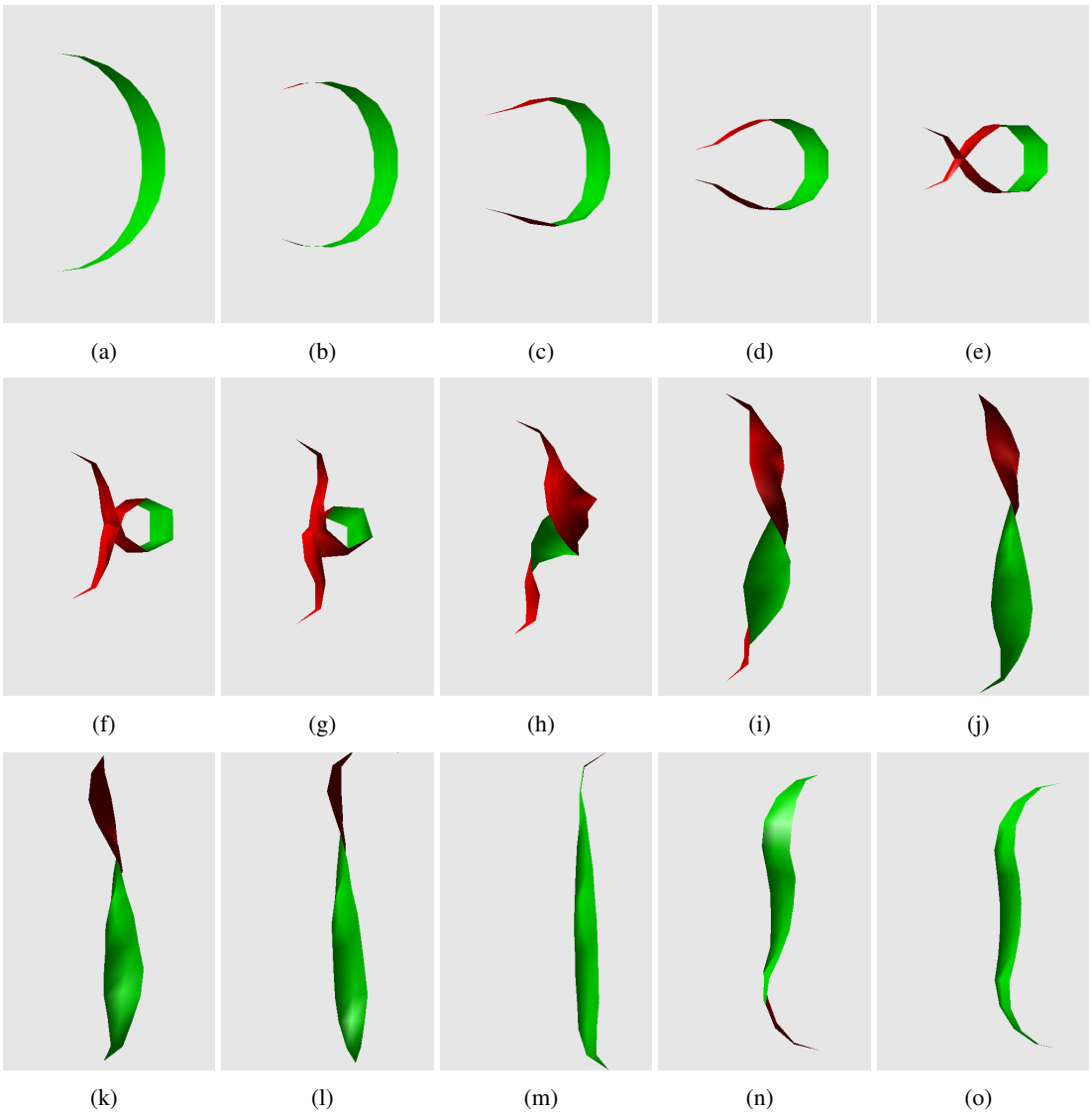


Figure 7.10: Playing with a 3D string.

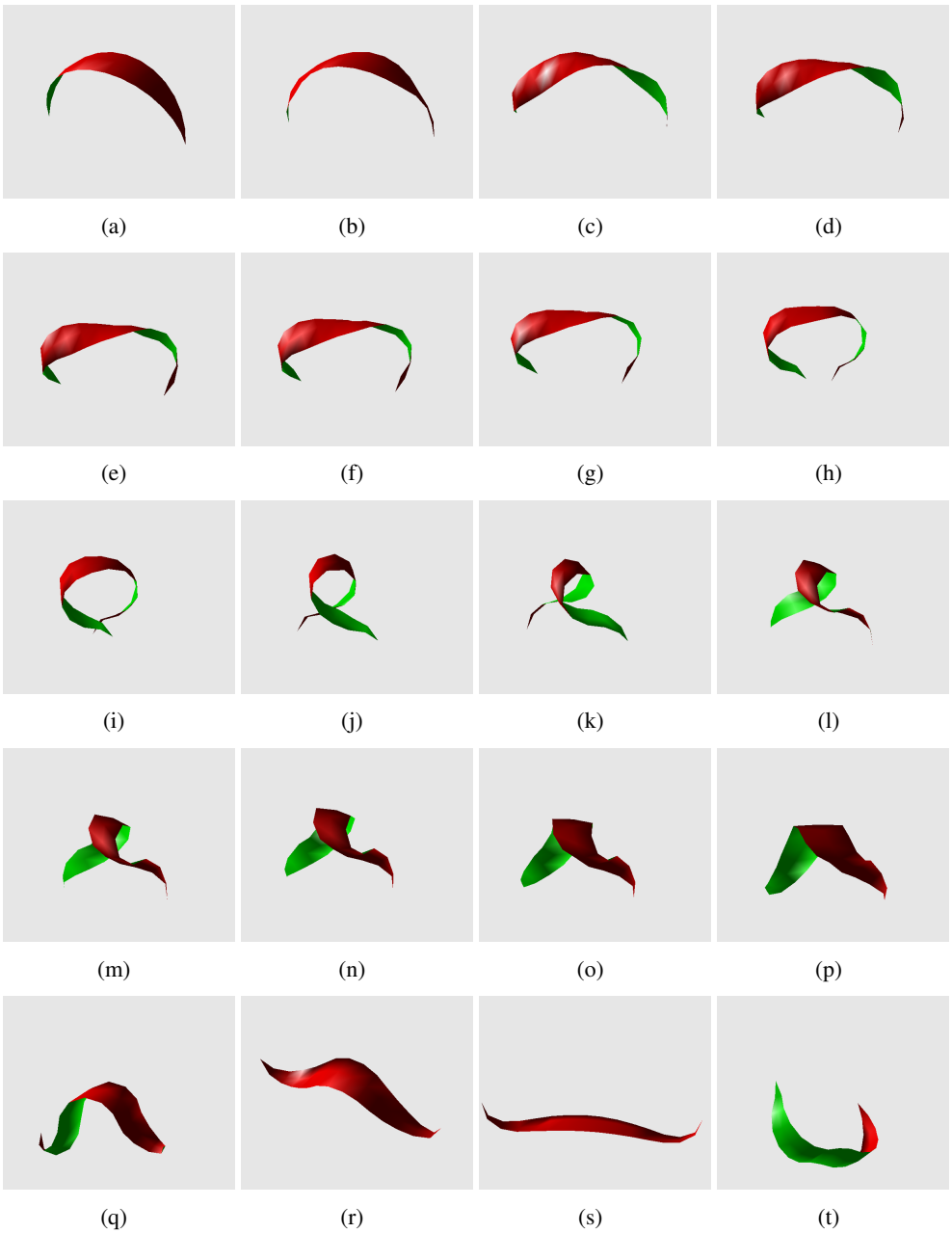
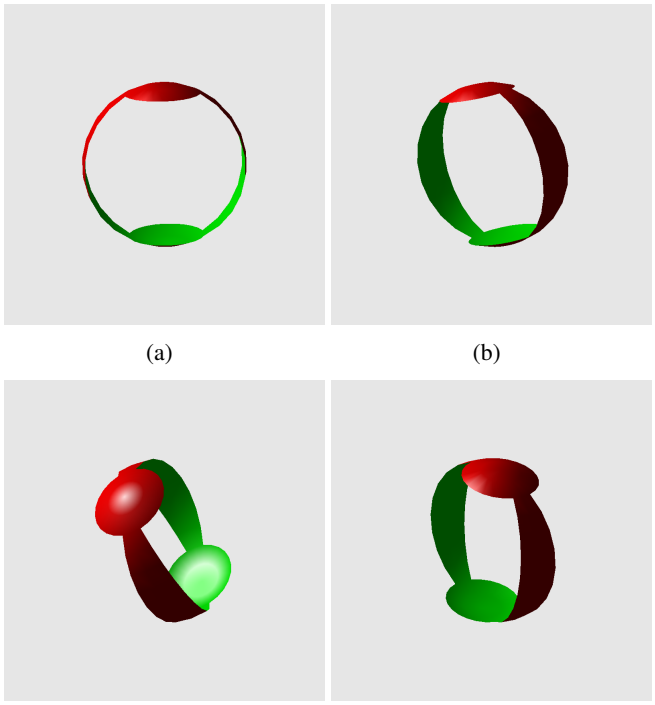


Figure 7.11: Playing with a 3D string using Thurston mode.



