Comput. & Graphics Vol. 16, No. 3, pp. 273–281, 1992
Printed in Great Britain.

0097-8493/92 $5.00 + .00
© 1992 Pergamon Press Ltd.

SWEEPING WITH ALL GRAPHICAL INGREDIENTS IN A TOPOLOGICAL PICTUREBOOK

VAROL AKMAN
Department of Computer Engineering and Information Science, Faculty of Engineering, Bilkent University, Bilkent, 06533 Ankara, Turkey

and

AHMET ARSLAN
Department of Electronics Engineering, Faculty of Engineering, Firat University, Elaziğ, Turkey

Abstract—Sweeping is a powerful method to generate 3-D shapes in geometric modeling. In this paper we formulate a general matrix to give a mathematical definition of twisted-profiled sweep objects as a discrete approximation. While conceptually simple, our result is, to our best knowledge, the first precise formulation of sweeping with all graphical ingredients, *viz.* twisting, scaling, rotation, and translation. Twisted-profiled sweeping surfaces defined by contour, profile, trajectory, and guide curves are thus represented in concatenated matrix formulation. In addition, we give interactive methods to generate sweep objects and present sample figures produced within the framework of our implementation Tb, a topological picturebook.

1. INTRODUCTION

A classification of *sweep objects* is given by Choi and Lee[15]. Their classification treats sweep surfaces that are obtained via coordinate transformations and blending. Another classification is given by Bronsvoort, van Nieuwenhuizen, and Post[13]. According to them, sweep objects can be defined in four types:

- *Translational sweep*—The contour is arbitrary and the trajectory is a straight line.
- *Rotational sweep*—The contour is arbitrary and the trajectory is a circle.
- *Circle sweep*—The contour is circular and the trajectory is arbitrary.
- *General sweep* (or *Generalized cylinder*)—Both the contour and the trajectory are arbitrary.

Here, we are interested mainly in generalized cylinders. Definitions of generalized cylinders are given by Bronsvoort and Klok[12], Post and Klok[24], and Bronsvoort, van Nieuwenhuizen, and Post[13]. Building generalized cylinders is conceptually easy; just use a closed 2-D contour curve and a 3-D trajectory curve along which the contour is swept. The spatial relationship between the contour and the trajectory is given by various methods. The first method is the so-called *Frenet frame*[12, 13, 19, 24]. Here, the trajectory is a 3-D vector valued function and the frame at each point of the trajectory is determined by three orthogonal vectors (commonly known as tangent, normal, and binormal vectors). Thus, a local coordinate system is defined at each point of the trajectory curve. The contour curve must be perpendicular to the tangent vector at each point of the trajectory. The second method, called *rotational minimizing sweep*, is a planar approximation of general sweep, where the trajectory is considered to be a polyline [13, 19, 24].

Other previous approaches for obtaining sweep objects are briefly as follows. Control points of Bézier or B-spline surfaces can be handled by sweeping; so, it is possible to generate some sweep surfaces that can be deformed locally[12]. In addition, a 3-D object can be subtracted from the original blank while it is swept along a 3-D path to obtain a new 3-D object[21, 26].

It is possible to deform a sweep object by scaling and twisting the contour curve as it is swept along the trajectory curve[24]. Deformed sweep objects obtained by scaling are called *profiled sweep objects*[13]. Coquillart presents an offset technique for control points of rational B-spline curves to produce sweep or profiled sweep objects[16]. Choi and Lee use simple transformations and blending to represent more complex sweep objects[18]. Woodward presents a skinning technique to produce 3-D shapes that resemble blended sweep objects[27].

In this paper, we are essentially interested in discrete sweeping and hence, every curve used in the sequel is considered a sequence of discrete points. Accordingly, it is not really important what curves are used. We present a general matrix for representing twisted-profiled sweep objects. The matrix provides quick response time and is especially useful for interactive systems*.

In Section 2, we formulate a general matrix as a mathematical definition of a sweep object deformed by scaling and twisting. In Section 3, we present interactive methods to generate sweep shapes and give some sample figures produced by our implementation, Topologybook (Tb in short)[†]. Tb is a general workbench that facilitates the production of publication-quality topological pictures and is fashioned along the lines of [18][‡]. The implementation of Tb is nearing its com-

* It is well known that matrix representations in computer graphics have the advantage that they suit well to the internal representation of the computer in which they are implemented.

[†] The choice of the symbols T and b has been inspired by their use in logic and music—areas whose chief concerns are truth and beauty.

[‡] Unfortunately, all the computer-generated images in this paper are wire-frame pictures in black and white. Although Tb generates color pictures on the screen, we presently lack a color hardcopy device.

pletion and will constitute the doctoral dissertation of the second author. Although we will briefly describe it in the Appendix, the reader is referred to our other publications for more information on Tb and its origins; these include [2–8].

2. TWISTED-PROFILED GENERAL SWEEP

Sweeping can be defined informally as follows [24]. A sweep surface $Sw(C, T)$ is produced by moving a given contour curve C along a given trajectory curve T. The plane of C must be perpendicular to T at any point of T. In the rest of this paper, C will be any planar (closed or open) curve and T will be any 3-D (closed or open) curve. If C is a closed curve and T is a 3-D curve segment, then the produced sweep object is called a *generalized cylinder* [24].

If C is deformed by scaling with a scale factor as it is swept along T, then the produced sweep object is called a *profiled general sweep* object. Here, we accept as the scale factor the distance of a given curve P to any selected axis. So, a profiled general sweep surface can be defined informally as $Sw_p(C, T, P)$. P represents a profile curve that is planar and has the same number of points with T.

If C is deformed by twisting and scaling while sweeping along T, the produced surface is called a *twisted-profiled general sweep object.* We denote the twist factor as Tw; this is a planar curve that has the same number of points with T and is mapped between the minimum and the maximum values of the twisting angles only at one dimension. Then, a twisted-profiled general sweep surface can be defined informally by four curves, *viz.*, $Sw_{tp}(C, T, P, Tw)$.

In the following subsections, we present a discrete mathematical definition of twisted-profiled general sweeping. In this definition, we use the conventional matrix notation to represent the transformations and obtain a general sweep matrix as a result. Some matrix representations for sweeping are given by Rogers and Adams[25]. Theirs, however, are restricted and work only for given planar curves and situations that depend on curve equations. In our representation, the matrix twists, scales, and sweeps a contour curve C along a trajectory curve T to handle a grid of general sweep surface points. We will next formulate the discrete mathematical definitions of the curves involved (*e.g.*, C, T, P, Tw) as vector functions.

