Physically Interacting with Four Dimensions

Hui Zhang and Andrew J. Hanson

Computer Science Department, Indiana University
huizhang@cs.indiana.edu, hanson@cs.indiana.edu

Abstract. We exploit the combination of a virtual world containing physically-interacting 4D objects with a multimodal haptics-driven user-interface model; the goal is to facilitate the development of accurate cognitive models enabling the visualization of 4D space. Our primary test domain supports tactile interaction with physically colliding and deformable curves and surfaces embedded in 4D, an important and challenging subject area of classical topology. We implement intricate interactions involving 4D curves and surfaces by haptically manipulating 3D projections of these objects.

1 Introduction

In the seventh book of the *Republic*, 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. 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 shading and occlusion that create perceptions far richer than the bleak shadows of Plato's vision, and which we interpret in our mental models as being truly 3D, despite the fact that in truth their dimension is reduced. Interactive control systems familiar to us all, such as the 1D steering wheel controlling the 2D motion of a car, or a 2D joystick controlling the 3D motion of an aircraft, provide further examples of physically-reactive controls that have lower dimensions than the domain of the object motion being manipulated.

Our task in this paper is to show how one can fully exploit the concept of projections to lower dimensions and physically reactive projection-based controllers to transform the task of interacting with 4D objects to a new level of physical reality. We like to think of this method intuitively as working in a "shadow world," a term widely used in the 4D visualization literature, with clear interactive implications and an ancient context in Plato's philosophy, though we will avoid the term for the most part due to implicit confusion with conventional computer graphics terminology. We start from the fairly familiar idea that a 2D mouse can control 3D objects represented as rendered images projected on a 2D screen, and extend that idea to a haptic interface that is restricted to a plane, but still empowered to sketch, touch, and take control of 3D objects projected to the controller plane. We then extend this concept upward by one dimension, creating 3D projections of 4D objects, rendering them with shading and occlusions, and providing 3D haptic controls and force-feedback in the resulting 3D world to sketch and control a 4D world. In this way we are able to create a realistic interactive interface that embodies the physical realities of a simulated 4D mathematical world of curves and surfaces just as standard 3D physical modeling embodies the properties of such objects in the human world. Thus we can make a non-trivial computer interface that correctly implements the intuitive physical properties of classes of 4D geometric problems whose comprehension is extremely challenging for the unaided human intellect.

2 Previous Work

The idea of cross-dimensional understanding has developed in many directions since Plato described his Cave. Abbott's *Flatland* asked how two-dimensional creatures might attempt to understand three-dimensional space [1,2]. Banchoff's pioneering work suggested how 3D computer-based projections could be used to study 4D objects [3,4]. Other representative efforts include a variety of ways to render 4D objects (see, e.g., Noll [5], Hollasch [6], Banks [7], Roseman [8], and Egli, Petit, and Stewart [9]), and to extend lighting model techniques to 4D (see, e.g., [10,11,12]).

Our own previous efforts have suggested how haptic exploration of 4D objects can exploit topological continuity by ignoring illusory 3D surface intersections and focusing on the intrinsic 4D geometry [13]. Typically, visual methods for understanding higher dimensions employ a projection to 3D as a fundamental step; this helps the viewer to identify salient global features of the whole object, and provides structural continuity when rotating either the object's (rigid) 3D projection, or performing higher-dimensional rotations to change the 3D projection. Haptic exploration, being intrinsically limited to the physical world, also must project higher dimensional objects to lower dimensions for exploration; within this context, surfaces embedded in 4D can be projected to 3D and explored topologically without regard to 3D artifacts to reveal complex topological relationships and structure. What is needed now to go beyond this framework is an enhancement of the environment that allows 4D intuition-building construction, interaction, weight, and deformation, as well as exploration of the 4D shapes themselves. The problem addressed here is therefore: "How can we physically interact with the fourth dimension?"