(a) (b)

(c) (d)

Figure 7.12: Modeling a 2D circle as two poles connected by two 3D thickened strings.

7.12 depicts our mesh model of a 2D circle. A more exact description of this model would be a sphere with all inessential parts removed, leaving only 2 ribbons and a small area of the north and south poles (to keep the 2 ribbons connected).

7.3.3 Turning a 2D Circle Inside Out

If the same procedure in Figure 7.10 is applied to a 2D circle (see Figure 7.12), we can turn a 2D circle inside out intuitively.

Figure 7.13 shows the whole procedure generating the inside-out deformation. Following what we did in Figure 7.10, we grab the north and south pole of the 2D circle, and push them part of the way

through one another, leaving two red tubes (see Figure 7.13 (a)-(e)). Now if one keeps pulling on the poles, both of the two ribbons are twisted 180 degrees, and they appear to move towards each other (but upside down, and inside out) (see Figure 7.13(f)-(o)).

As a counter-part of Figure 7.11, we can also perform the Thurston's motion on the 2D circle using twist-push-twist-pull motion. The result of such a procedure is shown in Figure 7.14.

7.4 Sphere Eversion Using a Partial Model

Our experiments on turning the 2D circle inside out have naturally led to a unique approach to physically-based Sphere Eversion. We created a partial model of the sphere by cutting a holes on the sphere to leave several ribbons, as well as north and south poles that keep the whole partial model connected. Figure 7.15 shows how we are able to push the north and south poles through one another so that the four ribbons get twisted 180 degrees and finish the eversion process.

Figure 7.16 shows another model of sphere eversion by applying the Thurston's model.

7.5 Summary

Our experiment has indicated the possibility of everting the sphere using a physical model in a natural and intuitive manner. The successful eversion of the partial sphere model suggests a potential direction to evert the sphere in a physically-driven intuitive manner.

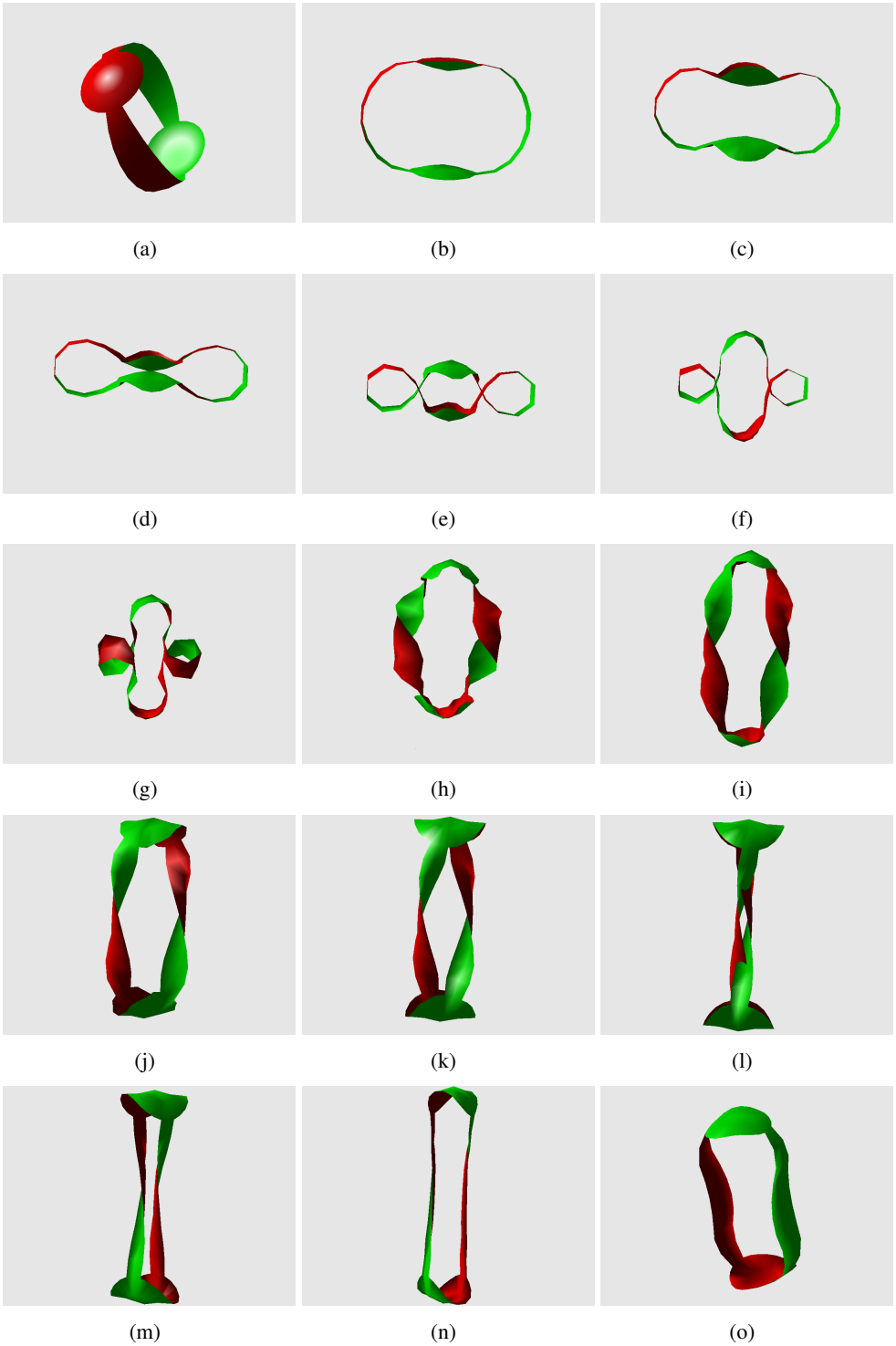


Figure 7.13: Turning a 2D circle inside out by pushing the poles.