C is a planar curve or polygon with n points. Each point $(x_i, y_i, 0)$ of C can be represented as:

$$C_i \quad (i = 1, 2, \ldots, n). \tag{1}$$

T is any 3-D curve or polygon with m points and (without loss of generality) the first point of T is in the origin. Each point (x_j, y_j, z_j) of T can be represented as:

$$T_j \quad (j = 1, 2, \ldots, m). \tag{2}$$

P can be given as a 1-D position vector with m component vectors that include scaling factors. But here, we take P as a planar curve and later calculate the scaling factors by the help of the curve. This is useful for interactive systems. Each point $(x_j, y_j, 0)$ of P can be represented as:

$$P_j \quad (j = 1, 2, \ldots, m). \tag{3}$$

Tw can be also given as a 1-D position vector with m vectors that include twisting factors. But, we give Tw as a planar curve and later calculate the twisting factors by the help of the curve. Each point $(x_j, y_j, 0)$ of Tw can be represented as:

$$Tw_j \quad (j = 1, 2, \ldots, m). \tag{4}$$

Planar curves P and Tw, and the calculation of the scaling and the twisting factors are presented in detail in Sections 2.1 and 2.2. In the sequel, we give a procedure to handle a twisted-profiled general sweep object. In this procedure, C is a curve centered at the origin. (The first point of T is also at the origin.) In other cases, they must be translated to origin and then the generated curves must be translated back to their actual positions at the end of the procedure. First, C is rotated around the z-axis by Tw for twisting. Second, C is scaled by P. Third, C is rotated around the x-, y- and z-axes by angles found for each point of T. Finally, each twisted, scaled, and rotated C is translated to every point of T.

2.1. *Twisting*

We have taken the twisting factor Tw as a planar curve and defined it as in Eq. (4). But, it represents only the twisting angles for each point of the trajectory curve T. That is, it can be represented as a 1-D vector function linearly interpolated at only one dimension of Tw. Tw can be drawn interactively as a planar curve in an interactive system and is then interpolated at one dimension to handle twisting angles (see Section 3 for details). The interpolated 1-D vector function θ includes rotating angles for each point of T and can be defined as follows:

$$\theta_j \quad (j = 1, 2, \ldots, m). \tag{5}$$

The following matrix rotates a contour curve C that is centered at the origin (for twisting as a function of θ):

$$[R_{tw}] = \begin{bmatrix} \cos\theta_j & \sin\theta_j & 0 & 0 \\ -\sin\theta_j & \cos\theta_j & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6}$$

2.2. *Scaling*

We take a profile curve P (a planar curve) as the scaling factor and define it as in Eq. (3). It also represents only one magnitude for each point of the trajectory curve T. Then, it can also be written as a 1-D vector function, which takes the distances of the vertices of P to the vertical (or horizontal) axis at only one dimension. For an interactive system, first, an axis is given for reference and P is drawn interactively. Then, distances of vertices of P at one dimension to the axis give the scaling factor for C (see Section 3 for details). Scaling factor s is a 1-D vector function and includes scaling radii for points of T. It can be defined as:

$$s_j \quad (j = 1, 2, \ldots, m). \tag{7}$$

The following matrix[§4] scales a contour curve C that is centered at the origin, as a function of s:

$$[S_p] = \begin{bmatrix} s_j & 0 & 0 & 0 \\ 0 & s_j & 0 & 0 \\ 0 & 0 & s_j & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{8}$$

2.3. *Rotation*

The twisted-profiled contour curve C must be rotated in 3-D with respect to the tangent vector at each point of trajectory curve T. That is, the plane of C must be made perpendicular to the tangent vector of T at each point. We calculate the discrete derivative for each point of T to handle the tangent vector. To this end, we add two dummy vertices, T_0 and T_{m+1}, to T. Here, $T_0 = T_1$ and $T_{m+1} = T_m$. In the following subsections, we use conventional 3-D matrices R_x, R_y, and R_z to handle the position of C, and Tx, Ty, and Tz to represent the x, y, and z distances of the vertices of T.

2.3.1. *Rotation about the x-axis*

$$[R_x] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c\alpha_j & s\alpha_j & 0 \\ 0 & -s\alpha_j & c\alpha_j & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$c\alpha_j \equiv \cos\alpha_j = \frac{Ty_{j-1} - Ty_{j+1}}{h_x}$$

$$s\alpha_j \equiv \sin\alpha_j = \frac{Tz_{j+1} - Tz_{j-1}}{h_x} \tag{9}$$

$$h_x = \sqrt{(Ty_{j-1} - Ty_{j+1})^2 + (Tz_{j-1} - Tz_{j+1})^2} \tag{10}$$

§ The reader will notice that we give detailed derivations of matrix representations. We do this for the sake of completeness; otherwise, we are aware of the fact that homogeneous matrices for translation, rotation, *etc.*, are well known.

2.3.2. *Rotation about the y-axis*

$$[R_y] = \begin{bmatrix} c\beta_j & 0 & -s\beta_j & 0 \\ 0 & 1 & 0 & 0 \\ s\beta_j & 0 & c\beta_j & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$c\beta_j \equiv \cos\beta_j = \frac{Tx_{j-1} - Tx_{j+1}}{h_y}$$

$$s\beta_j \equiv \sin\beta_j = \frac{Tz_{j+1} - Tz_{j-1}}{h_y} \tag{11}$$

$$h_y = \sqrt{(Tx_{j-1} - Tx_{j+1})^2 + (Tz_{j-1} - Tz_{j+1})^2} \tag{12}$$

2.3.3. *Rotation about the z-axis*

$$[R_z] = \begin{bmatrix} c\gamma_j & s\gamma_j & 0 & 0 \\ -s\gamma_j & c\gamma_j & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$c\gamma_j \equiv \cos\gamma_j = \frac{Tx_{j-1} - Tx_{j+1}}{h_z}$$

$$s\gamma_j \equiv \sin\gamma_j = \frac{Ty_{j+1} - Ty_{j-1}}{h_z} \tag{13}$$

$$h_z = \sqrt{(Tx_{j-1} - Tx_{j+1})^2 + (Ty_{j-1} - Ty_{j+1})^2} \tag{14}$$

2.4. *Translation*

Finally, C must be translated to each point of T to finish the sweeping operation. We use a 3-D translation matrix as follows:

$$[T_{xyz}] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ a & b & c & 1 \end{bmatrix} \quad \begin{aligned} a &= Tx_j - Tx_1 \\ b &= Ty_j - Ty_1 \\ c &= Tz_j - Tz_1 \end{aligned} \tag{15}$$