3 Motivation

People learn about the everyday world by combining sensory modalities, and knowledge of shape comes from a combination of sight, touch, and exploration. With a 3D touch-based computer interface, a 3D projection of, e.g., a 4D surface or curve can be used to explore intrinsic 4D geometry. We therefore use extensions of 3D methods to help us manipulate and comprehend the complicated case of shapes with collisions and weights in a 4D simulated world using a touch-based multimodal paradigm.

2D Example. To explain the basic features of our approach, we 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 projected shadow of the 3D crossing, we find the result in Figure 1(a), which is devoid of 3D information. This problem is typically overcome by employing the "crossing diagram" method illustrated in Figure 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 1(c). Now we can begin to see how to exploit a haptic probe: if the probe is constrained to a 2D plane, the user can still edit the planar projections of curves and use modifier keys to specify over and under crossings to create intertwined 3D objects back in the "real" 3D space. Hence, the 2D projection supports 3D interaction.

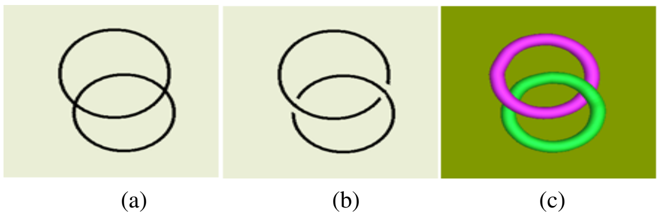


Fig. 1. (a) 2D projection diagram of crossing curves for two 3D rings. (b) Knot diagram representation of the linked rings provides 3D depth information. (c) Rendering with light and material adds apparent 3D geometry and shape to the 2D image.

3D Example. The technique 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 is that the resulting surfaces intersect in the 3D projection, even when in 4D space there are no intersections or singularities of any kind.

Figure 2 shows the 3D projection of a 4D ribbon linked with a 4D spherical surface, giving precise analogs to Figure 1, with the "bare shadow" in Figure 2(a), the crossing diagram in 2(b), and 4D depth pseudocoloring in 2(c,d).

Projection Editing. The key ideas of the overall scenario can now be summarized as follows:

- *Create a projection ("shadow") space in one lower dimension.*
- *Edit projected images of objects embedded in the higher dimension.* We must account for the lost depth information from the extra dimension, and cope with possibly having the haptic probe's status confused when one segment collides with another in the projection.
- *Adding an extra $\frac{1}{2}$ dimension.* Sketching in the reduced-dimension space can be supported by explicitly defining over and under crossings displaced slightly in the extra dimension when apparent collisions occur in the projected (shadow) image.
- *Create a crossing-diagram object.* This is basically the schematic equivalent of a 4D volume depth buffer.

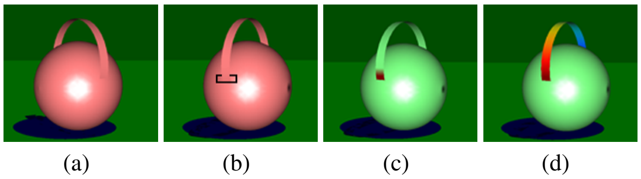


Fig. 2. (a) Two intersecting surfaces in 3D, typically resulting from the projection of a 4D scene to 3D. (b) Crossing diagram of the surfaces; the front portion of the ribbon surface is closer to the 4D projection point than the spherical surface. (c) Above-below crossing markings using 4D depth coding. (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.

– *Enhance the geometry and depth information in the higher dimension.* Exploit computer graphics and visualization methods to enhance the perception of the geometry by color depth coding and exaggerated occlusion or crossing diagrams.
– *Physically interact with the higher dimensions.* By mapping the projected lower-dimensional (shadow) user input to the full-dimensional object along the projection rays, we can sense physical artifacts such as collision and gravity in the higher dimensional simulation.

