Implicit Generalized Cylinders using Profile Curves

Cindy M. Grimm

cmg@cs.wustl.edu

Abstract

We introduce an implicit generalized cylinder which is constructed from an axis and one or more profile curves. This surface is related to the *sweep* which is traditionally formed by an axis curve and one or more cross sections. Instead of cross sections, this definition uses profile curves to define how far the surface is from the axis. This facilitates the construction of surfaces which have continually varying cross sections.

We extend this definition to a generalized cylinder with two axis curves. This model is useful for surfaces where one side of the surface curves sharply away from the axis while the axis itself is curving.

We also present a user interface for editing these sweeps.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling, Curve, Surface, Solid, and Object Representations, Splines

Additional Keywords: Isosurface, sweep, user interface

1 Introduction

We introduce an implicit generalized cylinder, which we call a *worm*, which is constructed from an axis and one or more profile curves. The worm is related to the *sweep*, which is traditionally formed by an axis curve and one or more cross sections which are “swept” along the axis curve. If the desired surface has a constantly varying cross section, this definition becomes unwieldy. Instead of specifying cross sections, the worm uses profile curves. Each profile curve is defined for the length of the axis and determines how far the surface is from the axis for a particular angle.

The types of surfaces appropriate for worms are ones which are fundamentally a cylinder, but have constantly varying cross sections, or profiles. Specifying profile curves instead of cross sections provides more control over how the cross sections vary, but less control over the cross sections themselves.

Both sweeps and worms have difficulty modeling surfaces where the axis is curving when the profile curve wants to expand (see Figure 11). This is because the cross section is oriented away from the desired area of expansion because of the curvature of the axis. To address this problem we extend the worm definition by replacing the axis curve with a ruled surface formed by two axis curves. This extended definition is given in Section 3.4 and produces surfaces such as the one in Figure 11.

The implicit function we construct is essentially the distance from the axis curve, minus the distance defined by the profile curve at that point and angle. The level zero surface, therefore, is approximately the same as the parametric worm surface. Notice, however, that the distance function defined in this way will have discontinuities at points which are equidistant from two points on the axis curve. For reasonable axis and profile curves these points will not affect the level zero surface. We examine how such discontinuities affect the surface in Section 5.

To make this primitive useful, we also show a simple system for directly manipulating the profile and axis curves (see Section 4). The interactive system uses a parametric approximation, defined in Section 3.3, to provide feedback.

2 Previous work

Constructing a parametric surface by sweeping a 2D curve along an axis is not new [Far88][BR91]. The traditional definition using cross sections is most suited to CAD/CAM modeling while the profile curve definition [KPL94] is more suited to free-form modeling where the cross sections are constantly changing.

One approach to defining an implicit sweep is to sweep an implicit solid along an axis [LE93] [EGS93]. This representation is difficult to use for free-form objects for the same reason that the parametric cross section representation is; figuring out what solid will give the desired silhouette is not always obvious. Additionally, the choice of solid to sweep and choice of scaling functions currently available is limited.

There are several implicit primitives which use the notion of an axis and a distance function. The sweep definition defined here is closely related to the one defined in [CBS96], which is a sweep of an anisotropic distance function [BS95]. The main difference between their approach and ours is the use of profile curves instead of cross sections; this allows us to have a constantly varying profile at the expense of detail in the cross section. Computation costs increase depending upon the number of profile curves because the n profile curves must be evaluated before evaluating the constructed cross section.

3 The definition of a worm

We first describe the components of the worm which are common to both the implicit and parametric definitions. Next, we define the implicit function. This is followed by a description of how to generate the parametric surface. Finally, we describe how to extend this definition to two axis curves.

3.1 The components of a worm

The basic shape of the worm is defined by the axis curve. At each point of the axis we define a frame consisting of three orthogonal vectors which are based on the tangent and normal vectors [1]. This frame is used as a local coordinate system for each point of the curve and has the property that the $x-y$ plane is perpendicular to the tangent of the curve at the point (see Figure 1). The cross section curve for a given point on the curve is constructed in this plane using the N profile curves.