A general twisted-profiled sweeping matrix Sw_{tp} is obtained when the transformation matrices specified above are multiplied in the following order:

$$[Sw_{tp}] = [R_{tw}][S_p][R_x][R_y][R_z][T_{xyz}]. \tag{16}$$

After some manipulations, the following is obtained:

$$[Sw_{tp}] = \begin{bmatrix} \begin{matrix} s_j\cos\theta_j c\beta_j c\gamma_j \\ +s_j\sin\theta_j s\alpha_j s\beta_j c\gamma_j \\ -s_j\sin\theta_j c\alpha_j s\gamma_j \end{matrix} & \begin{matrix} s_j\cos\theta_j c\beta_j s\gamma_j \\ +s_j\sin\theta_j s\alpha_j s\beta_j s\gamma_j \\ +s_j\sin\theta_j c\alpha_j c\gamma_j \end{matrix} & \begin{matrix} -s_j\cos\theta_j s\beta_j \\ +s_j\sin\theta_j s\alpha_j c\beta_j \end{matrix} & 0 \\ \begin{matrix} -s_j\sin\theta_j c\beta_j c\gamma_j \\ +s_j\cos\theta_j s\alpha_j s\beta_j c\gamma_j \\ -s_j\cos\theta_j c\alpha_j s\gamma_j \end{matrix} & \begin{matrix} -s_j\sin\theta_j c\beta_j c\gamma_j \\ +s_j\cos\theta_j s\alpha_j s\beta_j s\gamma_j \\ +s_j\cos\theta_j c\alpha_j c\gamma_j \end{matrix} & \begin{matrix} s_j\sin\theta_j s\beta_j \\ +s_j\cos\theta_j s\alpha_j c\beta_j \end{matrix} & 0 \\ \begin{matrix} s_j c\alpha_j s\beta_j c\gamma_j \\ +s_j s\alpha_j s\gamma_j \end{matrix} & \begin{matrix} s_j c\alpha_j s\beta_j s\gamma_j \\ -s_j s\alpha_j c\gamma_j \end{matrix} & s_j c\alpha_j c\beta_j & 0 \\ a & b & c & 1 \end{bmatrix}. \tag{17}$$

Consequently, a twisted-profiled sweep surface can be calculated in the following form. First, each point of C must be evaluated for matrix Sw_{tp}, and then, a new matrix Sw_{tp} must be modified for each incremented j.

$$[Sweep\ surface]_{i,j,\theta,s} = [C]_i[Sw_{tp}]_{j,\theta,s} \qquad 1 \le i \le n, \quad 1 \le j \le m \quad (18)$$

In the above formula, j varies slowly, *i.e.*, we compute the formula for a particular j and every i.

For the planar case, if C is a 2-D curve or a polygon centered at the origin in the x-y plane and T is a curve or polygon starting at the origin in the x-z plane, then a planar twisted-profiled sweep surface can be computed with the help of the following matrix:

$$[Sw_{ptp}] = [R_{tw}][S_p][R_y][T_{xz}]. \qquad (19)$$

After some manipulations, the following is obtained:

$$[Sw_{ptp}] = \begin{bmatrix} s_j\cos\theta_j c\beta_j & s_j\sin\theta_j & -s_j\cos\theta_j s\beta_j & 0 \\ -s_j\sin\theta_j c\beta_j & s_j\cos\theta_j & s_j\sin\theta_j s\beta_j & 0 \\ s_j s\beta_j & 0 & s_j c\beta_j & 0 \\ a & 0 & b & 1 \end{bmatrix}. \qquad (20)$$

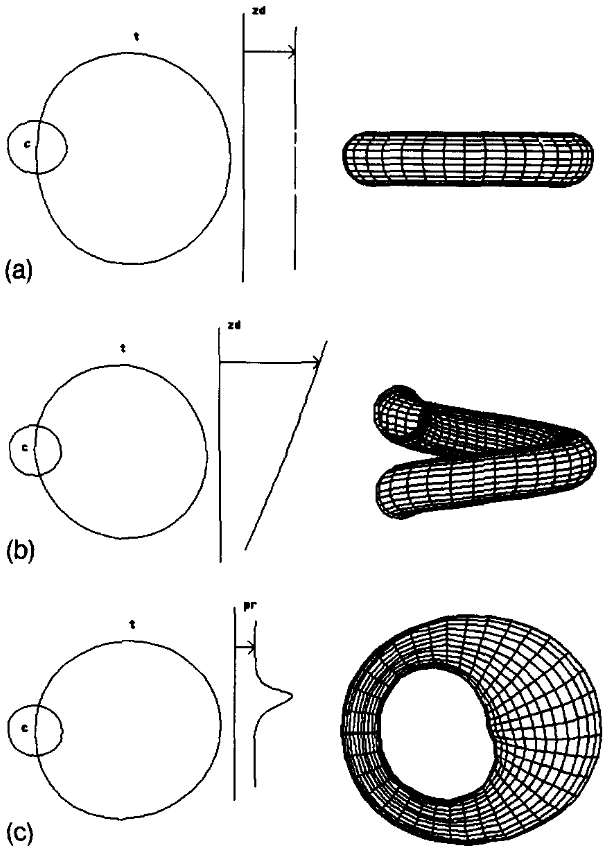

Fig. 1. Contour, (c), trajectory (t), and depth-modulation (zd) curves and produced objects. In (a), the distance of the points of zd to the axis is constant and the resultant object is a torus; in (b), the distance to the axis is varying and the resultant object is nonplanar. Finally, (c) shows the effect of a profile curve (pr).