4 Projection Editing Using Haptic Methods

We now describe the critical haptic portion of our interface. There have been a number of interesting attempts to exploit intuitive haptic interfaces to improve the sense of realism and to enhance the manipulation of virtual objects (see, e.g., [14], [15], and [16]). Another direction of research has focused on the haptic exploration of unknown objects by virtual fingers (see, e.g., [17] and [18]).

4.1 Design and Implementation

Procedure. The basic design of the system relies on the construction of a projection from 4D to 3D that can optionally annotate the under-over crossings in the projection, as well as supporting 6-degree-of-freedom 4D rotations to reposition the 3D projection arbitrarily to support the particular visualization requirements.

Constrained Haptic Space. Our basic force model simulates a “sticky” stylus in the projection space using a damped spring configuration model[19].

Collision Avoidance in Projected Space. When editing 2D projections of 3D curves or 3D images of surfaces embedded in 4D, collision detection and state management are essential issues. During editing, collision detection is continuously enabled to detect whether the object collides with itself or with other scene objects. The OpenHaptics HDAPI has been exploited in our implementation.

Making Over and Under Choices. When a collision occurs between a piece of an edited object and an existing object in projected space, users must make explicit over and under

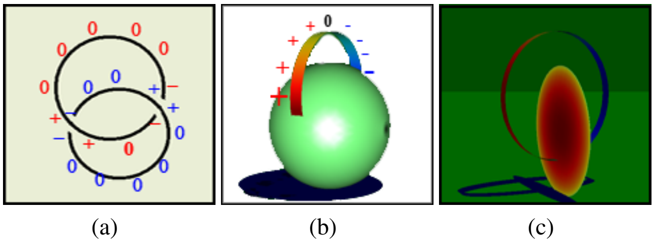


Fig. 3. (a) Recovering 3D depth from the 2D diagram.(b) Recovering 4D depth from the 3D diagram. (c) Projected image in YZW space with smoothed 4D depth.

choices before they can pass through the intersection. Thus the interface has a kind of interactive collision avoidance protocol.

4.2 Results

Editing Projections of 3D Curves. Traditional methods used by mathematicians for sketching knots and links make heavy use of crossing diagrams, and interactive graphics methods can assist this process (see, e.g., [20] and [21]). Using haptics-based projection-space editing can extend this paradigm still further. Figure 3(a) illustrates a simple example of the link-construction process; the resulting 3D curves in Figure 1(c) have been smoothed using the Minimum Distance Energy method [22] to fill out the $2\frac{1}{2}$D sketches to make a more natural shape.

Editing Projections of 4D Surfaces. We can analogously create 4D links working with the projected images of the 4D surfaces. We depend on the 3D collision mechanisms to locate possible crossings, and rotate or vary the 3D projection axes to help us see the 4D over and under crossings analogous to Figure 3(a), as shown in Figure 3(b). Applying a smoothing algorithm to the 4D depth of the sketched ribbon results in the more natural shapes of Figure 3(c).

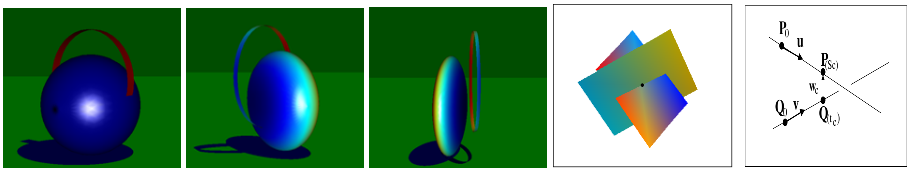
5 Collision in Four Dimensions

Applying general rotations to objects projected from higher dimensions smoothly alters the projected images. For example, if projections of 3D rings falsely appear to be linked, a properly chosen 3D rotation will detach the two projected images from each other. The analogous observation holds when we apply 4D rotations to 4D objects represented by their 3D projections, as shown in Figure 4(a)-(c).

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

5.1 4D Collision Detection

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



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

Fig. 4. (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.(d) 4D Collision: surfaces are color-coded to indicate their height 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. (e) Closest Points between 4D Lines.