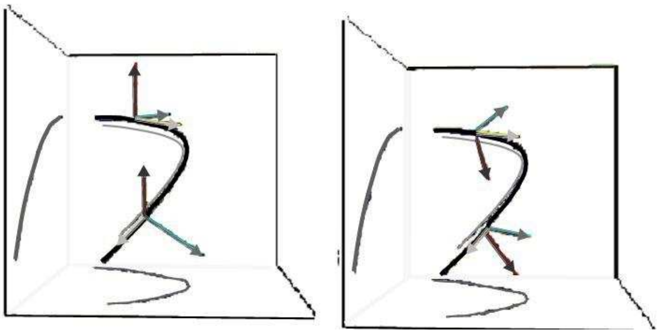

Figure 1: The axis curve with two frames marked. The normal vector is in red, the binormal in blue. Left: The normalized frame. Right: The original frame. Note that the y vector of the frames on the left points roughly up.

We now define the axis curve, the profile curves, and how to construct a cross section curve from the profile curves for a given point on the axis curve.

Let $A(s) : [0,1] \rightarrow \Re^3$ be the axis curve, defined as a B-spline with n control points, i.e.,

$$A(s) = \sum_{i=0}^{n-1} b_i(s) g_i$$

where the b_i are order 2 (or higher) B-spline basis functions. Then the tangent, normal, and bi-normal vectors at s are:

$$\begin{aligned} T(s) &= \frac{dA(s)/ds}{||dA(s)/ds||} \\ N(s) &= \frac{d^2A(s)/ds}{||d^2A(s)/ds||} \\ B(s) &= T(s) \times N(s) \end{aligned}$$

This frame can “swing” around the curve as the axis changes. It can be used in this form, but we normalize so that N' points in roughly the same direction at every point. This is possible if there exists an up vector $\vec{U}$ such that the tangent never points in the direction of the up vector. To normalize the frame,

$$\begin{aligned} T'(s) &= T(s) \\ x &= N(s) \cdot \vec{U} \\ y &= B(s) \cdot \vec{U} \\ N'(s) &= \frac{N(s)x+B(s)y}{||N(s)x+B(s)y||} \\ B'(s) &= \frac{B(s)x-N(s)y}{||B(s)x-N(s)y||} \end{aligned}$$

[1] We rotate the bi-normal and normal vectors of the frame so that the normal always points in a chosen direction, if possible. This keeps the profiles from “spinning” around the axis.

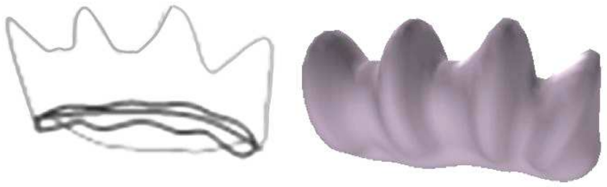

Figure 2: Left: The axis curve and the 4 profile curves around it for the top fin. Right: The resulting parametric approximation.

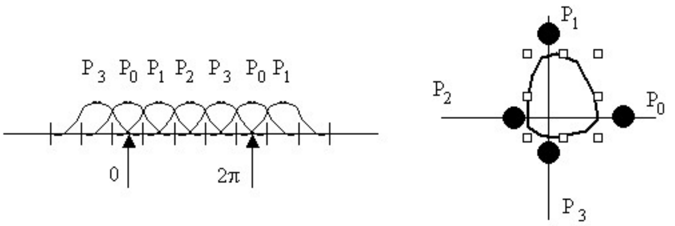

Figure 3: Left: The knot vector for the cross section curve. Each basis function is paired with a control point produced by evaluating the corresponding profile curve. Right: Drawing the cross section in polar coordinates.

The distance from the axis curve to the level zero surface is defined using a set of N profile curves, where $N > 0$. Each profile curve is a 1D curve with the same parameter space as the axis curve, i.e.,

$$P_i : [0, 1] \to \Re$$

and is guaranteed to be always positive and descending to 0 at both ends (this "closes off" the end of the worm). We use profile curves of order 1 and varying numbers of control points. To ensure the above conditions, all control points must be positive, except for the first and last points, which are set to the negative of the second and second to last control points, respectively. Refer to Figure 2 for an example of an axis curve with 4 profile curves.

