Volume xx (200y), Number z, pp. 1–12

Prototype Modeling from Sketched Silhouettes based on Convolution Surfaces

C.L. Tai [1], H.X. Zhang [2,1] †, C. K. Fong [1]

[1] Department of Computer Science, Hong Kong University of Science & Technology, Kowloon, Hong Kong
[2] State Key Laboratory of CAD&CG, Zhejiang University, P.R. China.

Abstract

This paper presents a hybrid method for creating three-dimensional shapes by sketching silhouette curves. Given a silhouette curve, we approximate its medial axis as a set of line segments, and convolve a linearly weighted kernel along each segment. By summing the fields of all segments, an analytical convolution surface is obtained. The resulting generic shape has circular cross-section, but can be conveniently modified via sketched profile or shape parameters of a spatial transform. New components can be similarly designed by sketching on different projection planes. The convolution surface model lends itself to smooth merging between the overlapping components. Our method overcomes several limitations of previous sketched-based systems, including designing objects of arbitrary genus, objects with semi-sharp features, and the ability to easily generate variants of shapes.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling; I.3.6 [Computer Graphics]: Interaction techniques

1. Introduction

In this paper, we present a method for designing prototype models from sketched silhouette curves based on a new enhanced convolution surface model. Prototype design aims at fast and intuitive creation of 3D shapes. In the conceptual design stage, prototype is designed to test and consolidate ideas before actual models are created for computer graphics and CAD applications. Existing 3D design software packages mainly provide powerful precise shape controls, but they are not always suitable for conceptual stage of design.

Despite today's highly computerized world, design artists still love to express their ideas and imagination on paper because the principle of drawing is simple and intuitive. The popularity of hand-held devices, such as PDA and Tablet PC, also contributes to the shift in the user interface preference. So in recent years, there have been many efforts in developing more intuitive interfaces for interactive design, facilitating conceptual stage design[11, 12, 15, 16, 19, 20, 22, 27, 33].

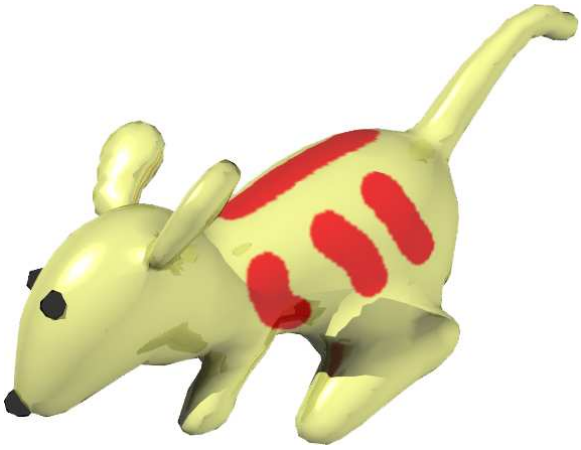


Figure 1: *A mouse shape designed using our method and rendered by POVRAY.*

A popular strategy for providing intuitive user interface is to accept input in the form of freeform sketches, drawn by the user using a mouse or a pen-like device on a graphic

† This author is partially supported by the National Natural Science Foundations (No. 60021201, 60203014) and the National Key Basic Research and Development Program (No. 2002CB312102)

tablet. For instance, Teddy is one such system developed for designing freeform polygonal shapes[16].

Inspired by these previous systems, we developed a system, named *ConvMo*, for designing prototype by generating convolution surfaces from silhouette curves. Our goal is to perform free-form implicit surface design by sketching curves; but, unlike Teddy system, we use menus and sliders as well. To enable more versatile design, we propose a new enhanced convolution model that allows users to design the cross-section shape by either sketching a closed profile curve or specifying shape parameter values. By incorporating the parametric curve formulation into the implicit model, our hybrid approach can easily produce interesting variants of shapes. Compared with the Teddy system, our method allows the user to more freely design objects with high topological complexity due to the nature of implicit surfaces. And our system supports the following additional features: shapes with holes, semi-sharp features, and free-form cross-section shapes. The proposed method is very useful for entertainment graphics applications, such as fast modeling and animation in cartoon styles.

Our main contributions are

- a method for creating convolution surface models from silhouette curves—it extracts a skeleton comprising of line-segments and convolves the skeleton with polynomial weight distribution;
- an enhanced convolution surface model that supports cross-section design.

2. Related Work

The term sketch methods usually refer to methods that attempt to infer or retrieve 3D information from strokes drawn on 2D interfaces. The most direct and popular goal is to reconstruct 3D shapes, however sketch methods have also been adopted to solve other problems. For example, a system by Tolba *et al.*[30] is a tool for perspective drawing. It projects 2D strokes on the unit sphere centered at the eye point and then views them in perspective. Bourguignon *et al.*[5] present an approach that directly uses the user-drawn strokes to infer the sketches representing the same scene from different viewpoints. These methods focus on illustrating 3D effects rather than actual 3D shapes, which is our area of interest.

Earlier sketch systems aimed at fast design of mechanical parts. They usually interpret the user's 2D sketches into exact geometric elements (e.g., lines, circular arcs, or B-spline curves), generate geometric constraints (such as right angles, tangencies, symmetry, and parallelism) possibly aided by the user's explicit selection[11, 26], and attempt to recover missing depth information[14].