4(d)(e)). 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, 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 projection, but that 3D collisions in the projection may not imply 4D collisions.*

4D Collision Detection Based on Projection. Figure 4(e) 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 have the 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 $\mathbf{L}_{\mathrm{SE}}$, 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 projected 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 $\mathbf{P}_e$. Obviously, $\mathbf{P}_0$, $\mathbf{P}_1$, $\mathbf{Q}_0$, and $\mathbf{Q}_1$ can be represented as:

$$\begin{aligned}
\mathbf{P}_0 &= (x_s, y_s, z_s, w_{sA}) \\
\mathbf{Q}_0 &= (x_s, y_s, z_s, w_{sB}) \\
\mathbf{P}_1 &= (x_e, y_e, z_e, w_{eA}) \\
\mathbf{Q}_1 &= (x_e, y_e, z_e, w_{eB})
\end{aligned} \tag{1}$$

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 τ.

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 4(e)). 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:

$$\begin{aligned} \mathbf{u} \cdot \mathbf{w}_c &= 0 \\ \mathbf{v} \cdot \mathbf{w}_c &= 0 \,. \end{aligned} \tag{2}$$

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 one to get two simultaneous linear equations. Then, letting $a = \mathbf{u}\cdot\mathbf{u}$, $b = \mathbf{u}\cdot\mathbf{v}$, $d = \mathbf{u}\cdot\mathbf{w}_0$, and $e = \mathbf{v}\cdot\mathbf{w}_0$, we solve for s_c and t_c as:

$$s_c = \frac{be - cd}{ac - b^2}, \quad t_c = \frac{ae - bd}{ac - b^2} \,. \tag{3}$$

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:

$$d(\mathbf{L}_1, \mathbf{L}_2) = \left| (\mathbf{P}_0 - \mathbf{Q}_0) + \frac{(be - cd)\mathbf{u} - (ae - bd)\mathbf{v}}{ac - b^2} \right| \,. \tag{4}$$

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 $0 \leq s \leq 1$. 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 $0 \leq t \leq 1$. The distance between segment $\mathbf{S}_1$ 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 [23], 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.

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 4D depth overlap at all. 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 nearest-approach computation.

5.2 4D Collision Management

Physics-based simulation of the 4D world involves challenging problems in 4D collision management. Self-collisions of flexible objects must be treated as well as collisions between distinct 4D objects, both rigid and flexible.

4D self-collision management methods follow exact parallels to 3D collision management (see, e.g., [20], [24]). The most basic form of self-collision treatment is as follows: When two 4D facets are detected to be at a distance $d < \tau$ from each other (i.e., the two corresponding facets are defined to be colliding if they are closer than a minimum distance τ), the pair of closest points in the facets is identified and Δ is defined as the 4D line passing through them. Then an equal (but opposite) displacement is applied to each facet along Δ. This displacement is just large enough to take the facets out of collision range, with a slight "safety margin" ε to allow for a sliding motion. Standard 3D self-collision mechanisms must also be extended to apply Newton's laws to four dimensions in a natural way. By stretching, compressing, bending, and twisting the 4D object while keeping its topological structure, we can convert the mathematical abstraction of any 4D object into an interactive reality.

4D collisions between distinct 4D rigid objects are handled in a similar way to self-collisions.

6 Physically Interacting with Four Dimensions Using Haptic Methods

Physics-based simulations of 4D collision detection and management are 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., [25]) and deformable objects (see, e.g., [26], [27],[20], and [28]). Haptic interfaces have also been used to implement realistic 3D physics-based simulations (see [14] and [16]). 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 projections.