At each point $A(s)$ along the axis curve we can define a cross section curve based on the frame and profile curves evaluated at s. An illustration of how the cross section curve is constructed from the profile curves is shown in Figure 3. The cross section curve is a closed curve which lies entirely in the plane defined by $N(s)$ and $B(s)$. Ideally, the cross section curve should be a periodic curve mapping $[0, 2\pi]$ into $\Re$ whose control points are the evaluated profile curves. We simulate this curve using a spline curve whose start and finish lines up. This is accomplished by duplicating control points at the beginning and end of the curve

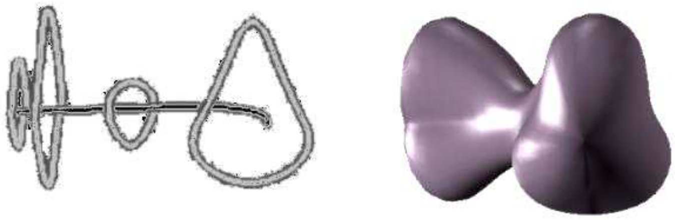

Figure 4: The axis curve and several different cross section curves of the fish body.

Let $D_s : [0, 2\pi] \to \Re$ be the cross section curve, which is an order 1 curve with $N + 3$ control points and the following knot vector:

$$k_i = (i - 2)\Delta \text{ with } \Delta = (2\pi)/N$$

We set the control points D_s^i to:

$$D_s^i = P_{(i-1) \bmod N}(s)$$

This ensures that D_s is most influenced by the 0^{th} profile curve at $\theta = 0$ and again at $\theta = 2\pi$. Note that $D_s(0) = D_s(2\pi)$. Examples of cross section curves, drawn in the appropriate frames, are shown in Figure 4.

3.2 Implicit definition

The implicit function $I : \Re^3 \to \Re$ is positive at the axis and decreases moving away, reaching 0 at a distance specified by the profile curves. I is calculated as follows: Let $p \in \Re^3$ be a point in space. Project p onto the axis, giving $p' = A(s)$. To determine θ, project $p - p'$ onto the frame at s:

$$\begin{aligned} x &= B'(s) \cdot (p - p') \\ y &= N'(s) \cdot (p - p') \\ \theta &= \arctan(\tfrac{y}{-x}) + \pi \end{aligned}$$

The level zero surface is at $D_s(\theta)$, so

$$I(p) = D_s(\theta) - ||p - p'||$$

Because the profile curves go to zero at the ends of the axis curve the ends of the worm are well-behaved.

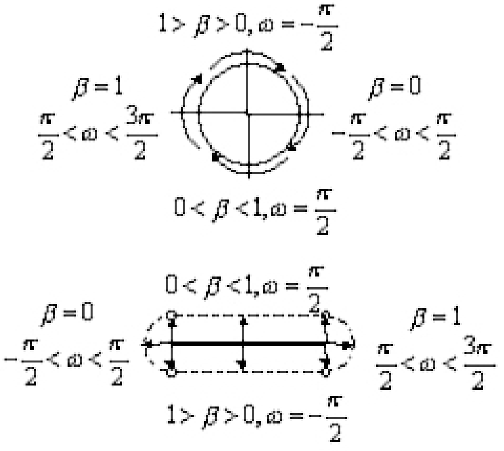

Figure 6: Top: How β and ω map to θ in the cross section curve. Bottom: Where β and ω lie on a slice of the ruled surface.

3.3 Parametric definition

To create a parametric approximation to the implicit surface, evaluate the axis and cross section at evenly spaced points.

$$P(s,\theta) = A(s) + \big(sin(\theta)B'(s) + cos(\theta)N'(s)\big)D_s(\theta)$$

The surface is essentially a tube of varying widths, and may intersect itself. Note that this surface is only an approximation because $P(s,\theta)$ projected onto A may not yield $A(s)$, but will generally be close. Examples of the implicit and parametric approximation of several surfaces are shown in Figure 5.

3.4 A worm with two axes

To extend the single axis definition to two axes we replace the axis curve calculation to a projection to a ruled surface and change the parameterization of the cross section curve. Since we are replacing the axis curve with a surface we will also need to change how the frame is calculated at each point. The cross section parameterization must change because the cross section is no longer a circle around a point but an ovoid around a line (see Figure 6). The cross section calculation is now a distance from a line instead of a point. Essentially, we split the circular cross section in half and add a linear segment in the middle.