Most of the recent freeform sketch methods can be roughly divided into two categories according to the object representation: *boundary-based* and *volume-based* approaches.

The Teddy[16] system is a typical boundary-based method. From a sketched silhouette, a polygonal mesh is generated, which can then be rotated using a virtual track-ball interface, and interactively edited using extrusion, bending, cutting operations, all controlled by strokes. Its innovative interface inspired many later sketch systems. However, typical of polygonal representation, it has limitations on surface smoothness and fairness, as well as design of structures with complex topology.

Michalik *et al.*[22] proposed a sketch- and constraint-based approach by using standard parametric curves and surfaces. They introduced auxiliary surfaces that allow for a reliable interpretation of the user's pen-strokes and achieved the goal of B-spline surface sculpting. Parametric representations are more compact and have higher degree of smoothness than polygonal meshes. However, it is well-known that parametric surfaces are cumbersome for representing objects with arbitrarily topology, and surface trimming has high computation price. These drawbacks limit the modeling ability of parametric surface approaches.

For the purpose of sculpture modeling, volume-based methods are more natural. The approach of Owada *et al.*[24] employed a similar interface as the Teddy system but used volumetric representation. Volumetric representations relax the topological limitation of boundary representations, but they are not as compact; moreover, achieving smooth result requires both higher storage and computation price.

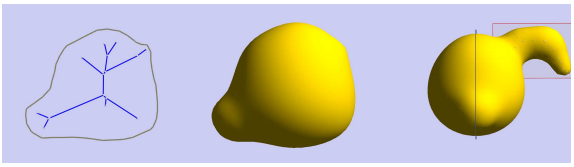
Our implicit surface-based approach can be viewed as a mixture of surface and volumetric approaches. Implicit surface representations are robust and compact for designing topologically complex objects. An implicit surface S can be defined in the following general form:

$$S = \{\mathbf{x} | \sum_{i=1}^{n} F_i(\mathbf{x}) - T = 0\}$$

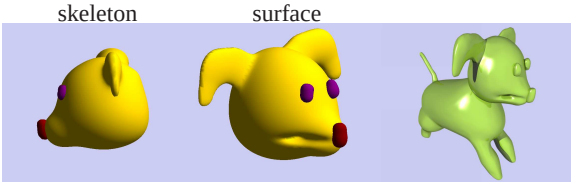
where F_i are the primitive field functions, and T is the threshold value of the iso-surface.

Convolution surface is a type of implicit surface obtained by convolving a kernel along a skeleton, composed of points, lines and polygons. Generally, convolution surface methods produce rotund shapes due to the nature of the kernel functions. Grimm proposed an approach that can sweep out implicit surfaces of different cross sectional shapes[13], but the resulting surface is a variation of distance function surfaces. With the same objective of providing more design freedom in mind, we propose an extended convolution surface model by incorporating a parametric component. Our approach implicitly controls the cross-section via space transform, supported by an intuitive user interface. The output is compatible to the original convolution surface model.

More recently, implicit surfaces based on radial basis function (RBF) are widely used in point cloud data interpolation and approximation problems[6, 31]. These methods are



(a) silhouette & skeleton (b) convolution surface (c) add component



(d) modify cross section of ear (e) soft carving (f) dog shape

Figure 2: *Method overview*

more efficient and produce better quality shapes. However, due to their isotropic property, they are inflexible for creating special feature shapes, such as darts and sharp edges. Dinh *et al.*[10] proposed a surface reconstruction method by using anisotropic basis functions, but their approach does not solve the feature representing problem completely.

Free-form design with sketching interface based on implicit surfaces has been investigated by other researchers as well. Karpenko *et al.*[19] proposed a system based on variational implicit surfaces method. They presented an elegant way to obtain 3D constraint points required by the RBF interpolating method. The RBF method, however, has high computation cost, both in solving the linear system of constraints and in the evaluation step. The authors of Teddy proposed an enhancement to their well-received system based on quadratic surface fitting[15]. This two-step approach can obtain polygonal meshes of higher fairness. But such a hybrid representation requires saving extra intermediate meshes. In contrast to these approaches, our convolution surface method has a compact representation, avoids solving a large-scale global constrained problem and computes the surface parameters in a more natural way.

3. Method Overview

The main steps of our convolution surface-based prototype design system are as follows (Figure 2):

1. The user draws a silhouette curve on a projection plane and the system automatically extracts the skeleton structure consisting of line segments (Section 5.1).
2. The system determines the parameters for each skeletal line segment, and generates a rotund generic convolution surface model that fits the silhouette curve (Section 5.2). The system outputs a polygonal mesh using the marching tetrahedra polygonization algorithm[2].
3. The user modifies the circular cross-section of the generic shape by either sketching a cross-section profile or by specifying shape parameter values provided by our enhanced convolution surface model (Section 4.2).
4. The user may perform further component-based editing operations, such as soft carving and hole making (Section 6).
5. The user rotates the partially designed shape to obtain a new projection plane, and repeats steps 1-4 to design another surface component, which can be merged smoothly with, or simply attached to, an existing component.

Furthermore, steps 3 and 4 can be repeated on any component at any time.

4. Convolution Surface Model

Convolution surfaces were introduced by Bloomenthal and Shoemake[4] as a natural extension to point-based field surfaces to include higher-dimensional geometric elements.