Example 1: 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 projection domain in which it is applied, with the unseen projection-direction coordinate held fixed. Just as one can lift 2D projected images of 3D rings with control constrained to the 2D projection plane, we can lift 3D projected images of 4D object configurations in the same way.
- Force of gravity and damping. 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):

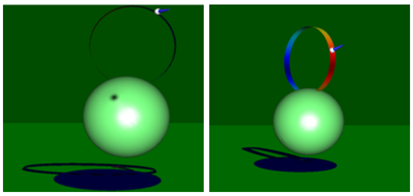


Fig. 5. The haptic interface for lifting a chain element in four dimensions, sensing collisions and weight

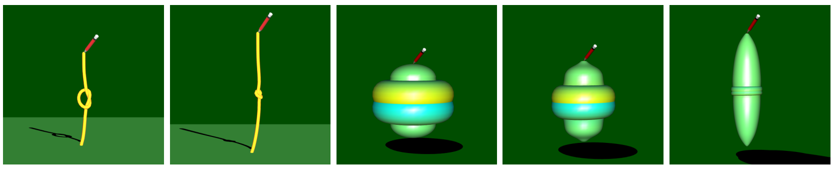


Fig. 6. Left: Tightening a 3D curve. Right: Tightening a 4D knotted sphere by pulling in 3D.

↪ Move the haptic proxy, attached to a ribbon at a user-specified location, in projection 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.
↪ The haptic proxy is free again to move the ribbon with constrained motions.

Example 2: Tightening the Spun Trefoil Knot. Just as there are many ways to represent and alter the same 3D knot using equivalent crossing diagram representations, there are also many ways to represent the same knotted sphere as a projection from 4D to 3D. Manipulating these deformations via a 4D touch-sensitive interface in the projection to 3D allows one to verify the validity of the transformations among the various projected forms (corresponding typically to what a knot-theorist would call “Reidemeister moves.”).

In our version of the standard “spun-knot” construction [29] of a 4D knotted sphere, the user interactively sketches a 3D knot K using the 3D editing interface, then selects a “spin plane” to spin K into a 4-dimensional knot.

Tightening a 4D spun knot is similar to a 3D knot-tightening simulation. The FTL (Follow the Leader) algorithm described in [20] can be extended to the fourth dimension to compute the 4D knotted sphere tightening process (see Figure 6). The overall algorithm for tightening the spun trefoil knot is the following:

↪ Read the new position of the location grasped by the haptic proxy (typically, the north pole of the knotted sphere).
↪ Prepare and apply forces on the connected physical mesh components.
↪ Apply forces and compute the new knotted sphere configuration (global motion).
↪ Compute tentative (self-)collisions.

$\hookrightarrow$ If a potential collision is detected between two 4D segments or facets, apply collision forces with friction and gravity.
$\hookrightarrow$ The haptic proxy is free again to move the grasped location.

7 Conclusion and Future Work

Our implementation is based on a standard OpenGL graphics system with a high-performance graphics 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.

We have created a multimodal computer interface to interact physically with the fourth dimension via its 3D projection. Our approach fluidly integrates visual information with haptic feedback and interaction. We have exploited these capabilities to build and interact with components of a 4D simulated world, a world that we can work with physically and that is no less real than a 3D computer simulation; we have thus presented a practical way to connect the mathematical abstraction of 4D objects to a visible, touchable interactive reality.

Among our objectives for future work are to extend the range of scene objects that we can support to include more complex knots, links, and Riemann surfaces, as well as three-manifolds in addition to curves and surfaces. In a more practical direction, one might imagine exploring high-dimensional information data sets by exploiting analogs of our interface acting along projection rays from the higher dimensional information space into a 3D control space.