The axis is a ruled surface defined by the two axis curves.

$$A_s(t,\beta) = (1-\beta)A_0(t) + \beta A_1(t)$$

We use the tangent in the direction of t at β to determine the plane of the cross section curve.

$$\begin{aligned} T(t,\beta) &= dA_s(t,\beta)/dt \\ N(t,\beta) &= dA_s(t,\beta)/d\beta \\ B(t,\beta) &= T(t,\beta) \times N(t,\beta) \end{aligned}$$

Again we normalize the frame, this time using $A_0(t) - A_1(t) \times T(t,\beta)$, projected into the plane defined by $T(t,\beta)$ as the up vector, $\vec{U}$. This keeps $B'(t,\beta)$ pointing away from the plane of the ruled surface.

For the implicit surface function, projecting to the axis now means projecting to the ruled surface A_s. This projection can be approximated by a polygonal mesh constructed by evaluating A_1 and A_2 at regularly spaced intervals (see Figure 11).

The cross section curve is re-parameterized to account for the center of the axis now being a line. Essentially, we break the cross section up into four sections, one for each of the ends ($\beta = 0, 1$, $\omega \in 2\pi$), and one for each side of the ruled surface ($0 < \beta < 1$, $\omega = \pi/2, -\pi/2$) (see Figure 6).

$$D'_s(\beta,\omega) = \begin{cases} D_s(\frac{\omega}{2}) & \beta = 0 & \frac{-\pi}{2} < \omega < \frac{\pi}{2} \\ D_s(\frac{\pi}{4} + \frac{\beta\pi}{2}) & \beta \in [0,1] & \omega = \frac{\pi}{2} \\ D_s(\frac{\pi}{2} + \frac{\omega}{2}) & \beta = 1 & \frac{\pi}{2} < \omega < \frac{3\pi}{2} \\ D_s(\frac{5\pi}{4} + \frac{(1-\beta)\pi}{2} & \beta \in [0,1] & \omega = \frac{-\pi}{2} \end{cases}$$

After projecting to the ruled surface we either are at the edges of the surface, in which case $\beta = 0$ or $\beta = 1$ and we calculate ω from the frame as before, or we are in the middle of the surface and ω is either $\pi/2$ if we are above the surface or $-\pi/2$ if we are below the surface. The remainder of the distance calculation is the same, i.e., the construction of D_s for a particular value of $s \in [0,1]$.

To compute the parametric surface step evenly along D'_s.

4 User interface

Sweeps began as a method for specifying CAD/CAM models which had a specific, possibly varying cross section (such as a chair leg or piston). They are less useful for free-form surfaces because there the cross section is constantly changing and it is the profile of the surface that is of interest. Therefore we use a profile curve representation instead of the more traditional cross section one. Note, though, that individual cross sections can still be edited by changing the appropriate control points of the profile curve at the given value.

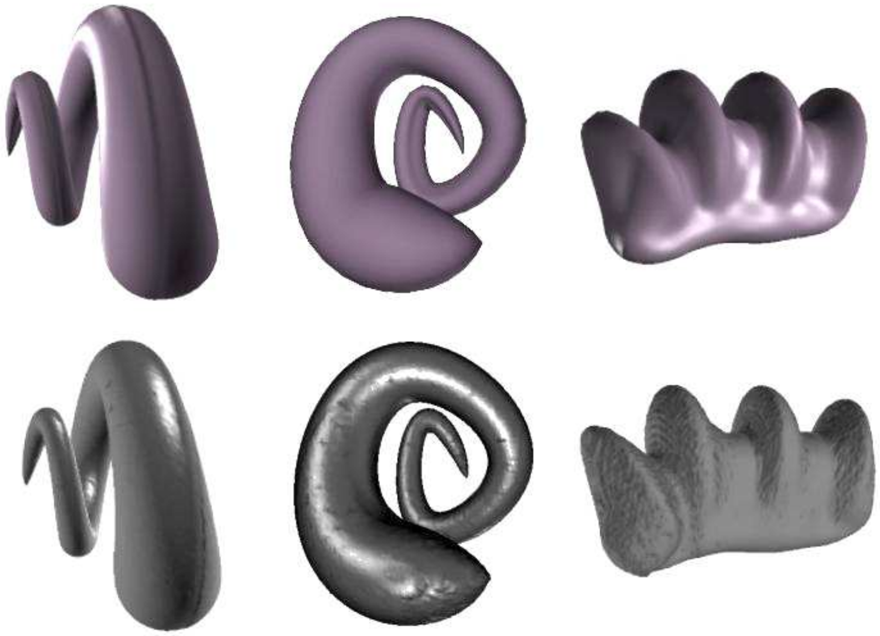

Figure 5: Top: The parametric approximation. Bottom: The implicit surface.