A modeling skeleton set $\mathbf{S} \subset \mathbb{R}^3$ can be represented as a field function $g : \mathbb{R}^3 \mapsto \mathbb{R}$, defined as follows:

$$\begin{cases} g(\mathbf{p}) > 0, & \mathbf{p} \in \text{skeleton } \mathbf{S}; \\ g(\mathbf{p}) = 0, & \text{otherwise}; \end{cases} \tag{1}$$

where $\mathbf{p}$ is a point in $\mathbb{R}^3$ Euclidean space. Let $f : \mathbb{R}^3 \mapsto \mathbb{R}$ be a *kernel function* representing the field generated by a single point in the skeleton, and $\mathbf{x}$ be a point in the skeleton $\mathbf{S}$, then the total field contributed by the skeleton at a point $\mathbf{p}$ is the convolution of the functions f and g as follows:

$$F(\mathbf{p}) = (f \otimes g)(\mathbf{p}) := \int_{\mathbf{S}} g(\mathbf{x}) f(\mathbf{p} - \mathbf{x}) d\mathbf{x} \tag{2}$$

A convolution surface with threshold T is then defined by

$$S = \{\mathbf{p} | F(\mathbf{p}) - T = 0, \mathbf{p} \in \mathbb{R}^3\} \tag{3}$$

Although Bloomenthal and Shoemake demonstrated their results using a cubic approximation to a Gaussian, we utilize the Cauchy kernel and its closed-form solution as presented by McCormack and Sherstyuk[21, 28]

$$f(\mathbf{p}) = \frac{1}{(1 + s^2 r^2)^2} \tag{4}$$

where $r = \|\mathbf{p}\|$ and s is a parameter for controlling the kernel width. This allows us to develop an analytical polynomial-weighted model in the next section; the model can, however, be used with any kernel.

The model proposed by McCormack and Sherstyuk is of uniform distribution, i.e., the resulting surface has constant radius of influence along the skeletal element. To facilitate the design of generalized cylindrical shapes with fewer skeletal elements, Jin *et al.*[17, 18] proposed a polynomial weighted model and derived analytical solutions for line segments, arcs, and quadratic curves.

For efficiency and robustness, we choose to use line segment as the skeleton element in our system. To reduce the

number of segments required, which is critical in interactive design, we adopt the polynomial-weighted convolution model. This model is briefly reviewed in the next subsection, and an enhanced model is presented in Section 4.2 for more versatile cross-sectional design.

4.1. Polynomial-weighted convolution model

Let the skeleton $\mathbf{S}$ be an arc-length parameterized curved segment $\mathbf{c}(t)$ $(0 \le t \le l)$. A convolution surface with a polynomial weight $w(t)$ distribution can be defined as

$$F(\mathbf{p};\mathbf{c}(\cdot),w(\cdot)) = \int_0^l w(t) f(\mathbf{p}-\mathbf{c}(t))\mathrm{d}t. \tag{5}$$

For a line segment $\mathbf{L}(t) = \mathbf{b} + t\vec{\mathbf{a}}$, $0 \le t \le l$, the squared distance from a point $\mathbf{p}$ to the line $\mathbf{L}(t)$ is given by $r^2(t) = \|\mathbf{b}+t\vec{\mathbf{a}}-\mathbf{p}\|^2 = d^2+t^2-2ht$, where $\vec{\mathbf{d}} = \mathbf{p}-\mathbf{b}$, $d = \|\vec{\mathbf{d}}\|$, and $h = \vec{\mathbf{d}}\cdot\vec{\mathbf{a}}$. Thus, for Cauchy kernel, equation (5) becomes

$$F(\mathbf{p};\mathbf{L},w) = \int_0^l \frac{w(t)}{(1+(d^2+t^2-2ht)s^2)^2}\mathrm{d}t. \tag{6}$$

In particular, we use linear weight distribution: varying from w_0 to w_1 along the line segment, the field function is

$$F(\mathbf{p}) = w_0 F_1(\mathbf{p}) + \frac{w_1-w_0}{l} F_t(\mathbf{p}), \tag{7}$$

where

$$\begin{aligned} F_1(\mathbf{p}) := F(\mathbf{p};\mathbf{L},1) = \tfrac{1}{2p^2}\left[\tfrac{h}{s^2h^2+p^2} + \tfrac{l-h}{s^2(l-h)^2+p^2}\right] \\ + \tfrac{1}{2sp^3}\left[\tan^{-1}\left(\tfrac{sh}{p}\right) + \tan^{-1}\left(\tfrac{s(l-h)}{p}\right)\right], \end{aligned} \tag{8}$$

and

$$\begin{aligned} F_t(\mathbf{p}) := F(\mathbf{p};\mathbf{L},t) = \\ \tfrac{1}{2s^2}\left[\tfrac{1}{s^2h^2+p^2} - \tfrac{1}{s^2(l-h)^2+p^2}\right] + hF_1(\mathbf{p}) \end{aligned} \tag{9}$$

with p being a distance term $p^2 = 1+s^2(d^2-h^2)$.

4.2. Enhanced convolution model

The polynomial-weighted model still produces only circular cross-section shapes. Since many objects in the real world have anisotropic shapes, it motivates us to enhance the original convolution model to achieve more versatile design.