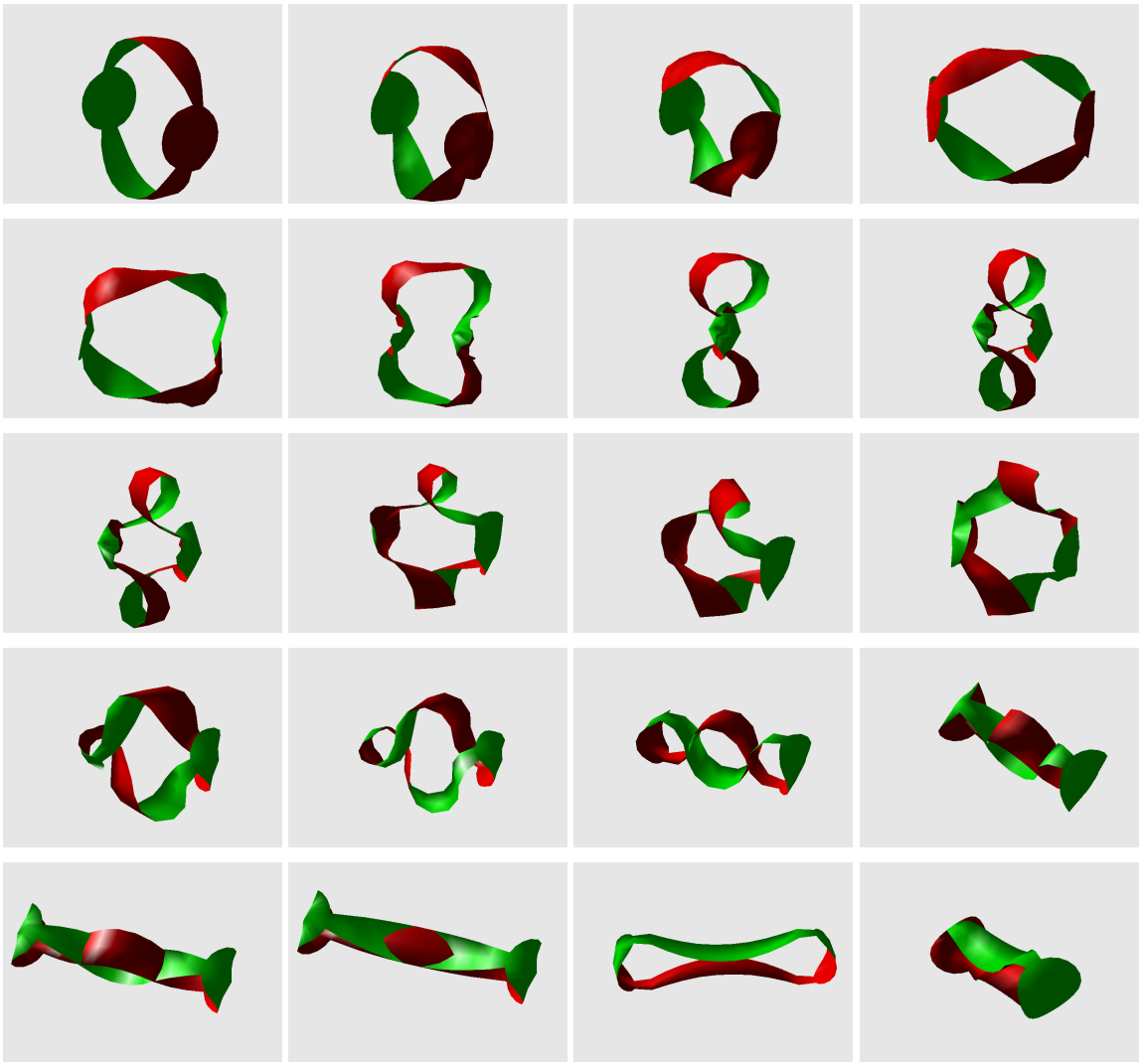


Figure 7.14: Turning a 2D circle inside out using Thurston's motion.

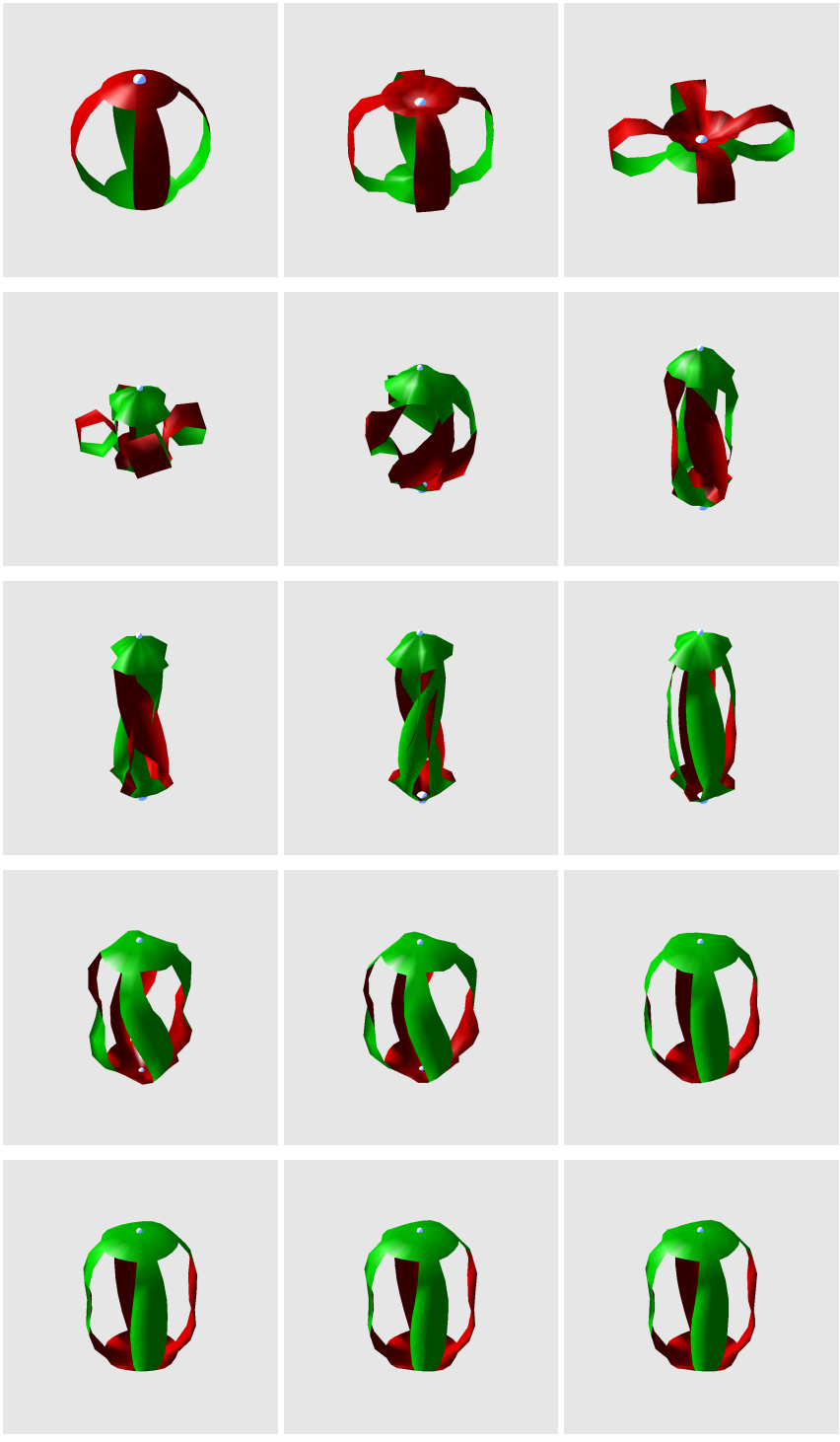


Figure 7.15: Turning a partial model of the sphere inside out, part 1.