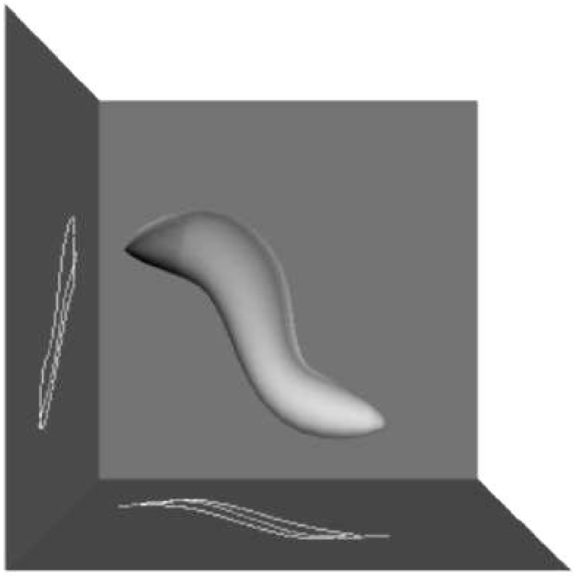

Figure 7: The interactive environment with the three walls and an example worm.

The environment we use for editing the worms is shown in Figure 7. The surface is surrounded on three sides by walls which move as the camera moves to maintain the box corner (the specific angle of the box walls can be adjusted independently). This approach is a more flexible variation of the traditional modeler which uses three orthogonal views down the X, Y, and Z axis. All editing takes place on the walls, in a manner similar to [CMZ$^+$99]. The axis and the two profile curves which are most "up" and "down" are drawn on each of the three walls (which profile curves are drawn will vary from wall to wall). Additionally, the user may click on a wall axis curve and the system will draw the cross section curve for that point. The axis and profile curves are also drawn in 3-space where they appear relative to the surface. The two profile curves which are the upper and lower curves on the back wall are drawn hi-lighted.

To alter the axis curve, the user draws on one of the walls with the left mouse button. These changes are applied to the two appropriate dimensions, with the third dimension sampled from the current axis curve, as shown in Figure 8. Note that any 2D curve editing techniques may be applied here, such as moving the control points directly, sketching over [Bau94] [BC90] or direct manipulation [FB91].

To edit the profile curves, the user draws on the wall.

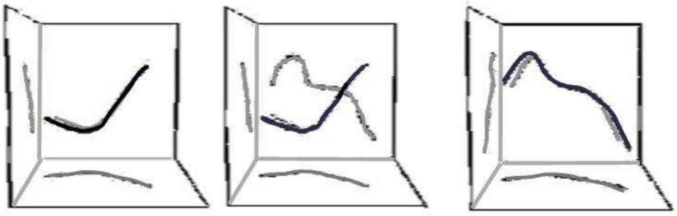

Figure 8: An example of editing the axis. Left: The original axis. Middle: The sketched points. Right: The new axis.

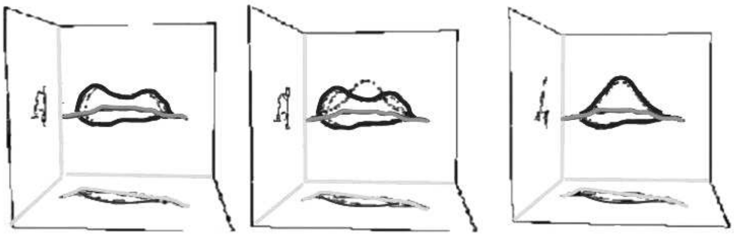

Figure 9: An example of editing a profile curve Left: The projected axis and original projected profiles. Middle: The sketched points. Right: The new projected profile curve.