We refer to the initial polynomial-weighted convolution surface with circular cross-section as the *generic surface*. The user designs a *target surface* by modifying the generic surface. Thus the main issue is how to define a mapping from the field function of the generic surface to the field function of the target surface $\mathcal{M} : F_{generic} \mapsto F_{target}$. For the purpose of interactive design, the mapping procedure should be intuitive and have low response time. Based on the idea of spatial deformation, we assume that the mapping between the two field functions is effected by a transform $\mathcal{T} : \Omega_{generic} \mapsto \Omega_{target}$, where Ω_* is the domain of function F_*. That is, if $F_{generic}$ and the corresponding $\mathcal{T}$ transform function of a line segment are given, then

$$F_{target}(\cdot) = \mathcal{M}(F_{generic}) = F_{generic}(\mathcal{T}^{-1}(\cdot)). \tag{10}$$

To evaluate a target surface, the implicit function evaluator first transforms a point back to the generic surface space and then evaluates the original convolution surface model to obtain a field value for the target surface.

To find a suitable representation for the transform $\mathcal{T}$, we define a local coordinate system for each line segment $\mathbf{l}$ as shown in Figure 3. Let the y-axis be normal to the reference plane supporting the silhouette curve, the z-axis be along the line segment direction, and the x-axis be in the direction $\mathbf{l}\times\mathbf{n}$. Without losing generality, let the origin be the start point of the line segment. We can then decompose a spatial transformation into three scalar mappings along these axis directions

$$\mathcal{T} = \mathcal{T}_x \cdot \mathcal{T}_y \cdot \mathcal{T}_z, \tag{11}$$

where $\mathcal{T}_x$, $\mathcal{T}_y$ and $\mathcal{T}_z$ are transform functions defined on the *y*-*z* plane, *z*-*x* plane, and *x*-*y* plane, respectively. In general, they can be curves defined in the B-spline form for achieving complex design; but for simplicity of interface, we adopt the following decomposition:

$$\mathcal{T} = \mathcal{S}_x \cdot \mathcal{S}_y \cdot \mathcal{T}_z \tag{12}$$

where the mapping $\mathcal{T}_z$ defines the cross-section shape perpendicular to the silhouette supporting plane, and the mappings $\mathcal{S}_x$ and $\mathcal{S}_y$ expand or contract the cross-section in the *x* and *y* directions, respectively. We define $\mathcal{T}_z$ in terms of polar coordinate as follows:

$$\mathcal{T}_z : (r,\theta) \mapsto (r',\theta')$$

and define the scaling mappings as

$$\mathcal{S}_x : x \mapsto \beta_x x \qquad\qquad \mathcal{S}_y : y \mapsto \beta_y y.$$

Our system provides two ways for the user to specify $\mathcal{T}_z$: sketch a closed profile curve or specify a parameter defining a super-quadric. The scaling factors β_x and β_y are usually equal to 1 in the first case, but are useful in the second case.

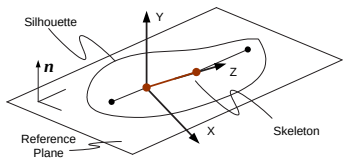


Figure 3: *Local coordinate system of a line segment*

Class 1. Bezier curve model in polar coordinates. We consider the mapping image of the unit circle, i.e., $R(\theta) = \mathcal{T}_z(1,\theta)$. Then a class of transform functions $\mathcal{T}_z$ is given by

$$\mathcal{T}_z : (r,\theta) \mapsto (r\cdot R(\theta),\theta), \tag{13}$$

with the inverse of $\mathcal{T}_z$ easily found as follows:

$$\mathcal{T}_z^{-1} : (r', \theta') \mapsto (r'/R(\theta'), \theta'). \quad (14)$$

Our system allows the user to sketch $R(\theta)$ as the cross-section shape and automatically converts it to a Bezier curve in polar coordinates. The closed curve is constrained to be of a star shape. For completeness, we include here some derivations on polar-form Bezier curves. An n-degree polar-form Bezier curve segment $b(\theta)$ with $\theta_s \le \theta \le \theta_e$ is defined as

$$b(\theta) = \sum_{i=0}^{n} r_i B_{n,i}(\frac{\theta - \theta_s}{\theta_e - \theta_s}). \quad (15)$$

where

$$B_{n,i}(t) = \binom{n}{i} (1-t)^{n-i} t^i, \; 0 \le t \le 1 \quad (16)$$

are the Bernstein polynomials, r_i are the control radii. Clearly, when $r_0 = \cdots = r_n = R_0$, the curve represents an arc of radius R_0. A closed curve $R(\theta)$ $(0 \le \theta \le 2\pi)$ can be constructed by stitching together several curve segments, with each segment $b_k(\theta)$, $\theta_k \le \theta \le \theta_{k+1}$ $(0 \le k < m)$ defined by the control radii $r_{k,0}, r_{k,1}, \ldots, r_{k,n}$. In our implementation, we let $n = 3$, $m = 12$, and $\theta_k = 2k\pi/m$. The system samples points on the sketched curve and fits the points with a closed polar-form Bezier curve. By the endpoint interpolation property of Bezier curves, $b_k(\theta_k) = r_{k,0}$ and $b_k(\theta_{k+1}) = r_{k,n}$. To ensure that the segments connect up at their endpoints, $r_{k,n} = r_{k+1,0}$ $(0 \le k < m)$ must be satisfied. The radii $r_{k+1,0}$ $(0 \le k < m)$ are simply computed by averaging the nearest two sampling points at the angle θ_k, and the remaining $r_{k,j}$ are obtained by solving a least-squares problem (see Appendix A). To achieve tangent continuity, the control radii can be modified according to the following sufficient condition (see Appendix B for detail derivation)