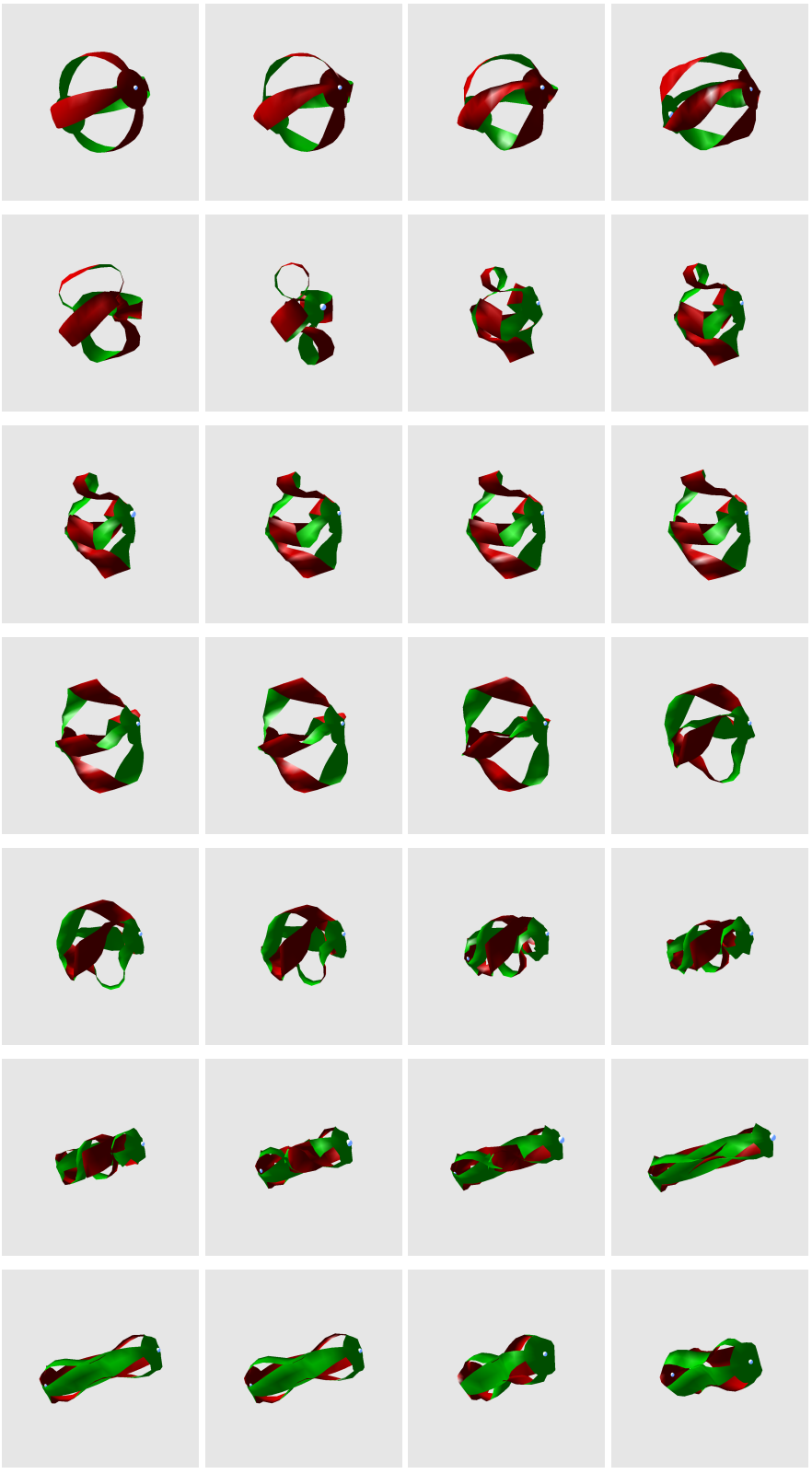


Figure 7.16: Turning a partial model of the sphere inside out, part 2.

8

Conclusions and Future Work

In this chapter, we provide a brief outline of some directions for extending the research methods and potential applications of our framework for physical interaction with four dimensions. Most of the elements of this dissertation, as suggested in the thesis title, are based on the framework of physical simulation. However, in our mathematical visualization applications, especially those related to knot theory, we have found many cases that require us to examine issues beyond physical simulation.

Visualizing Reidemeister Moves

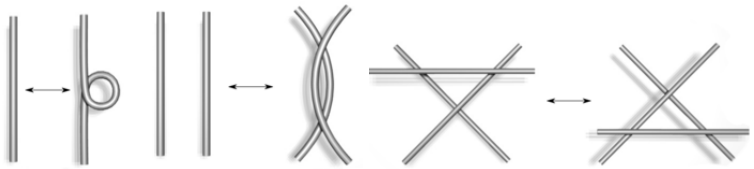
In the mathematical discipline of knot theory, a *Reidemeister Move* refers to one of three local changes to a local section of the standard diagram of a knot or link. In 1927, J.W. Alexander and G.B. Briggs, and independently Kurt Reidemeister, demonstrated that two knot diagrams belonging to the same knot, up to planar isotopy, can be related by a sequence of manipulations each of which is one of the three Reidemeister moves. Each move operates on a small region of the diagram and

is one of these three types:

1. Twist or untwist the curve in either direction.

2. Move one curve segment completely over or under another.

3. Move a curve segment completely over or under a crossing.



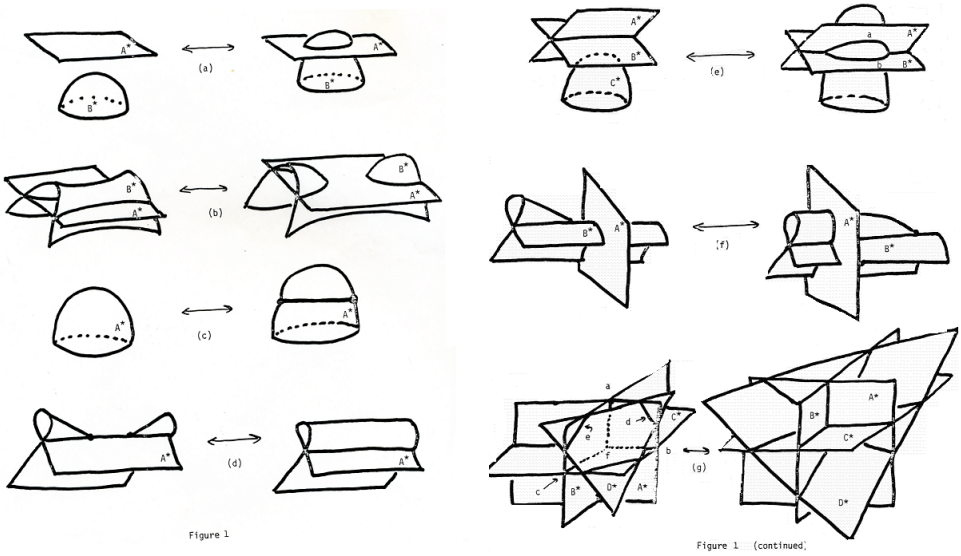
(a) (b) (c)

Figure 8.1: The three types of Reidemeister moves for 3D knots projected to a 2D knot diagram.

Reidemeister moves can be used to simplify the study the question of knot equivalence by reducing it to a two-dimensional problem. Despite their power, using Reidemeister moves to demonstrate knot equivalence can still be quite difficult. Current efforts concerning visualizing mathematical knots and modeling the dynamics of mathematical knots almost all focus on physical (or pseudo physical) simulation, which conserves both topological-constraints and lengths. Physical simulation can help to examine knot equivalence in most of the cases we are interested in, e.g., tying and untying a 3D knot or examining knot equivalence using a sequence of knot manipulations. However, Reidemeister moves are not always physical since the do not necessarily preserve the length of the knot (although a real rope should not elongate too much). Finding an improved physical model for controlling and visualizing Reidemeister moves would be a challenging and useful result.

4D Reidemeister Moves and Tying/Untying the 4D Knotted Sphere

Similarly, effective manipulation of 4D knotted spheres could potentially be greatly assisted by a friendly interface supporting 4D Reidemeister moves [Roseman(1998)](see Figure 8.2). Physical interaction with the 4D spun trefoil knot as shown in this dissertation provides an example where a physical model (i.e., mass and spring system) can be extended to four dimensions to study topological properties of 4D surfaces. While the currently implemented framework does not yet fully support interactively untying the twist-spun trefoil knot, which is actually not knotted, this should in principle be possible if the correct 4D Reidemeister moves are applied.



(a) (b)

Figure 8.2: Different types of Reidemeister-like moves that reduce deformations of a 4D surface to a three-dimensional problem

We anticipate that our physically based model for sphere eversion can be combined with knot relaxation methods to support interactively untying the twist-spun trefoil "knot."

Pseudo-Haptic Interface and Application

The "pseudo-haptic" interface that we implemented using just the mouse achieves some psychological feedback properties of haptics without requiring a force-feedback device. This could be promising in some mathematical visualization research fields, especially for tasks and interfaces involving the creation and sketching of topological objects. We have seen interest in extending our techniques for hand-held devices, tablet computing, and touchable wall interfaces. Many potential users are interested in such interfaces since they do not necessarily require a specific 3D interactive device to be installed.

Sphere Eversion

The partial model we used for physically based sphere eversion can be used for further research. While more work needs to be done to understand all facets of the mathematical and interactive processes, as well as to add more sophisticated modeling methods, we have established a framework for introducing sphere eversion to interactive, dynamic learning environments.

A

Rotations in Four Dimensions

Rotation in four-space is initially difficult to conceive because the first impulse is to try to rotate about an axis in four-space. Rotation about an axis is an idea fostered by our experience in three-space, but it is only coincidence that any rotation in three-space can be determined by an axis in three-space.

For example, consider the idea of rotation in two-space. The axis that we rotate "about" is perpendicular to this space; it isn't even contained in the two-space. In addition, given an origin of rotation and a fixed axis in three-space, the set of all rotated points for a given rotation matrix lie in a single plane, just like the two-space case.

Rotations in three-space are more properly thought of not as rotations about an axis, but as rotations in a 2D plane. This way of thinking about rotations is consistent with both two-space (where there is only one such plane) and three-space (where each rotation "axis" defines a unique rotation plane perpendicular to the normal vector to that plane).