References

1. Abbott, E.A.: Flatland. Dover Publications, Inc. (1952)
2. Dewdney, A.K.: The Planiverse: Computer Contact with a Two-Dimensional World. Poseidon Press (1984)
3. Banchoff, T.F.: Visualizing two-dimensional phenomena in four-dimensional space: A computer graphics approach. In Wegman, E., Priest, D., eds.: Statistical Image Processing and Computer Graphics. Marcel Dekker, Inc., New York (1986) 187–202
4. Banchoff, T.F.: Beyond the third dimension: Geometry, computer graphics, and higher dimensions. Scientific American Library (1990)
5. Noll, M.A.: A computer technique for displaying n-dimensional hyperobjects. Communications of the ACM **10** (1967) 469–473
6. Hollasch, S.: Four-space visualization of 4D objects. Master's thesis, Arizona State University (1991)
7. Banks, D.: Interactive display and manipulation of two-dimensional surfaces in four dimensional space. In: Symposium on Interactive 3D Graphics, ACM (1992) 197–207
8. Roseman, D.: Twisting and turning in four dimensions (1993) Video animation, Department of Mathematics, University of Iowa, and the Geometry Center.
9. Egli, R., Petit, C., Stewart, N.F.: Moving coordinate frames for representation and visualization in four dimensions. Computers and Graphics **20** (1996) 905–919
10. Carey, S.A., Burton, R.P., Campbell, D.M.: Shades of a higher dimension. Computer Graphics World (1987) 93–94

11. Steiner, K.V., Burton, R.P.: Hidden volumes: The 4th dimension. Computer Graphics World (1987) 71–74
12. Hanson, A.J., Heng, P.A.: Illuminating the fourth dimension. Computer Graphics and Applications **12** (1992) 54–62
13. Hanson, A.J., Zhang, H.: Multimodal exploration of the fourth dimension. In: Proceedings of IEEE Visualization. (2005) 263–270
14. Baxter, W.V., Scheib, V., Lin, M.C.: dAb: Interactive haptic painting with 3D virtual brushes. In Fiume, E., ed.: SIGGRAPH 2001, Computer Graphics Proceedings, ACM SIGGRAPH, ACM (2001) 461–468
15. Thompson, T., Nelson, D., Cohen, E., Hollerbach, J.: Maneuverable nurbs models within a haptic virtual environment (1997)
16. Kim, L., Sukhatme, G., Desbrun, M.: Haptic editing for decoration and material properties (2003)
17. Okamura, A.M.: Haptic Exploration of Unknown Objects. PhD thesis, Stanford University, Department of Mechanical Engineering, California, USA (2000)
18. Yu, W., Ramloll, R., Brewster, S.: Haptic graphs for blind computer users. In: First International Workshop on Haptic Human-Computer Interaction, British HCI Group, Springer (2000) 102–107
19. SensAble, Inc.: 3D Touch SDK OpenHaptics Toolkit Programmer's Guide. (2004)
20. Brown, J., Latombe, J.C., Montgomery, K.: Real-time knot-tying simulation. The Visual Computer **20** (2004) 165–179
21. Scharein, R.G.: Interactive Topological Drawing. PhD thesis, Department of Computer Science, The University of British Columbia (1998)
22. Huang, M., Grzeszczuk, R.P., Kauffman, L.H.: Untangling knots by stochastic energy optimization. In: VIS '96: Proceedings of the 7th conference on Visualization '96, Los Alamitos, CA, USA, IEEE Computer Society Press (1996) 279–ff.
23. Eberly, D.: 3D Game Engine Design. Morgan Kaufmann Publisher (2001)
24. Phillips, J., Ladd, A., Kavraki, L.: Simulated knot tying (2002)
25. Barzel, R., Barr, A.H.: A modeling system based on dynamic constraints. In: SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM Press (1988) 179–188
26. Terzopoulos, D., Platt, J., Barr, A., Fleischer, K.: Elastically deformable models. In: SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM Press (1987) 205–214
27. Terzopoulos, D., Witkin, A.: Physically based models with rigid and deformable components. IEEE Comput. Graph. Appl. **8** (1988) 41–51
28. Wejchert, J., Haumann, D.: Animation aerodynamics. In: SIGGRAPH '91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM Press (1991) 19–22
29. Friedman, G.: Knot spinning. Handbook of knot theory (2005) 187–208