The current axis and two "up" and "down" profile curves are drawn on the wall to indicate scale. If the user draws above the axis curve, the upper profile curve is changed, otherwise the lower profile curve is chosen. Recall that the profile curve is a 1D curve; we set the value of the profile curve to be the distance from the axis curve to the new points. Consider sketching a new curve (refer to Figure 9). Each of the new points sketched on the wall is projected down to the projected axis curve for that wall. This gives a parameter value and distance value for each of the sketched points. To fit the profile curve to the sketched points we use a variation of the least squares technique [Str88] which finds the location of the control points which best minimizes the squared error (see Appendix A) and provides some filtering for the (generally) noisy input.

4.1 Implementation details

A 3D profile curve is drawn relative to the axis curve by evaluating the parametric equation $P(s,\theta)$ for each control point at a fixed θ. The peak of profile curve i occurs at $\theta_i = i\frac{2\pi}{N-1}$. Since the axis and profile knot vectors are both evenly spaced, we evenly space along the axis curve to determine s for each control point ($s = j\frac{1}{n-1}$). Note that this generally makes the profile curve slightly shorter than the axis curve.

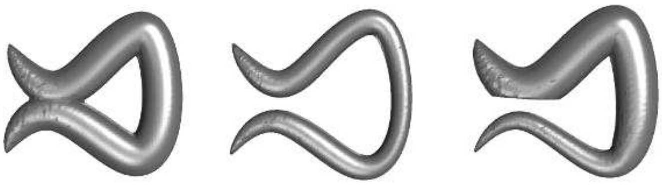

Figure 10: An example of a discontinuity in I. The discontinuities form a horizontal line between the two ends of the worm. Left: The discontinuities lie inside the surface. Middle: Outside the surface. Right: Inside and outside the surface.

To draw the cross section at a point $s \in [0,1]$ we evaluate the parametric equation for the $N+3$ control points of the cross section curve D_s with $\theta_i = i\frac{2\pi}{N}$. (Evaluate the profile curves at s to set the control points for D_s.)

To draw the curves on the wall, we project the control points onto the wall and evaluate the new curve. Recall that the walls rotate around with the camera; if the wall has normal $\vec{u}$, is a distance d_0 along that normal, and the eye point of the camera is at $\vec{p_c}$ then the projected point of $\vec{q}$ is

$$\begin{aligned} s_1 &= \vec{q}\cdot\vec{u} - d_0 \\ s_2 &= \vec{p_c}\cdot\vec{u} - d_0 \\ q' &= \tfrac{s_1}{s1-s2}\vec{q} + (1 - \tfrac{s_1}{s1-s2})\vec{p_c} \end{aligned}$$

5 Discontinuities

In this section we describe the discontinuities that occur in the implicit function I and how this problem manifests itself. Any point $p \in \Re^3$ which is equidistant from one or more points on the axis curve will have a discontinuity. This discontinuity can take three forms; either all values from every point are positive, all values are negative, or some are positive and some are negative. For this example we are interested in the level zero surface.

If all values are negative, then we have the case shown in the middle of Figure 10 where the point p is outside all surfaces. Similarly, if all the values are positive then p is inside the surface (see left of Figure 10) — this is the case where the parametric surface is likely to self-intersect. The interesting case is when the discontinuity goes from positive to negative. In this case, the surface is "clipped" by the profile curve of the negative surface (see right of Figure 10). In reality, the section clipped should be an open hole because on one side the function is positive, and on the other it is negative. It shows as a closed surface because the marching cubes algorithm only looks for positive to negative crossings.

6 Current and future work

There are a variety of possible interaction techniques yet to be explored. Some obvious ones are editing the cross section curve directly in its plane, similar to [GPB+95], fixing the drawn 3D profile curve locally to a vector and pulling on it, or editing the axis curve using any of the techniques discussed in [GA98] or [CMZ+99].

There are some open problems in defining the surface using two axis curve. The most serious is finding a parameterization of the two curves which will yield a ruled surface that does not cross back on itself.

7 Conclusion

We have presented an extension to the traditional axis and distance function which allows for varied profiles and shapes at the cost of introducing discontinuities in the implicit function. We believe there is a set of surfaces which are both interesting and well-behaved that can be modeled using this technique. We also introduced some simple editing techniques to simplify building these surfaces.

8 Acknowledgments

Thanks to Jules Bloomenthal and Paul Heckbert for their implicit surface polygonizer.