Once this idea is established, it is easy to construct the basis for 4D rotation matrices, since only two coordinates will change for a given rotation. There are six 4D basis rotation matrices, corresponding to the XY, YZ, ZX, XW, YW and ZW planes. There are given by (using angle θ):

$$R_{xy} = \begin{vmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$R_{yz} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$R_{zx} = \begin{vmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$R_{xw} = \begin{vmatrix} \cos\theta & 0 & 0 & \sin\theta \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\sin\theta & 0 & 0 & \cos\theta \end{vmatrix}$$

$$R_{yw} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & 0 & \sin\theta \\ 0 & 0 & 1 & 0 \\ 0 & -\sin\theta & 0 & \cos\theta \end{vmatrix}$$

$$R_{zw} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\theta & \sin\theta \\ 0 & 0 & -\sin\theta & \cos\theta \end{vmatrix}$$

B

Mathematical Background of Sphere Eversion

A topologist, it has been said, is a mathematician who can't distinguish his doughnut from his coffee cup. This is because both items have one hole and are thus topologically equivalent (see, e.g., Figure B.1). That is, one can be transformed into the other via elastic compression and expansion, without tearing. A subfield of topology, called differential topology, deals with surfaces that lack singularities such as creases and pinch-points [Francis(1987)].

Even though the rules of differential topology don't allow singularities, they do allow surfaces to intersect themselves. This may seem contradictory, so some definitions will be helpful before continuing.

- The rank of a matrix A is the dimension of the image of A.
- $f : M^2 \to \mathbb{R}^3$ is an immersion of "a closed surface in space [if it is] a smooth map whose differential is everywhere of maximal rank." This map need not be one-to-one.
- A one-to-one immersion $f : M^2 \to \mathbb{R}^3$ is an embedding if it has continuous inverse.

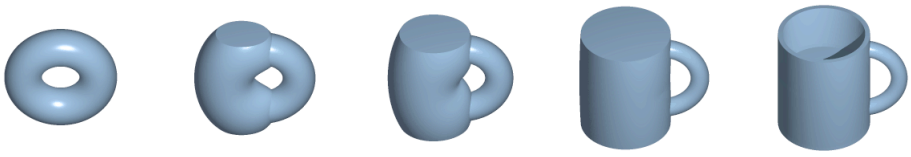


Figure B.1: A continuous deformation (homotopy) of a doughnut into a coffee cup.

- "Two regular curves are said to be regularly homotopic if one can be deformed into the other through a series of regular curves."
- An eversion of a surface is a regular homotopy from (x, y, z) to $(-x, -y, -z)$.
- The winding number or turning number "of a regular curve is the total number of counter-clockwise turns the curve makes." The analogous concept for regular surfaces is described in [Center()] as the number of "bowls," plus the number of "domes," minus the number of "saddles."
- A (closed) curve in the plane is a map from the circle into the plane.
- A curve in the plane is said to be regular if, "as a point runs around the circle at constant speed, its image moves smoothly and with a velocity other than zero in the plane."
- To evert an object means to manipulate it in such a fashion that its normal vectors reverse direction. That is, eversion is the process of turning something inside-out.

Returning to the rules of differential topology, we can now resolve the earlier confusion regarding singularities and self-intersections. An illegal pinch-point would be a place where the differential of the map is *singular* along any curve through the point, but a legal self-intersection (allowed in an immersion, but not an embedding) can occur whenever a mapping happens to send two points of M^2 to the same point in $\mathbb{R}^3$.

C

History of Sphere Eversion

C.1 A "Turning Number" for Surfaces

Intuitively, the 3D analog of looking at places where we are traveling a certain direction is to look for three dimensional "smiles" (bowls) and "frowns" (domes). At these places, one color would be facing up. That is, the normal to the surface would be $(0,0,1)$. Therefore the standard embedding of the sphere would have a three-dimensional "turning number" of 1, while that of the antipodal embedding would be -1. This would be an incomplete line of thought, however. We must also look for saddle points, and note that, when a saddle and dome or saddle and bowl come together, they cancel out. Thus the characteristic number for surfaces is $bowls + domes - saddles$. For the sphere, this number is 1 no matter which direction the normals point, be it outward or inward. In this way, then, the sphere differs from the circle. If there exists an analog of Theorem 7.2.2 for surfaces, then the sphere should be evertible.

C.2 Smale's Proof

In 1957, as a consequence of a more general theorem, Stephen Smale proved just such a result:

Theorem C.2.1. *Any two C^2 immersions of S^2 in E^3 are regularly homotopic.*

This means that any two immersions of the sphere (S^2) in 3-space (E^3) can be transformed into

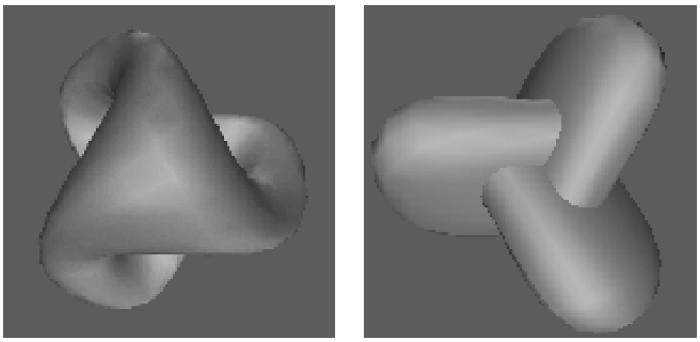


Figure C.1: The back and front of Boy's surface. ©The Geometry Center

one another through smooth motions and self-intersections. The proof of this theorem is extremely complicated and abstract, and for years no one was able to explicitly describe a process for turning the sphere inside out.

C.3 Subsequent Attempts at Eversion

Shapiro and Phillips. If self-intersection is permitted, the intuitive procedure is to push the two hemispheres of the sphere through one another. Unfortunately, this leaves a crease along the equator.

By 1960, Arnold Shapiro had devised a valid method for everting the sphere, which he explained to B.Morin but did not publish [Francis and Morin()]. One of the intermediate surfaces in his procedure was a double cover of Boy's surface, an immersion of the real projective plane.

In 1966, Anthony Phillips provided graphical illustrations of Shapiro's ideas in Scientific American [Phillips(1966)]. The sequence begins with a sphere that is light-colored on the outside and dark-colored on the inside. First, the top and bottom of the sphere are pushed part of the way through one another, leaving a pale ring around the middle of a dark sphere. Now, one of the lobes is distended

to give a figure resembling a cup of ice cream. This figure is then elongated until it looks like a saddle on legs. The bottom legs are then twisted 180^o. At this point, the artist switches to a different view, using cross sections of the surface to illuminate the many convolutions now present in the interior. Next, lobes are folded and pinched. Next, following a small motion near the seat of the saddle, the top lobes are moved through one another, as are the bottom lobes. This is the key step of the eversion, which has the effect of interchanging the colors. The eversion proceeds "backwards" to a pale sphere with a dark ring around it, and from there to the fully everted sphere.

William Thurston. However, Phillips's illustrations become progressively more difficult to follow as the eversion progresses. William Thurston developed the idea of corrugations. Unfortunately, there does not seem to be any published work on this theory. However, the important thing to know is that corrugations let pieces of a surface move around without creating sharp bends. This gives us a visually comprehensible way to evert the sphere, illustrated in Figure 7.1.

This image, the result of Thurston's work with The Geometry Center, depicts the eversion of the sphere in 12 steps. Eight corrugations are extruded from a sphere (gold on the outside, purple on the inside), then the poles are pushed partially through one another, resulting in an object that resembles a gold tire with a purple hubcap. Next, the poles are twisted in opposite directions. This had the effect of converting the middle bulge into a twisting at each corrugation. Continue to rotate the ends in opposite directions, until each has been rotated 180^o. Now, we have only to drag the bulges back through the sphere [Francis and Sullivan(2004)].

Bibliography

[Abbott(1952)] E. A. Abbott. *Flatland*. Dover Publications, Inc., 1952.

[Banchoff(1986)] T. F. Banchoff. Visualizing two-dimensional phenomena in four-dimensional space: A computer graphics approach. In E. Wegman and D. Priest, editors, *Statistical Image Processing and Computer Graphics*, pages 187–202. Marcel Dekker, Inc., New York, 1986.

[Banchoff(1990)] T. F. Banchoff. Beyond the third dimension: Geometry, computer graphics, and higher dimensions. *Scientific American Library*, 1990.

[Banchoff and Strauss(1978)] Thomas F. Banchoff and Charles M. Strauss. Real-time computer graphics analysis of figures in four-space. In *Hypergraphics: Visualizing Complex Relationship in Art, Science and Technology, Volumn 24 of AAAS Selected Symposium*, pages 159–176. Westview Press, 1978.

[Banks(1992)] D. Banks. Interactive display and manipulation of two-dimensional surfaces in four dimensional space. In *Symposium on Interactive 3D Graphics*, pages 197–207. ACM, 1992.