$$r_{k,n} = r_{k+1,0} = \frac{1}{2}[r_{k,n-1} + r_{k+1,1}]. \quad (17)$$

Class 2. Super-quadrics model. For simplicity of designing the commonly used elliptic and squarish cross-section shapes, our system provides an alternative way of specifying $\mathcal{T}_z$ in terms of super-quadrics. In parametric form, super-quadrics can be represented as

$$\begin{cases} x &= r\ \mathrm{sign}(\cos\theta)\ |\cos\theta|^{1/\alpha} \\ y &= r\ \mathrm{sign}(\sin\theta)\ |\sin\theta|^{1/\alpha} \end{cases} \quad (0 < \theta < 2\pi) \quad (18)$$

Here $\mathrm{sign}(a)$ is the sign function, i.e., $\mathrm{sign}(a) = 1$ when $a > 0$,otherwise $\mathrm{sign}(a) = -1$. And $\alpha > 0$ acts as a shape control parameter. When $\alpha = 1$, we get a circle; the shape becomes star shape with sharper corners when $\alpha < 1$; it converges to a square when $\alpha \to \infty$. Hence, the advantage of the super-quadrics model is that only one parameter value has to be specified. The mapping $\mathcal{T}_z$ is then given by:

$$\mathcal{T}_z : (r, \theta) \mapsto (r[|\cos\theta|^\alpha + |\sin\theta|^\alpha]^{-1/\alpha}, \theta). \quad (19)$$

Many objects have components that are of 2.5-dimension, i.e., which can be defined by extruding a curve to some uniform thickness (Figure 4(b)). However, such components are not within the modeling domain of Equation 19 because the field function is not uniform along the line segment. To achieve the extrusion effect along the y-axis direction, we introduce the following extrusion mapping by modifying the previous equation:

$$\mathcal{T}_z : (\sqrt{(r'\cos\theta')^2 + |r'\sin\theta'|^\alpha}, \theta') \mapsto (r', \theta'). \quad (20)$$

Here $\alpha \ge 2$ controls the stiffness of the shape just as in the case of generic super-quadric. Since the term $(r'\sin\theta')^\alpha$ acts as a soft threshold function at 1 along the y-axis, the resulting shape has nearly uniform thickness of 2 for large values of α, say $\alpha = 50$. Other extrusion thickness can be obtained by combining the effect with the scaling mapping $\mathcal{S}_y$.

Figure 4 shows an example of designing the tail of a cartoon style dinosaur. Sub-figures (b) and (c) use the super-quadrics model, and (d) illustrates the freeform cross-section design. In our experimental system, the shape parameters α, β_x and β_y are directly controlled through sliders.

5. Convolution Surface From Silhouette Curve

With our system ConvMo, the user designs a prototype shape by incrementally adding new components by sketching silhouette curves of different viewing directions. The system generates a convolution surface from each silhouette, and then allows the user to sculpt the surface into a more interesting shape by modifying its cross-section and performing other component-based editing.

To design an object, the user sketches the silhouette of a component as a simple curve with no self intersections in the input canvas. The system then converts the curve into a simple polygon by sampling the input device movement. To obtain stable input, filtering is required. Let W be the canvas width. We sample such that the length of each polygon edge is at most $\mu_0 W$, and at least $\mu_1 W$ if the edge forms an angle of more than 20 degree with the previous edge (we let $\mu_0 = 0.05$ and $\mu_1 = 0.01$). If the polygon is self-intersecting, the user is requested to sketch another curve.

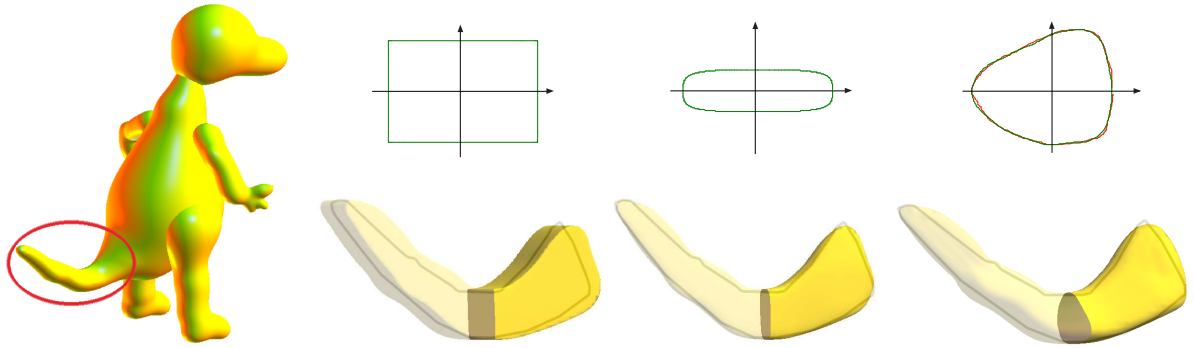
To generate a convolution surface from this simple polygon, two major issues have to be addressed:

- find a suitable skeleton with a small number of primitives;
- determine the parameters of the field contribution of skeletal primitives.