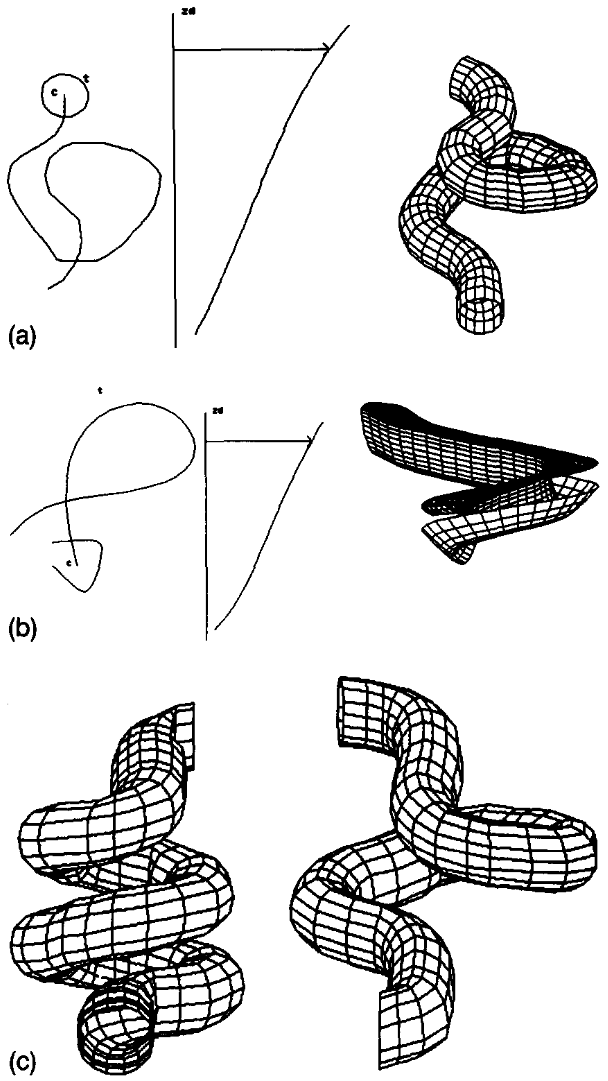

Fig. 2. Some depth-modulated objects produced by Tb. Note that having a closed [as in (a) and (c)] or open contour curve [as in (b)] does not really matter.

3. IMPLEMENTATION AND EXAMPLES

We will present two interactive methods to produce swept shapes. For each method, curves are specified and drawn interactively by the help of say, a mouse. They can be produced in two ways: *free form* and *approximated form*. In the former, a curve is produced by entering points of the curve by the mouse. There is no further operation to smooth the curve. In the latter, a curve is produced by entering each control point of the curve by the mouse. In this case, the curve is approximated by the help of Bézier or B-spline [9, 10, 17] algorithms and hence, the produced curve is smooth.

Our first interactive method is *nonplanar general sweep*. An important aspect of the method is to enter the 3-D points by the help of a mouse. This is provided by a curve, called a *depth modulation curve*. Inputs to the procedure that accomplishes this task are three curves: a *contour*, a *trajectory*, and a *depth modulation* curve. The depth modulation curve gives a "nonplanarity" to the trajectory and now, the trajectory is exactly a 3-D curve. The output of the procedure is a 3-D (nonplanar or planar) object obtained by sweeping the

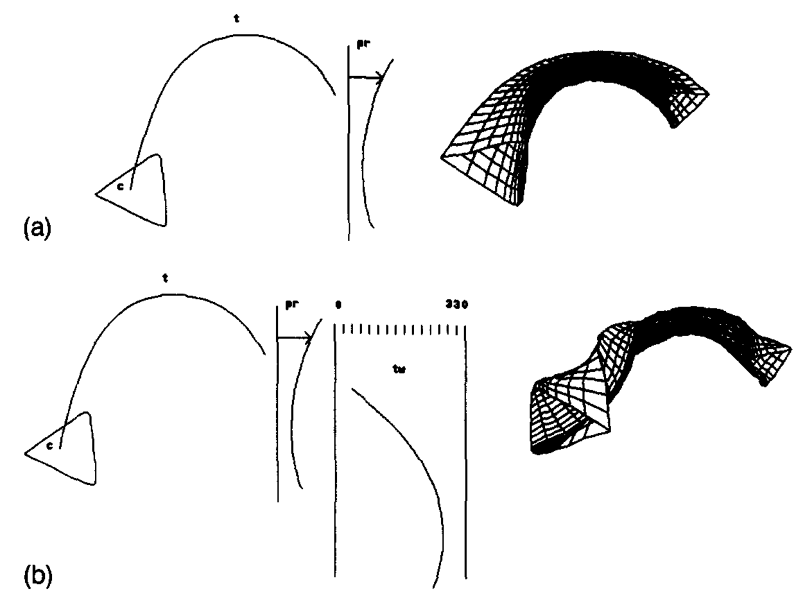

Fig. 3. Contour (c), trajectory (t), profile (pr), and twist (tw) curves and the produced objects. The object in (a) is a profiled object whereas the object in (b) is a twisted-profiled object.

contour curve along the trajectory (Fig. 1). The procedure helps one obtain quite interesting shapes with little effort. It gives flexibility to the user in generating complex shapes, *e.g.*, knots, braids, springs, *etc.* Some samples produced by Tb are shown in Fig. 2.

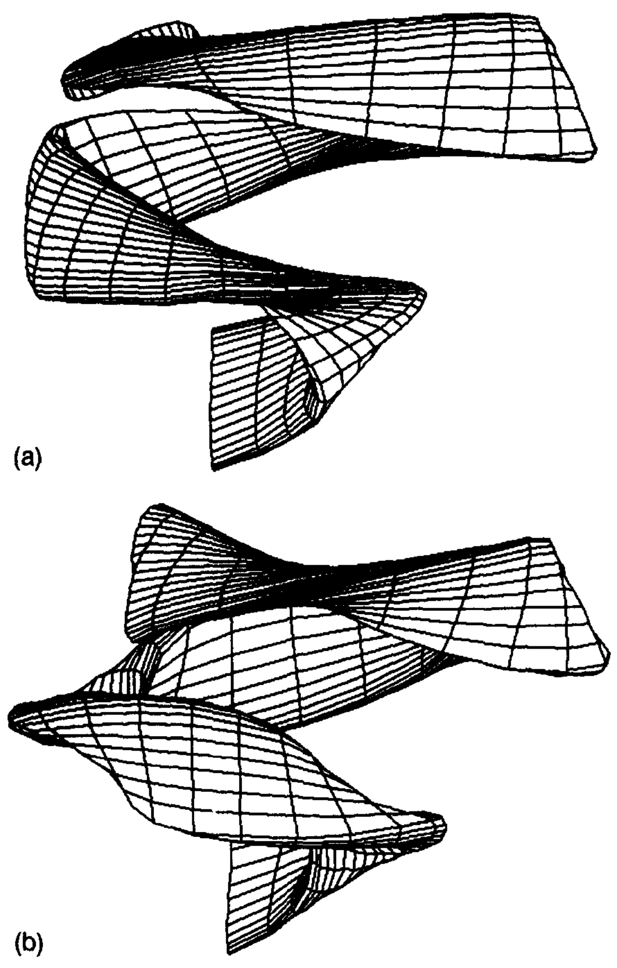

Fig. 4. Sample twisted-profile objects [(a) and (b)] produced by Tb.