[Barzel and Barr(1988)] Ronen Barzel and Alan H. Barr. A modeling system based on dynamic constraints. In *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer*

graphics and interactive techniques, pages 179–188, New York, NY, USA, 1988. ACM Press. ISBN 0-89791-275-6. doi: http://doi.acm.org/10.1145/54852.378509.

[Bathe(1982)] Klaus-Jurgen Bathe. *Finite Element Procedures in Engineering Analysis*. Prentice Hall, 1982.

[Baxter et al.(2001)Baxter, Scheib, and Lin] W. V. Baxter, V. Scheib, and M. C. Lin. dAb: Interactive haptic painting with 3D virtual brushes. In Eugene Fiume, editor, *SIGGRAPH 2001, Computer Graphics Proceedings*, pages 461–468. ACM SIGGRAPH, ACM, 2001. URL `citeseer.nj.nec.com/foster01practical.html`.

[Bishop(1975)] R. L. Bishop. There is more than one way to frame a curve. *American Mathematical Monthly*, 82(3):246–251, March 1975.

[Brisson(1978a)] David W. Brisson, editor. *Hypergraphics: Visualizing Complex Relationships in Art, Science and Technology*, volume 24 of *AAAS Selected Symposium*, Boulder, Colorado, 1978a. Westview Press.

[Brisson(1978b)] David W. Brisson. Visual comprehension of n-dimensions. In *Hypergraphics: Visualizing Complex Relationship in Art, Science and Technology, Volumn 24 of AAAS Selected Symposium*, pages 109–145. Westview Press, 1978b.

[Brown et al.(2004)Brown, Latombe, and Montgomery] J. Brown, J. C. Latombe, and K. Montgomery. Real-time knot-tying simulation. *The Visual Computer*, 20(2-3):165–179, 2004.

[Carey et al.(1987)Carey, Burton, and Campbell] S. A. Carey, R. P. Burton, and D. M. Campbell. Shades of a higher dimension. *Computer Graphics World*, pages 93–94, October 1987.

[Carter(1995)] Scott Carter. *How surfaces intersect in space: An introduction to topology*. World Scientific, 2 edition, 1995.

[Center()] The Geometry Center. Outside in: The script of outside in(part 4).

[Chen et al.(2006)Chen, Barner, and Steiner] Pei Chen, Kenneth E. Barner, and Karl V. Steiner. A displacement driven real-time deformable model for haptic surgery. *haptics*, 0:75, 2006. doi: http://doi.ieeecomputersociety.org/10.1109/HAPTICS.2006.147.

[Cohen et al.(1999)Cohen, L.Markosian, Zeleznik, Hughes, and R.Barzel] J.M. Cohen, L.Markosian, R.C. Zeleznik, J.F. Hughes, and R.Barzel. An interface for sketching 3d curves. In *SI3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics*, pages 17–21, New York, NY, USA, 1999. ACM Press. ISBN 1-58113-082-1. doi: http://doi.acm.org/10.1145/300523.300655.

[C.W.Reynolds(1999)] C.W.Reynolds. Steering behaviors for autonomous characters. In *The proceedings of the 1999 Game Developers Conference*, pages 763–782, 1999.

[Desbrun et al.(2000)Desbrun, Meyer, and Barr] Mathieu Desbrun, Mark Meyer, and Alan H. Barr. Interactive animation of cloth-like objects for virtual reality. In *Cloth modeling and animation*, pages 219–239, Natick, MA, USA, 2000. A. K. Peters, Ltd. ISBN 1-56881-090-3.

[Dewdney(1984)] A. K. Dewdney. *The Planiverse: Computer Contact with a Two-Dimensional World*. Poseidon Press, 1984.

[Dickson(2000)] Stewart Dickson. Tactile mathematics. In *Mathematics and Art: Proceedings*

of the International Colloquium on Art and Mathematics, Maubeuge, France, 2000. ISBN 3-540-43422-4.

[Dobrinov and Eichhorn(2007)] Atanas Dobrinov and Volkmar Eichhorn. A modular software haptic interface for teleoperated nanomanipulation and afm probe based characterization of carbon nanotubes. In *WHC '07: Proceedings of the Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 543–550, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-2738-8.

[Eberly(2001)] D.H. Eberly. *3D Game Engine Design*. Morgan Kaufmann Publisher, 2001.

[Egli et al.(1996)Egli, Petit, and Stewart] R. Egli, C. Petit, and N. F. Stewart. Moving coordinate frames for representation and visualization in four dimensions. *Computers and Graphics*, 20 (6):905–919, November–December 1996.

[Fitzpatrick and Karshmer(2004)] Donal Fitzpatrick and Arthur I. Karshmer. Multi-modal mathematics: Conveying math using synthetic speech and speech recognition. In Joachim Klaus, Klaus Miesenberger, Wolfgang L. Zagler, and Dominique Burger, editors, *ICCHP*, volume 3118 of *Lecture Notes in Computer Science*, pages 644–647. Springer, 2004. ISBN 3-540-22334-7.

[Forsyth(2004)] Ben Forsyth. Intelligent support of interactive manual control: Design, implementation and evaluation of look-ahead haptic guidance. Master's thesis, The University of British Columbia, 2004.

[Francis and Morin()] G. K. Francis and B. Morin. Arnold shapiro's eversion of the sphere. *Math.*

Intell.

[Francis and Sullivan(2004)] George Francis and John M. Sullivan. Visualizing a sphere eversion. *IEEE Transactions on Visualization and Computer Graphics*, 10(5):509–515, 2004. ISSN 1077-2626. doi: http://dx.doi.org/10.1109/TVCG.2004.33.

[Francis(1983)] George K. Francis. Drawing seifert surfaces that fiber the figure-8 knot complement in s3 over s1. *The American Mathematical Monthly*, 90(9):589–599, November 1983.

[Francis(1987)] George K. Francis. *A Topological Picturebook*. Springer-Verlag, New York, 1987.

[Gardberg(1996)] Anna S. Gardberg. Everting the sphere. December 1996.

[Gottschalk et al.(1996)Gottschalk, Lin, and Manocha] S. Gottschalk, M. C. Lin, and D. Manocha. OBBTree: A hierarchical structure for rapid interference detection. *Computer Graphics*, 30(Annual Conference Series):171–180, 1996. URL `citeseer.ist.psu.edu/gottschalk96obbtree.html`.

[Gray(1993)] A. Gray. *Modern Differential Geometry of Curves and Surfaces*. CRC Press, Inc., Boca Raton, FL, 1993.

[Hadamard(1996)] Jacques Hadamard. *The Mathematician's Mind: The Psychology of Invention in the Mathematical Field*. Princeton University Press, Princeton, NJ, 1996. Revised from original 1945 edition.

[Haeberli(1989)] Paul Haeberli. Dynadraw. *Grafica OBSCURA*, July 1989. URL `http://www.sgi.com/grafica/dyna/index.html`.

[Hanson(1994a)] A. J. Hanson. A construction for computer visualization of certain complex curves. *Notices of the Amer.Math.Soc.*, 41(9):1156–1163, November/December 1994a.

[Hanson and Heng(1992)] A. J. Hanson and P. A. Heng. Illuminating the fourth dimension. *Computer Graphics and Applications*, 12(4):54–62, July 1992.

[Hanson and Ma(1995a)] A. J. Hanson and H. Ma. Space walking. In *Proceedings of Visualization '95*, pages 126–133. IEEE, IEEE Computer Society Press, 1995a. ISBN 0-8186-7187-4.

[Hanson and Ma(1995b)] A. J. Hanson and H. Ma. Space walking. In *Proceedings of Visualization '95*, pages 126–133. IEEE, IEEE Computer Society Press, 1995b. ISBN 0-8186-7187-4.

[Hanson and Zhang(2005)] A. J. Hanson and H. Zhang. Multimodal exploration of the fourth dimension. In *Proceedings of IEEE Visualization*, pages 263–270, 2005.

[Hanson et al.()Hanson, Ishkov, and Ma] A.J. Hanson, K. Ishkov, and J. Ma. Meshview. A portable 4D geometry viewer written in OpenGL/Motif, available by anonymous ftp from ftp.cs.indiana.edu:pub/hanson.

[Hanson(1994b)] Andrew J. Hanson. Geometry for n-dimensional graphics. pages 149–170, 1994b.

[Hanson et al.(1999)Hanson, Ishkov, and Ma] Andrew J. Hanson, Konstantine I. Ishkov, and Jeff H. Ma. Meshview: Visualizing the fourth dimension. Technical report, Indiana University, 1999.

[Hilbert and Cohn-Vossen(1952)] D. Hilbert and S. Cohn-Vossen. *Geometry and the Imagination*. Springer-Verlag, Chelsea, New York, 1952.