A Least squares

A least-squares problem is formulated by

$$Ax = b$$

where each row of A is a constraint on the solution, $x = (AA^T)^{-1}A^Tb$. There must be more constraints than variables, i.e., more rows in A than columns.

Let

$$C : [0,1] \rightarrow \Re^n \text{ s.t. } C(s) = \sum_{i=0}^{n-1} g_i b_i(s)$$

be the order o curve we are fitting, with knot vector $K = \{k_j\}_{0 \leq j < n+o+2}$. For curve fitting, each row of the matrix is of the form:

$$C(s_j) = \sum_{i=0}^{n-1} g_i b_i(s_j) = p_j$$

where we want the curve at $s = s_j$ to pass through the point p_j. The variables in this case are the n control points g_i and the row of the matrix A is the basis functions b_i evaluated at s_j. To ensure that the end points of the curve are correct, we also add the following constraints:

$$\begin{aligned} C(\Delta k_0 + k_{o+1}) &= \Delta p_0 \\ C(\Delta k_n + k_{n+o+1}) &= \Delta p_{\text{last}} \end{aligned}$$

for o Δs evenly spaced between 0 and 1.

If the s_j values for the points are evenly spaced and the knot vector is evenly spaced then it is sufficient to ensure that there are more constraints than control points. Otherwise, care must be taken to ensure that there is at least one constraint given in the support of each basis function.

To find an appropriate fit, we try different numbers of control points for the curves until the error from the curve to the sample points is at a particular threshold.

REFERENCES

[Bau94] T. Baudel. A mark-based interaction paradigm for freehand drawing. *UIST '94 proceedings*, pages 185–192, November 1994.

[BC90] M. J. Banks and E. Cohen. realtime spline curves from interactively sketched data. *Computer Graphics*, 24(2):99–107, August 1990.

[BR91] Mark Bloomenthal and Richard Riesenfeld. Approximation of Sweep Surfaces by Tensor Product NURBS. *Proceedings of the SPIE Conference*, 1610:132–144, November 1991.

[BS95] Carole Blanc and Christophe Schlick. Extended Field Functions for Soft Objects. *Proceedings of the first eurographics workshop on implicit surfaces, Grenoble, Fr*, pages 21–32, April 1995.

[CBS96] Benoît Crespin, Carole Blanc, and Christophe Schlick. Implicit Sweep Objects. *Proc. Eurographics '96*, 15(3):165–174, August 1996.

[CMZ+99] J. M. Cohen, L. Markosian, R. C. Zeleznik, J. F. Hughes, and R. Barzel. An Interface for Sketching 3D Curves. *Proceedings of 1999 Symposium on Interactive 3D Graphics*, April 1999.

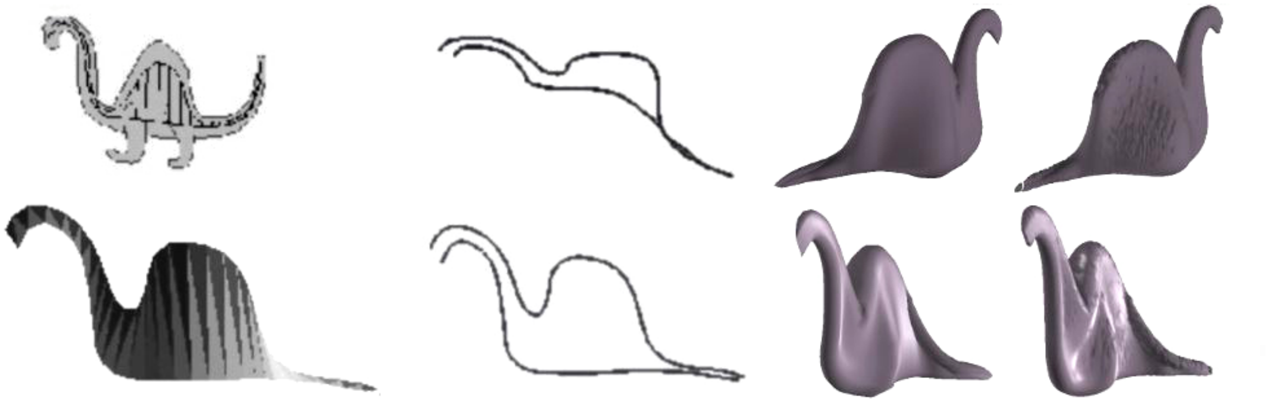

Figure 11: A surface which is difficult to do with a single axis curve, but simple to do with a ruled surface axis. Left: A sketch of this type of surface and the ruled axis constructing it with the actual ruled surface below. Middle left: The two axis curve viewed from above and the side. Middle right: The parametric approximation. Right: The implicit surface. This surface was made with 6 profile curves