Briefly, our approach finds an approximate medial axis comprising of line segments, and determines the weight at each end of a line segment as a function of the segment length and the distance to the silhouette polygon.

5.1. Skeleton extraction

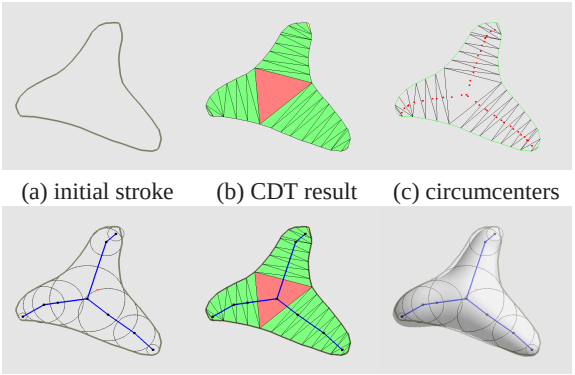
There are many algorithms reported in the literature for finding skeletons[7, 23]. A well known class of algorithms



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

Figure 4: *Designing the tail of dinosaur. (a) is the generic dinosaur shape. (b) uses Equation 20 for T_z with $\alpha = 100$ and $\beta_y = 2$, and (c) uses the standard super-quadric model for T_z with $\alpha = 2$ and $\beta_y = 3$. (d) uses the sketching method to obtain a Bezier curve for T_z.*

finds the medial axis using Voronoi diagram or Delaunay triangulation[8]. A simple method is to find a rough skeleton by applying the constrained Delaunay triangulation (CDT)[29] and connecting the midpoints of the internal edges[16, 25]. Since this skeleton is not a medial axis (i.e., the locus of the centers of maximally inscribed circles inside the 2D shape), additional computation is needed to find the proper radii of influence to produce a good-fit convolution surface. In contrast, we determine an approximate medial axis, comprising of line segments, so that the parameters of the convolution model can be more directly determined. Our algorithm involves two steps: (1) apply the CDT to the simple polygon, with the polygon edges constrained to be in the triangulation, and link up the circumcenters of all the triangles according to their connectivity (Figure 5(a)-(c)); (2) prune redundant circumcenters to obtain a simplified skeleton with fewer line-segment primitives (Figure 5(d)-(e)).



(a) initial stroke (b) CDT result (c) circumcenters

(d)influence circles (e) filtered skeleton (f) result surface

Figure 5: *Generating convolution surface from silhouette*

We refer to the initial polygon edges as the *external edges*, and refer to the edges added by the Delaunay triangulation as the *internal edges*. A triangle in the CDT is then called a *junction triangle*, *sleeve triangle*, or *terminal triangle* if it has three, two or one internal edges, respectively[25].

To prune the redundant circumcenters, we process the triangles in order, starting from a junction triangle until a terminal or another junction triangle is reached. Let $\mathbf{p}$, $\mathbf{q}$, $\mathbf{r}$ be three most-recently-found circumcenters obtained in that order. The point $\mathbf{r}$ is pruned if its associated triangle is a sleeve triangle and it satisfies one of the following two criteria (Figure 6).

Overlapping criterion The circumcircle of $\mathbf{r}$ lies mostly inside the circumcircle of $\mathbf{q}$. We test this criterion using

$$\|\mathbf{q}-\mathbf{r}\| + \tau R_{\mathbf{r}} \leq R_{\mathbf{q}},$$

where $R_{\mathbf{x}}$ denotes the radius at circumcenter $\mathbf{x}$; $\tau = 0.6$ is chosen so that a small but noticeable feature contributed by the point $\mathbf{r}$ would still be retained.

Ordering criterion The circumcircle of $\mathbf{r}$ lies on the half-plane defined by $(\mathbf{q}, \mathbf{n})$, where $\mathbf{n}$ is the normal to the common edge of the triangles associated to $\mathbf{q}$ and $\mathbf{r}$, directed towards the triangle of $\mathbf{r}$. We test it with

$$(\mathbf{r}-\mathbf{q})\cdot\mathbf{n} \leq 0.$$

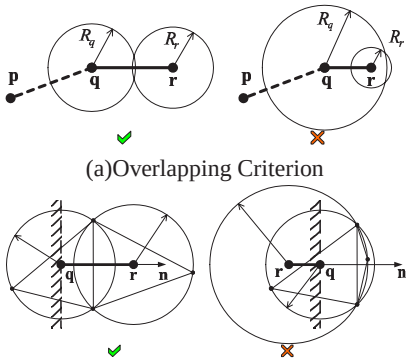
We remove the point $\mathbf{q}$ if the following criterion is satisfied.