[Hlavác et al.(1996)Hlavác, Leonardis, and Werner] Václav Hlavác, Ales Leonardis, and Tomás Werner. Automatic selection of reference views for image-based scene representations. In *ECCV (1)*, pages 526–535, 1996.

[Hollasch(1991)] S. Hollasch. Four-space visualization of 4D objects. Master's thesis, Arizona State University, 1991.

[Huang et al.(1996)Huang, Grzeszczuk, and Kauffman] M. Huang, R. P. Grzeszczuk, and L. H. Kauffman. Untangling knots by stochastic energy optimization. In *VIS '96: Proceedings of the 7th conference on Visualization '96*, pages 279–ff., Los Alamitos, CA, USA, 1996. IEEE Computer Society Press. ISBN 0-89791-864-9.

[Iwata et al.(2004)Iwata, Yano, Uemura, and Moriya] Hiroo Iwata, Hiroaki Yano, Takahiro Uemura, and Tetsuro Moriya. Food texture display. *haptics*, 00:310–315, 2004. ISSN Pending.

[J.Lander(1999)] J.Lander. Devil in blue faceted dress: Real-time cloth animation. *Game Developer*, 1999.

[Kamada and Kawai(1988)] Tomihisa Kamada and Satoru Kawai. A simple method for computing general position in displaying three-dimensional objects. *Computer Vision, Graphics, and Image Processing*, 41(1):43–56, 1988.

[Kennedy(2002)] J. M. Kennedy. Optics and haptics: The picture. In *The conference on Multimodality of Human Communication: Theory, Problems and Applications.* University of Toronto, 2002.

[Kim et al.(2003)Kim, Sukhatme, and Desbrun] L. Kim, G. Sukhatme, and M. Desbrun. Haptic

editing for decoration and material properties, 2003. URL `citeseer.ist.psu.edu/kim03haptic.html`.

[Kim(1978)] Scott E. Kim. An impossible four-dimensional illusion. In *Hypergraphics: Visualizing Complex Relationship in Art, Science and Technology, Volumn 24 of AAAS Selected Symposium*, pages 187–239. Westview Press, 1978.

[Larsson and Akenine-Möller(2001)] T. Larsson and T. Akenine-Möller. Collision detection for continuously deforming bodies. In *Eurographics 2001*, pages 325–333, Manchester, September 2001. Eurographics Association. doi: http://www.mrtc.mdh.se/index.phtml?choice=publications&id=0354.

[Lecuyer et al.(2000)Lecuyer, Coquillart, Kheddar, Richard, and Coiffet] Anatole Lecuyer, Sabine Coquillart, Abderrahmane Kheddar, Paul Richard, and Philippe Coiffet. Pseudo-haptic feedback: Can isometric input devices simulate force feedback? *vr*, 00:83, 2000. ISSN 1087-8270.

[Lécuyer et al.(2004)Lécuyer, Burkhardt, and Etienne] Anatole Lécuyer, Jean-Marie Burkhardt, and Laurent Etienne. Feeling bumps and holes without a haptic interface: the perception of pseudo-haptic textures. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 239–246, New York, NY, USA, 2004. ACM. ISBN 1-58113-702-8.

[L.Kim et al.(2004)L.Kim, Sukhatme, and M.Desbrun] L.Kim, G.S. Sukhatme, and M.Desbrun. A haptic-rendering technique based on hybrid surface representation. *IEEE Comput. Graph. Appl.*, 24(2):66–75, 2004. ISSN 0272-1716. doi: http://dx.doi.org/10.1109/MCG.2004.

1274064.

[Massie and Salisbary(1994)] Thomas H. Massie and J. Kenneth Salisbary. The PHANToM haptic interface: a device for probing virtual objects. *ASME Dynamic Systems and Control*, 55(1): 295–301, 1994.

[Maybeck(1979)] Peter S. Maybeck. *Stochastic Models, Estimation, and Control, Volume 1*. Academic Press, New York, NY, 1979. The first chapter is available at www.cs.unc.edu/∼welch/Kalman/index.html.

[Monagan(1996)] M.B. Monagan. *Maple V programming guide*. New York, 1996.

[Mulder et al.(1998)Mulder, Groen, and van Wijk] Jurriaan D. Mulder, Frans C. A. Groen, and Jarke J. van Wijk. Pixel masks for screen-door transparency. In David Ebert, Hans Hagen, and Holly Rushmeier, editors, *Proceedings of Visualization '98*, pages 351–358. IEEE, IEEE Computer Society Press, 1998. URL `citeseer.ist.psu.edu/mulder98pixel.html`.

[Negenborn(2003)] Rudy Negenborn. Robot localization and kalman filters. Master's thesis, Utrecht University, 2003.

[Noll(1978)] A. Michael Noll. Displaying n-dimensional hyperobjects by computer. In *Hypergraphics: Visualizing Complex Relationship in Art, Science and Technology, Volumn 24 of AAAS Selected Symposium*, pages 147–158. Westview Press, 1978.

[Noll(1967)] M. A. Noll. A computer technique for displaying n-dimensional hyperobjects. *Communications of the ACM*, 10(8):469–473, August 1967.

[Okamura(2000)] A. M. Okamura. *Haptic Exploration of Unknown Objects*. PhD thesis, Stanford University, Department of Mechanical Engineering, California, USA, June 2000.

[Okamura and Cutkosky(2001)] A. M. Okamura and M. R. Cutkosky. Feature detection for haptic exploration with robotic fingers. *The International Journal of Robotics Research*, 20(12): 925–938, December 2001.

[Park and Niemeyer(2004)] June Gyu Park and Günter Niemeyer. Haptic rendering with predictive representation of local geometry. In *HAPTICS*, pages 331–338, 2004.

[Phillips(1966)] Anthony Phillips. Turning a surface inside out. *Scientific American*, May 1966.

[Phillips et al.(1993)Phillips, Levy, and Munzner] Mark Phillips, Silvio Levy, and Tamara Munzner. Geomview: An interactive geometry viewer. *Notices of the Amer. Math. Society*, 40 (8):985–988, October 1993. Available by anonymous ftp from geom.umn.edu, The Geometry Center, Minneapolis MN.

[Provot(1997)] Xavier Provot. Collision and self-collision handling in cloth model dedicated to design garments. In *Proc. Computer Animation and Simulation'97*, pages 177–189, 1997.

[Raymaekers et al.(2005)Raymaekers, Vanacken, Cuppens, and Coninx] C. Raymaekers, L. Vanacken, E. Cuppens, and K. Coninx. A comparison of different techniques for haptic cloth rendering. In *HAPTEX '05: Proceedings of the VR Workshop on Haptic and Tactile Perception of Deformable Objects*, pages 56–63. Magnenat-Thalmann N.[edit.], e.a., s.l., 2005.

[Roseman(1998)] D. Roseman. Reidemeister-type moves for surfaces in four-dimensional space.

Knot theory (Warsaw, 1995), pages 347–380, 1998.

[Roseman(1993)] D. Roseman. Twisting and turning in four dimensions, 1993. Video animation, Department of Mathematics, University of Iowa, and the Geometry Center.

[Sánchez and Sáenz(2005)] Jaime Sánchez and Mauricio Sáenz. 3d sound interactive environments for problem solving. In *Assets '05: Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility*, pages 173–179, New York, NY, USA, 2005. ACM. ISBN 1-59593-159-7. doi: http://doi.acm.org/10.1145/1090785.1090817.

[Scharein(1998)] R. G. Scharein. *Interactive Topological Drawing*. PhD thesis, Department of Computer Science, The University of British Columbia, 1998.

[Sen(2004)] *3D Touch SDK OpenHaptics Toolkit Programmer's Guide*. SensAble, Inc., 2004.

[Simon(2001)] Dan Simon. Kalman filtering. *Embedded Systems Programming*, 14(6):72–79, June 2001.

[Smale(1958)] Stephen Smale. A classification of immersions of the two-sphere. *Transactions of the American Mathematical Society*, 90:281–290, 1958.

[Snibbe et al.(1998)Snibbe, Anderson, and Verplank] S. Snibbe, S. Anderson, and B. Verplank. Springs and constraints for 3d drawing. In *Proceedings of the Third Phantom Users Group Workshop*, Dedham, MA, 1998.