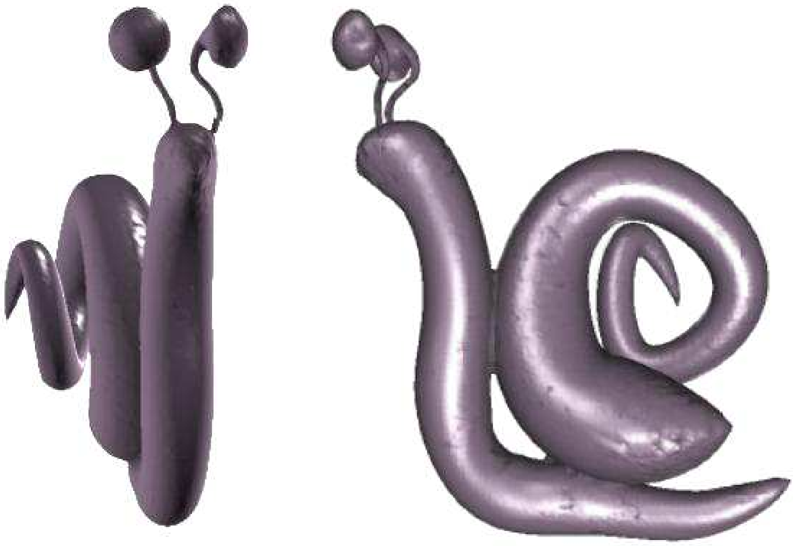

Figure 12: A snail made from 4 worms, one for the body, one for the shell, and two for the eyes.

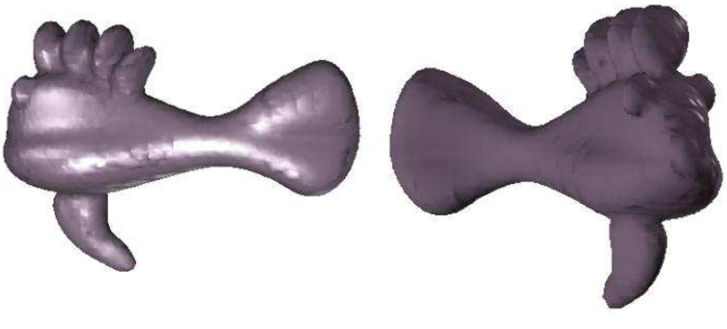

Figure 13: A fish made from 5 worms, body, top fin, bottom fin, and two for the eyes.

[EGS93] Mark Ensz, Mark Ganter, and Duane Storti. Implicit Function Alteration via Radius Mapping and Direct Function Modification. *Unknown*, 1993. www.me.washington.edu/ cdam /PEOPLE/MARK/papers/markpaper/ensz.asme.html.

[Far88] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design*. Academic Press, 1988.

[FB91] B. Fowler and R. H. Bartels. Constraint based curve manipulation. *Siggraph course notes 25*, July 1991.

[GA98] C. Grimm and M. Ayers. A Framework for Synchronized Editing of Multiple Curve Representations. *Eurographics '98*, pages 51–61, September 1998.

[GPB$^+$95] C. Grimm, D. Pugmire, M. Bloomenthal, J. Hughes, and E. Cohen. Visual Interfaces for solids modeling. *UIST '95*, pages 51–61, November 1995.

[KPL94] Myung-Soo Kim, Eun-Joo Park, and Hwan-Yong Lee. Modeling and Animation of Generalized Cylinders with Variable Radius Offset Space Curves. *The Journal of Visualization and Computer Animation*, 5(4):189–207, 1994.

[LE93] Chek Lim and Mark Ensz. Algebraic Computer-Aided-Design with Maple V2. *Proceedings of the Maple Summer Workshop and Symposium, Ann Arbor, Michigan*, 1993. Mathematical Computation with Maple V: Ideas and Applications.

[Str88] Strang. *Linear Algebra and its Application*. HBJ, 1988.