Proportionality criterion The points $\mathbf{p}$, $\mathbf{q}$, and $\mathbf{r}$ are nearly collinear, measured by

$$\frac{(\mathbf{q}-\mathbf{p})\cdot(\mathbf{r}-\mathbf{q})}{\|\mathbf{q}-\mathbf{p}\|\|\mathbf{r}-\mathbf{q}\|} \geq \cos(\pi/18),$$

and their circumradii vary proportionally to their distances apart:

$$-0.1 < \frac{R_{\mathbf{q}}-R_{\mathbf{p}}}{R_{\mathbf{r}}-R_{\mathbf{p}}} - \frac{\|\mathbf{q}-\mathbf{p}\|}{\|\mathbf{r}-\mathbf{p}\|} < 0.1$$



(a)Overlapping Criterion

(b)Ordering Criterion

Figure 6: *Skeleton filtering criteria.*

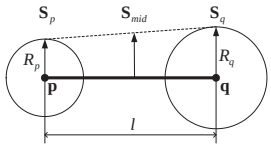


Figure 7: *Convolution surface from a line segment.*

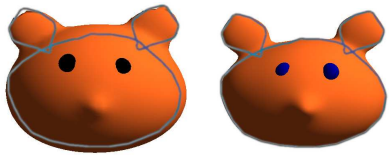


Figure 8: *The resulting implicit surfaces without (left) and with (right) the global fitting step.*

5.2. Constructing generic convolution surfaces

We determine the parameters that control the field contribution of each skeletal line segment in two fitting steps: a local inferring step and a global fitting step.

In the local inferring step, for each line segment of length l with endpoints $\mathbf{p}$ and $\mathbf{q}$, we infer the weights w_0 and w_1 such that the field contribution produces a convolution surface that locally fits the silhouette curve well. Rewriting Equation 7 in symmetric form, we have

$$F(\mathbf{p}) = w_0 F_{(l-t)}(\mathbf{p}) + w_1 F_t(\mathbf{p}) \tag{21}$$

where $F_{(l-t)}(\mathbf{p}) = lF_1(\mathbf{p}) - F_t(\mathbf{p})$. We then set the field contribution at two selected points—the point $\mathbf{s}_p$ at distance R_p above $\mathbf{p}$ and the point $\mathbf{s}_q$ at distance R_q above $\mathbf{q}$ (Figure 7)—to be equal to the desired implicit surface threshold T:

$$\begin{cases} T = F(\mathbf{s}_p) = w_0 F_{(l-t)}(\mathbf{s}_p) + w_1 F_t(\mathbf{s}_p), \\ T = F(\mathbf{s}_q) = w_0 F_{(l-t)}(\mathbf{s}_q) + w_1 F_t(\mathbf{s}_q). \end{cases} \tag{22}$$

Since $F_t(\mathbf{s}_p)$ and $F_{(l-t)}(\mathbf{s}_q)$ are relatively small, we can assume that they are negligible, and approximately compute the two weights as follows:

$$\begin{cases} w_0 &= T/F_{(l-t)}(\mathbf{s}_p), \\ w_1 &= T/F_t(\mathbf{s}_q). \end{cases} \tag{23}$$

This resulting isosurface, however, would bulge in the middle beyond the silhouette boundary. Hence, to ensure that the surface passes through $\mathbf{s}_{mid}$, which is the point at a distance $(R_p + R_q)/2$ above $(\mathbf{p}+\mathbf{q})/2$, we scale the weights w_0 and w_1 by a factor of κ:

$$\kappa = \frac{T}{w_0 F_{(l-t)}(\mathbf{s}_{mid}) + w_1 F_t(\mathbf{s}_{mid})}. \tag{24}$$

For skeletal endpoints that have multiple connecting line segments (i.e., points attributed to a sleeve or a junction triangle), we divide its weight by that number of line segments so as to cancel out the multiple field contributions.

Finally, we multiply each field contribution $F_i(\mathbf{p})$ by a control scale λ_i and sum all the contributions from the skeletal segments $L_i (i = 1, 2, \ldots, n)$:

$$F(\mathbf{p}) = \sum_{i=1}^{n} \lambda_i F(\mathbf{p}; L_i) \tag{25}$$

Usually, λ_i is equal to 1. In the next section, we employ negative λ_i for soft carving operation.

This local inferring step produces a convolution surface that, in general, does not interpolate the given silhouette curve. Nevertheless in our experiments we have found the fitting to be adequate for the purpose of prototype design. Note that a convolution surface is the result of summing the fields of all skeletal primitives, thus it is in fact a global shape. We propose a global interpolation step for obtaining a tighter fitting (Figure 8).

Let the input stroke be consisting of points $\{\mathbf{p}_i, \ldots, \mathbf{p}_m\}$. We solve the following constrained least-squares problem for $\Lambda = [\lambda_1, \ldots, \lambda_n]^\top$:

$$\min_{\Lambda \geq \mathbf{0}} (\mathbf{F}\Lambda - \mathbf{T})^\top (\mathbf{F}\Lambda - \mathbf{T}) \tag{26}$$

where $\mathbf{F}_i = [F(\mathbf{p}_1; L_i), \ldots, F(\mathbf{p}_m; L_i)]^\top$, $\mathbf{F} = [\mathbf{F}_1, \ldots, \mathbf{F}_n]$, and $\mathbf{T} = [T, \ldots, T]^\top$. The condition $\Lambda \geq \mathbf{0}$ here guarantees that the resulting surface has the correct topology. Since $\mathbf{F}^\top \mathbf{F}$ is semi-positive, this constrained quadratic convex programming problem can be solved by a simplified version of Hildreth-d'Esopo method[9]. Nested within this algorithm is a Gauss-Seidel iteration, and the problem size n is usually between 10 to 100, thus the computation cost is small for this fitting step.