The second method is *twisted-profiled nonplanar general sweep.* Five curves are necessary for this method: a contour, a trajectory, a depth modulation curve, a profile, and a twisting curve. The contour curve is twisted and scaled to different angles and sizes according to the profile curve, while sweeping is performed (Fig. 3). This procedure helps one obtain more interesting shapes with little effort. It gives more flexibility to the user in generating even more complex shapes (Fig. 4).

4. CONCLUSION

We presented a mathematical definition of twisted-profiled general sweep objects. Then, we gave two methods to handle sweep objects. The first method discussed how to produce exact 3-D shapes. The second one gave an interactive approach for deformed sweep objects. Deformations include twisting and tapering. We used two curves for these deformations: Twisting curve and profile curve. Many complex sweep objects can be generated by the help of the above methods. Future work for sweeping must be concentrated on fast local deformations. Local deformations for a sweep object are presented by Woodward[27]. He uses the B-spline surface generating method and sweeps control points. The required deformed shape is generated by locally deforming control points. His method can be easily adapted to our needs, but the deformation of control points may be somewhat difficult in interactive systems.

Acknowledgements—We thank the Editor-in-Chief of *Computers & Graphics* and anonymous referees of the journal for their invaluable contributions to the form and content of this paper. As usual, we are solely responsible for any remaining errors.

REFERENCES

1. H. Abelson and A. diSessa. *Turtle Geometry: The Computer as a Medium for Exploring Mathematics*. MIT Press, Cambridge, MA (1982).

2. V. Akman. *Steps into a Geometer's Workbench*, Technical Report CS-R8726, Center for Mathematics and Computer Science, Amsterdam, Netherlands (1987).
3. V. Akman. *Unobstructed Shortest Paths in Polyhedral Environments*, Lecture Notes in Computer Science, Vol. 251, Springer-Verlag, Berlin (1987).
4. V. Akman, A. Arslan, and W. R. Franklin. Excursions to a topological picturebook. In H. P. Santo (Ed.). *Proceedings of 1st International Conference on Computational Graphics and Visualization Techniques*, Vol. 2, Sesimbra, Portugal, 344-361 (1991).
5. V. Akman, A. Arslan, W. R. Franklin, and P. J. W. ten Hagen. Implementing a topological picturebook. In *Proceedings of 13th IMACS World Cóngress on Computation and Applied Mathematics*, Dublin, Ireland (1991).
6. A. Arslan. *An Electronic Topological Picturebook*, Ph.D. Proposal, Department of Computer Engineering and Information Science, Bilkent University, Ankara, Turkey (1990).
7. A. Arslan and V. Akman. An electronic topological picturebook. In *Preliminary Proceedings of NATO Advanced Study Institute on Cognitive and Linguistic Aspects of Geographic Space*, Las Navas del Marques, Spain (1990).
8. A. Arslan, V. İşler, and V. Akman. A procedure to sweep arbitrary curves. In A. E. Harmanci and E. Gelenbe (Eds.). *Proceedings of 5th International Symposium on Computer and Information Sciences*. Vol. 2, Cappadocia, Turkey, 895-904 (1990).
9. B. A. Barsky. A description and evaluation of various 3-D models. *IEEE Computer Graphics and Applications* 4(1), 38-52 (1984).
10. R. H. Bartels, J. C. Beatty, and B. A. Barsky. *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann, Los Altos, CA (1988).
11. B. G. Baumgart. *GEOMED: Geometric Editor*, Technical Report STAN-CS-74-414, Computer Science Department, Stanford University, Stanford, CA (1974).
12. W. F. Bronsvoort and F. Klok. Ray tracing generalized cylinders. *ACM Transaction on Graphics* 4(4), 291-303 (1985).
13. W. F. Bronsvoort, P. R. V. Nieuwenhuizen, and F. H. Post. Display of profiled sweep objects. *Visual Computer* 5, 147-157 (1989).
14. L. Cardelli. *Building User Interfaces by Direct Manipulation*, Technical Report No. 22, Systems Research Center, Digital Equipment Corporation, Palo Alto, CA (1987).
15. B. K. Choi and C. S. Lee. Sweep surfaces modeling via coordinate transformations and blending. *Computer-Aided Design* 22(2), 87-96 (1990).
16. S. Coquillart. A control-point-based sweeping technique. *IEEE Computer Graphics and Applications* 7(10), 36-45 (1987).
17. D. R. Forsey and R. H. Bartels. Hierarchical B-spline refinement. *ACM Computer Graphics* 22(4), 205-211 (1988).
18. G. K. Francis. *A Topological Picturebook*, Springer-Verlag, Berlin (1987).
19. F. Klok. Two moving coordinate frames for sweeping along a 3-D trajectory. *Computer-Aided Geometric Design*. Vol. 3, 217-229 (1986).
20. D. Kneale, Shaping ideas: A topologist wows the world of math by seeing the unseen. *Wall Street Journal* (March 18, 1983).
21. R. R. Martin and P. C. Stepson. Sweeping of three-dimensional objects. *Computer-Aided Design* 22(4), 223-234 (1990).
22. A. P. Pentland, SUPERSKETCH™ *User Manual*, Artificial Intelligence Center, SRI International, Menlo Park, CA (1985).
23. A. P. Pentland. *Perceptual Organization and the Representation of Natural Form*, Technical Report No. 357, Artifical Intelligence Center, SRI International, Menlo Park, CA (1985).
24. F. H. Post and F. Klok. Deformations of sweep objects in solid modeling. In A. A. G. Requicha (Ed.). *Eurographics '86 Proceedings*. North-Holland, Amsterdam, 103-114 (1986).
25. D. F. Rogers and J. A. Adams. *Mathematical Elements for Computer Graphics*, McGraw-Hill, New York (1989).
26. W. P. Wang and K. K. Wang. Geometric modeling for swept volume of moving solids. *IEEE Computer Graphics and Applications* 6(12), 8-17 (1986).
27. C. D. Woodward. Methods for cross-sectional design of B-spline surfaces. In A. A. G. Requicha (Ed.). *Eurographics '86 Proceedings*, North-Holland, Amsterdam, 129-142 (1986).
28. C. D. Woodward. Skinning techniques for iteractive B-spline surface interpolation. *Computer-Aided Design* 20(8), 441-451 (1988).