[Sreng et al.(2006)Sreng, Lécuyer, Mégard, and Andriot] Jean Sreng, Anatole Lécuyer, Christine Mégard, and Claude Andriot. Using visual cues of contact to improve interactive manipulation

of virtual objects in industrial assembly/maintenance simulations. *IEEE Trans. Vis. Comput. Graph.*, 12(5):1013–1020, 2006.

[Steiner and Burton(1987)] K. V. Steiner and R. P. Burton. Hidden volumes: The 4th dimension. *Computer Graphics World*, pages 71–74, February 1987.

[Terzopoulos and Witkin(1988)] Demetri Terzopoulos and Andrew Witkin. Physically based models with rigid and deformable components. *IEEE Comput. Graph. Appl.*, 8(6):41–51, 1988. ISSN 0272-1716. doi: http://dx.doi.org/10.1109/38.20317.

[Terzopoulos et al.(1987)Terzopoulos, Platt, Barr, and Fleischer] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 205–214, New York, NY, USA, 1987. ACM Press. ISBN 0-89791-227-6. doi: http://doi.acm.org/10.1145/37401.37427.

[van Wijk and Cohen(2005)] Jarke J. van Wijk and Arjeh M. Cohen. Visualization of the genus of knots. pages 567–574, 2005.

[van Wijk and Cohen(2006)] Jarke J. van Wijk and Arjeh M. Cohen. Visualization of seifert surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):485–496, 2006. doi: http://dx.doi.org/10.1109/TVCG.2006.83.

[Vázquez et al.(2001)Vázquez, Feixas, Sbert, and Heidrich] Pere-Pau Vázquez, Miquel Feixas, Mateu Sbert, and Wolfgang Heidrich. Viewpoint selection using viewpoint entropy. In *VMV '01: Proceedings of the Vision Modeling and Visualization Conference 2001*, pages 273–280.

Aka GmbH, 2001. ISBN 3-89838-028-9.

[Vázquez et al.(2002)Vázquez, Feixas, Sbert, and Llobet] Pere-Pau Vázquez, Miquel Feixas, Mateu Sbert, and Antoni Llobet. Viewpoint entropy: a new tool for obtaining good views of molecules. In *VISSYM '02: Proceedings of the symposium on Data Visualisation 2002*, pages 183–188, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association. ISBN 1-58113-536-X.

[Wejchert and Haumann(1991)] Jakub Wejchert and David Haumann. Animation aerodynamics. In *SIGGRAPH '91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, pages 19–22, New York, NY, USA, 1991. ACM Press. ISBN 0-89791-436-8. doi: http://doi.acm.org/10.1145/122718.122719.

[Welch and Bishop(1995)] Greg Welch and Gary Bishop. An introduction to the Kalman filter. Technical Report TR 95-041, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC 27599-3175, 1995.

[Whitney(1937)] Hassler Whitney. On regular closed curves in the plane. *Compositio Mathematica*, 4:274–284, 1937.

[Wolfram(1991)] Stephen Wolfram. *Mathematica — A System for Doing Mathematics by Computer*. Redwood City, California, 2 edition, 1991.

[Yu et al.(2000)Yu, Ramloll, and Brewster] W. Yu, R. Ramloll, and S.A. Brewster. Haptic graphs for blind computer users. In *First International Workshop on Haptic Human-Computer Interaction*, pages 102–107. British HCI Group, Springer, 2000.

[Zahariev and MacKenzie(2003)] M. A. Zahariev and C. L. MacKenzie. Auditory, graphical and haptic contact cues for a reach, grasp, and place task in an augmented environment. In *Proc. of 5th Intl. Conf on Multimodal Interfaces (ICMI)*, pages 273–276, New York, 2003. ACM, ACM Press. ISBN 0-8186-7187-4. URL `http://doi.acm.org/10.1145/958481`.

[Zhang and Hanson(2006)] Hui Zhang and Andrew J. Hanson. Physically interacting with four dimensions. In *ISVC (1)*, pages 232–242, 2006.

[Zilles and Salisbury(1995)] C. B. Zilles and J. K. Salisbury. A constraint-based god-object method for haptic display. In *IROS '95: Proceedings of the International Conference on Intelligent Robots and Systems-Volume 3*, page 3146, Washington, DC, USA, 1995. IEEE Computer Society. ISBN 0-8186-7108-4.

Index

4D Mass-Spring system, 125

4D Spun Trefoil, 131

4D Torus, 134

4D depth cue, 95

4D knotted sphere, 131

4D light, 21

4D visualization, 3

Abbott, Edwin Abbott

- A Square, 25
- Flatland, 23

Andrew J. Hanson, 20

- Movie
  - *knot*^{4}, 21
  - 4Dice, 21
  - FourSight, 21
  - Visualizing Fermat's Last Theorem, 21
- Software
  - MeshView, 18

BSP tree, 98

Computer 3D Printing, 17

Computer haptics, 6

Dimensional Progressions, 26

Distance in 4D

- Distance between 4D line segments, 109
- Distance between 4D point and triangle, 111

Dynamic Inverse Constraints, 152

Euler Integration, 148

Fechner, Gustave, 22

- Space has Four Dimensions, 22
- Vier Paradoxe, 22

Forces in 4D

- External force, 126
- Internal force, 126

Geomview, 18

George Francis, 14

A Topological Picturebook, 14

Grid drawing interface, 55

Haptic “rubber band line”, 130

Hooke’s law, 37

Hypercube, 27

KnotExplore, 22

Linear springs in Mass-spring system, 145

Link Diagram, 165

Mathematical visualization, 1

Meshview, 63

PHANTOM device, 7

Plato

Allegory of the Cave, 22

The Republic, 22

Pseudo haptics, 52

Reidemeister Move, 165

Riemann surface, 21

Screen-door view, 62

Shadow driven editing, 120

Sphere Eversion, 142

Thomas Banchoff

Book

Beyond the Third Dimension, 15

Movie

Hypercube, 20

The Veronese Surface, 20

Viewpoint entropy, 48

Viewpoint selection, 46

Curriculum Vitae

First Name: Hui

Last Name: Zhang

Place of Birth: Fuzhou, China

Date of Birth: Nov. 22, 1977

Educational Institutions Attended

Indiana University, Bloomington	2002-2008
Zhejiang University, China	1995-2002

Degrees Awarded

Ph. D. (Computer Science) Indiana University, Bloomington	2008
M. Sc. (Computer Science) Zhejiang University, China	2002
B. Sc. (Computer Science) Zhejiang University, China	1999

Honours and Awards

Graduate Research Assistantship	2003-2008
Graduate Teaching Scholarship	2002
Honorable Mention Prize in the 2005 SensAble 3D Touch Developer Challenge	2005

PUBLICATIONS

Hui Zhang, Andrew J. Hanson. *Shadow-driven 4D Haptic Visualization.* In Proceedings of IEEE Visualization, 2007, accepted for publication in a special issue in IEEE Transactions on Visualization and Computer Graphics (TVCG).

Hui Zhang, Sidharth Thakur, Andrew J. Hanson. *Haptic Exploration of Mathematical Knots.* In Proceedings of ISVC2007. Accepted.

Hui Zhang, Andrew J. Hanson. *Physically Interacting with Four Dimensions.* ISVC(1), volume 4291 of Lecture Notes in Computer Science, pages 232-242. Springer, 2006.

Chen Yu, Hui Zhang, Linda B. Smith. *Learning Through Multimodal Interaction.* In Proceedings of the 5th International Conference of Development and Learning. Bloomington, IN, 2006.

Hui Zhang, Chen Yu, Linda B. Smith. *An Interactive Virtual Reality Platform for Studying Embodied Social Interaction.* In Proceedings of the CogSci06 Symposium Toward Social Mechanisms of Android Science. Vancouver, Canada, 2006.

Chen Yu, Hui Zhang, Linda B. Smith. *Using Virtual Humans to Study Embodied Social Interaction.* In Proceedings of the 5th International Conference of the Cognitive Science. Vancouver, Canada, 2006.

Linda B. Smith, Chen Yu, Hui Zhang. *Using Virtual Humans to Study the Role of Social Cues in Learning.* In Proceedings of the annual meeting of the XVth Biennial International Conference on

Infant Studies, Westin Miyako, Kyoto, Japan, June 19, 2006.

Andrew J. Hanson, Hui Zhang. *Multimodal Exploration of the Fourth Dimension.* In proceedings of IEEE Visualization, pages 263-270, 2005.

Jianguang Weng, Yueting Zhuang, Hui Zhang. *Error-Bounded Solid Voxelization for Polygonal Model Based on Heuristic Seed Filling.* ISVC 2005: 607-612.