APPENDIX—REVIEW OF Tb

Several methods exist for generating 3-D models of real or imaginary objects on a computer. The most popular approach is to use polygons as low-level primitives that define more complex objects. Obviously, it is difficult and time-consuming for a designer to define an object by such simple, low-level primitives. Therefore, a higher level primitive such as a B-spline (or a Bézier) surface is preferable. Such parametric patches require many computations when it comes to rendering, although they can be represented in more compact form.

In this Appendix, we discuss the desirable functionalities of an interactive graphics system to visualize the complicated shapes of geometry, topology, and knot theory. Essentially our Tb is a rudimentary graphical workbench to help topologists illustrate their ideas more effectively. Central to our implementation is a paradigm of solid modeling, *viz.*, *shape = sweep + control*. Oddly enough, this powerful paradigm has not been fully utilized in a system-wide manner in the past. That is, programs have been implemented to make use of sweeping in one way or another, but no effort has been concentrated to make sweeping the "underlying" primitive for *everything* in a modeler. This is precisely what we are trying to do with Tb.

A preliminary version of Tb was written in the C programming language and runs on a color Sun workstation. An early yet elegant work which inspired us is Baumgart's GEOMED[11]. Another more recent approach is Pentland's SUPERSKETCH™, which we regard as a very important system[22, 23].

Still, the real influence behind our software owes its existence to *A Topological Picturebook* of George K. Francis—a book that was written to encourage mathematicians to illustrate their work and to help artists to understand the abstract ideas expressed by such drawings[18]. Here we are running the risk of oversimplifying Francis' work considerably, for we cannot yet match the quality and the complexity of the figures in that book. For example, complicated constructions, such as the *tetrahedral eight knot*—as it appears in Bill Thurston's acclaimed work (*cf.* [20] for a popular account) in 3-manifolds—are not yet doable within our system. Our ultimate goal is to produce such figures without undue emotional trauma.[¶]

A topological picturebook may sound somewhat futile vis-à-vis the fact that sketching and visual presentation of topological constructions/proofs are slowly losing their stronghold they once held. It may appear that the science of the "deformation of shapes" is becoming somewhat sterile in terms of

[¶] We cannot unfortunately reproduce Francis' drawings of the tetrahedral eight knot (and many other complicated constructions) here for copyright reasons, *cf.* [18]. This is a pity because a glimpse of his drawings would give the reader a very precise idea of what we have set ourselves to do.

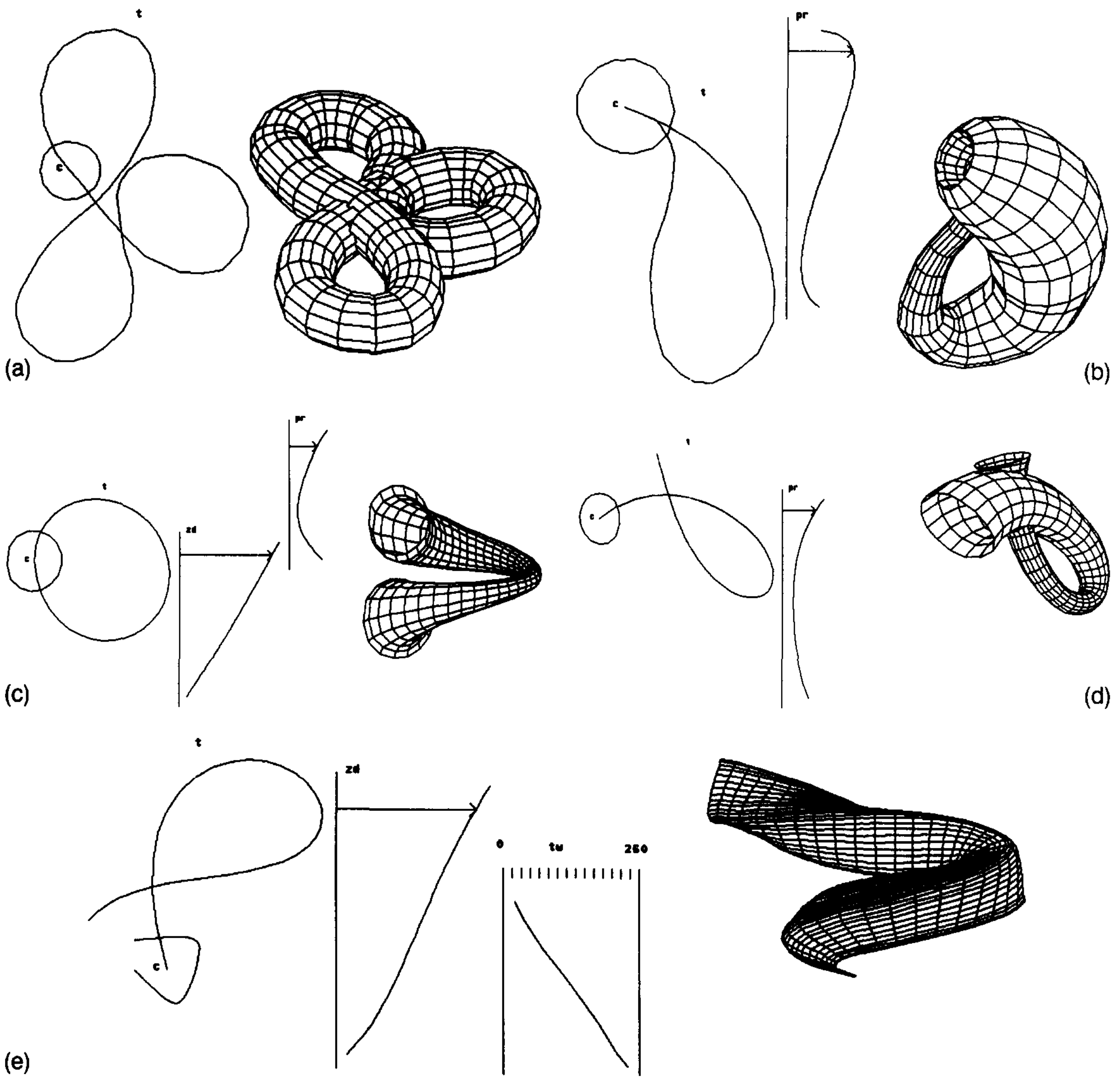

Fig. 5. Sample figures [(a), (b), (c), (d), and (e)] depicting various topological shapes obtained by Tb. The reader will notice that the program can achieve some nontrivial effects in order to simulate self-interesting objects.

figures. However, we believe that there is a definite place for a topological picturebook of the sort to be described here. The following excerpt succinctly supports our view[18].

"The pedagogy that underlies the entire book, and which I bring out specially here, comes from Bernard Morin of Strasbourg. Pictures without formulas mislead, formulas without pictures confuse. I don't know if Bernard would say it this way, but it is how I have understood his work. [· · ·] Morin's vivid, pictorial description of his bold constructions has inspired their realization in many a drawing, model, computer graphic and film. But he insists that ultimately, pictorial descriptions should also be clothed in the analytical garb of traditional mathematics."

A.1. *Realistic views*

There are assorted techniques that can be used to give a realistic view of an object as a 2-D image. The following are some important ones taken from Francis[18]:

1. "For complicated objects it is often impossible to find a view which does not hide some important structure behind a surface sheet. One remedy is to remove a regular patch from the object, creating a transparent *window* through which this structure can be seen in the picture."
2. "To distinguish an edge whose other face is hidden from an edge which merely separates two visible faces, drafting teachers recommend heavier lines for the former relative to the latter."
3. "The shading technique I use makes no pretense of accuracy and realism. It merely encodes positional information and helps distinguish rounded contours from sharp borders. It is based on a few optical principles."
4. "Line patterns based on boxes in affine projection suffer from the Necker cube illusion: 'Which is front and which is back?' Thickening the facing borders helps decide which view is intended."
5. "I prefer to shade a picture by means of a grid of parallel curves on the surface."
6. "Expert drawing teachers such as Kimon Nicolaides rely on practical rules which show only qualitative respect for the laws of geometrical photometry."

We are taking all these issues into account in our system.

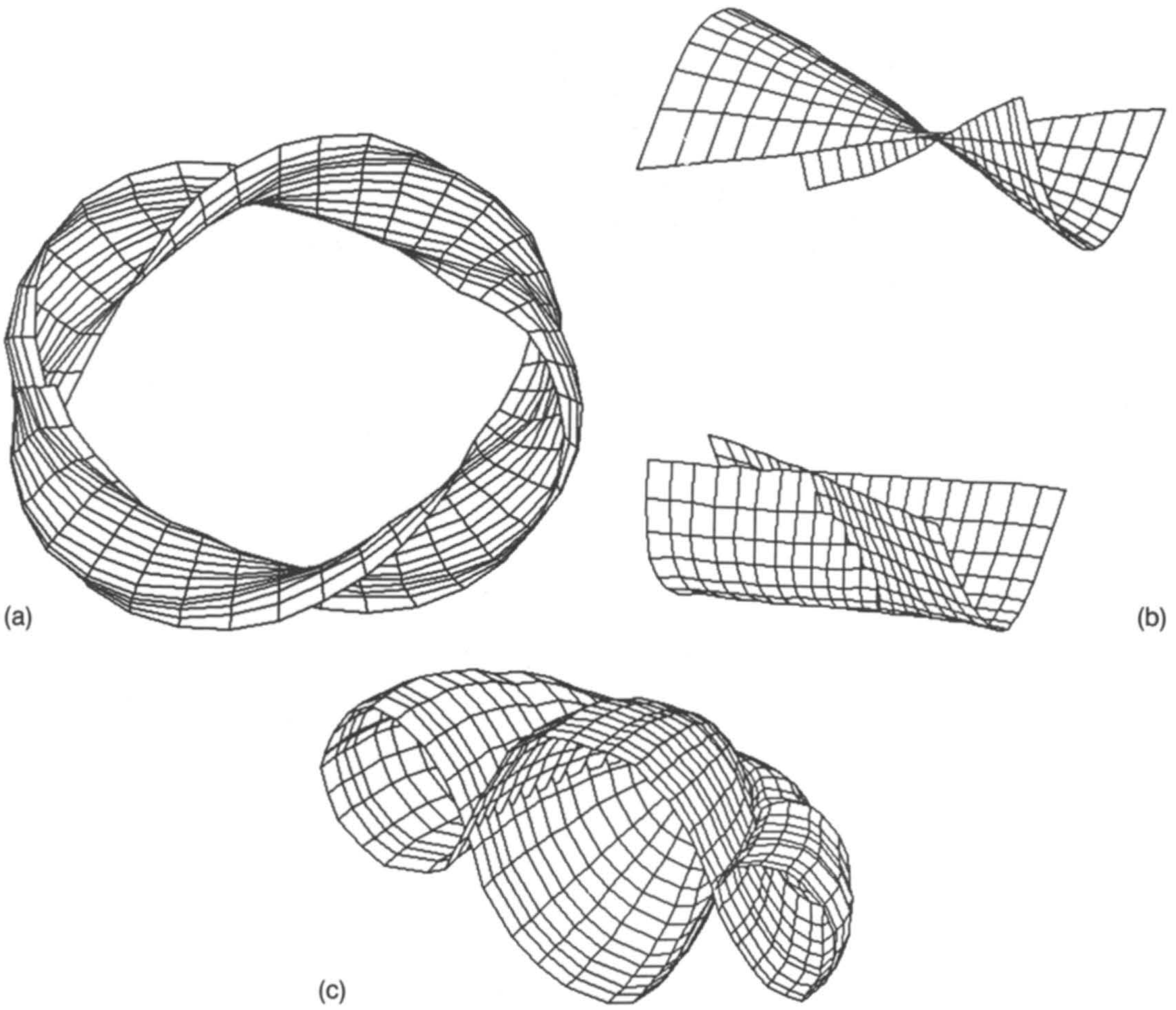

Fig. 6. Further samples [(a), (b), and (c)] treating self-interesting objects.

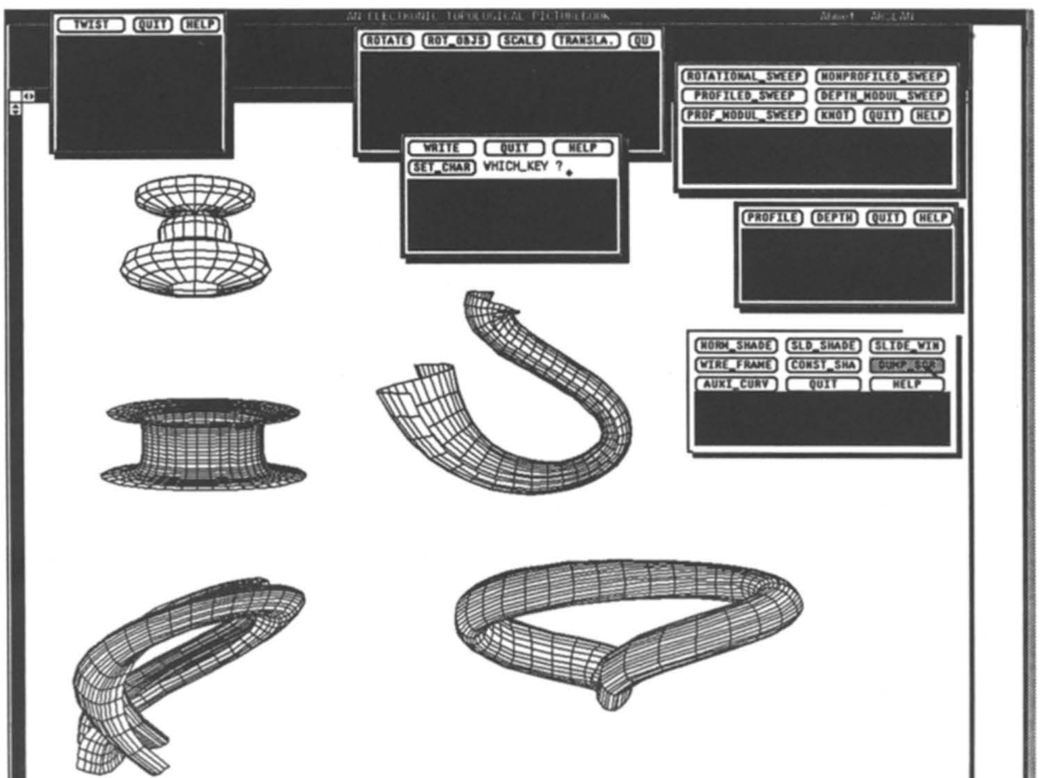

Fig. 7. Screen dump showing Tb in action. The black boxes under the menus are artifacts caused by color to black-and-white conversion.

A.2. *User interface*

Computer systems based on workstations mostly have their own user interface management systems. Such a system presents a set of user interface components to programmers. In general, these components are called *interactors* (windows, menus, scroll bars, buttons, text areas, mouse, and so on). A programmer must only perform a suitable organization of the interactors. In other words, a programmer must build a user interface system by using events, states, or attributes of the supplied interactors.

Tb has a useful interface system based on SUNVIEW (Fig. 7). Of course, the main goal of a user interface system must be to provide the above requirements. But Tb will be a professional's program and sometimes, speed is preferred over comprehensibility and orthogonality. Tb is based on the following guidelines:

1. Interactors with a main purpose are shown on the screen all the time.
2. Excluding text input, only the mouse is used for interactions.
3. Interactors do not occupy too much space on the screen.
4. Interactors are well placed.
5. It is made sure that interactors have not too many items.
6. Interactors do not overlap with each other.
7. Interactors are not shown on the screen when they are not necessary.
8. Mouse buttons are preferred over menus.
9. Help menus are available.

Our software can be regarded as a first order approximation to a good mathematical picture modeler. The interaction of Tb with the user is straightforward. For example, a torus can be rendered by selecting only five control points to produce the contour and a radius. Our fundamental aim is to hide details of processing from the user. In other words, the user will design objects as if he is doing free-form sculpturing, by carrying out high level operations. This is in the precise spirit of Pentland's sketching system[22, 23] mentioned earlier.

The main goal of our work has been to provide a graphical workbench to assist topologists and artists to illustrate their ideas more effectively. An auxiliary goal is to improve the publication-quality production of mathematical figures. Future

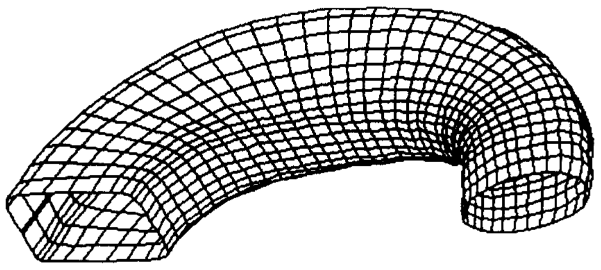
Fig. 8. Blending in Tb.

enhancements that will be added to Tb are as follows (some of these exist—as a first-order approximation—in the present version of Tb):

1. Global or local *twisting* of sweep objects; *cf.* Fig. 4 and Fig. 6.
2. *Puncturing*—While topology is intimately related to deformations, there are points where one would need tools to puncture or to cut-and-paste surfaces. For example, in order to build a 2-tori, one might choose to start with two separate tori. Then, a small disc is punctured on each torus. If one matches these discs (*i.e.*, glues the tori at these holes), then one obtains a 2-tori.
3. *Blending*—This is the well-known problem of say, how to smoothly glue the two tori of the previous example so that the resulting 2-tori has a fine appearance; *cf.*, Fig. 8.
4. Incorporating text and formulas in various fonts. (Interface with $\TeX$ is being considered.)
5. Local deformations of sweep objects and *surgery*. For example, we would like to create a small "bump" or a tiny "well" on a given surface.
6. Shading capabilities. Transparency to inspect the sublayers of an object.
7. Anti-aliasing. This is clearly necessary for publication-quality pictures.
8. *Shadows*—Francis is not really interested in the problem of depicting shadows cast by one object on another (although he notes that the shape of a shadow still obeys the objective laws of linear perspective)[18].