Unfortunately, reflecting the general 'over-fitting' problem of interpolation methods, the fairness of the global fitting surface could be worse than the non-interpolating one when the input silhouette has a zigzag shape. Therefore, this global fitting step is kept as optional in our system.

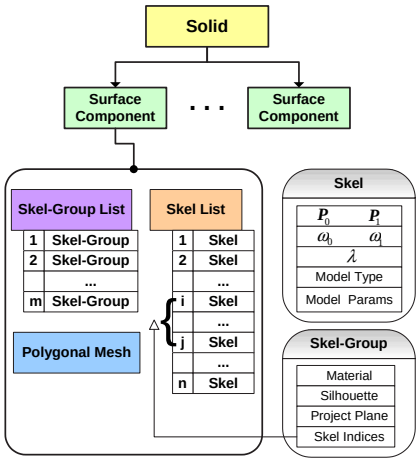


Figure 9: *Data structure of our prototype design system.*

6. Implementation Details

The design primitives are organized into hierarchical levels in our prototype system (Figure 9). In the top level, the solid, or the design object, consists of several *surface components*. Each surface component is represented by a convolution surface defined by Equation 25. For example, the dinosaur shape in Figure 4 has six surface components. Every weighted component $\lambda_i F_i$ in Equation 25 corresponds to a skeleton element or *skel*. Each skel has associated shape parameters of the enhanced convolution surface model. These skels are naturally grouped into sets, named *skel-group*s, according to whether they belong to the same silhouette curve (or same set of silhouette curves consisting of a boundary curve and one or more loop, see Section 6.3). Input information, such as projection plane, silhouette curve(s) and rendering material, is stored in the skel-group structure. Currently, as a tradeoff between system flexibility and complexity, all the operations are performed at the skel-group level. Based on this structure, the system can provide a set of powerful and flexible design operations. Since all the design operations have compact representation, it is very easy to support copy, paste, 'un-do' and 're-do' edit commands, which are very common in practical user interfaces. All the representation information can be saved to a simple script file. Thus the whole design procedure can be recorded, and the user can further modify the surface at the skel level if desired.

6.1. Merging new components

The user progressively builds a more complex object by adding new components in two ways. First, the user may design a separate surface component and simply attach it to an existing surface component. Second, the user can sketch a new silhouette curve and designate the resulting component (the associated skel-group) to be merged smoothly with an existing surface component. To sketch a new silhouette, the user first identifies the depth of the projection plane by drawing a small loop on an existing component, then rotates the object to obtain a projection plane. This interface is similar to the extrusion operation of Teddy. The difference is that in our case the drawn loop serves only to indicate the depth of the projection plane, and not as the base loop for extrusion.

Similar to conventional modeling systems, we also provide skel-group translate, rotate, clone and mirror operations. The cross-section user interface is defined at this level. With these operations, the user can control the procedure of designing new components more precisely.

6.2. Soft carving

We introduce a carving operation by defining a group of carving skels lying outside a surface boundary. Carving skels have negative λ_i in Equation 25. The user first creates a disconnected component as the carving tool by drawing a silhouette, and then designates the resulting skels to be carving skels by negating the λ_i values. By moving the carving tool to the desired position and orientation overlapping with the object to be carved, the desired shape is carved out by merging the two components (Figure 10). This operation resembles the carving process familiar to artists: rough shape is first created and details are created by carving out some volume of mass.

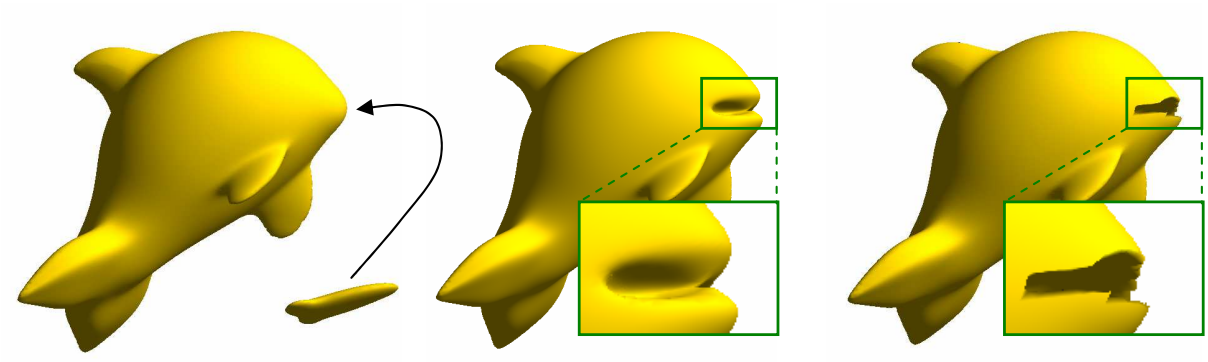
Similar to the early work of Wyvill *et al.*[32], volume boolean operators can also be applied to the components. For instance, we can define the subtract operator as follows:

$$S_{A-B} = \{\mathbf{x} | \min(F_A(\mathbf{x}) - T, T - F_B(\mathbf{x})) = 0\}$$

to sharply cut a part from a shape. The output is similar to carving. However, since the resulting field function is not continuous, the subtract operator sometimes causes sampling problem in the polygonalization algorithm and results in jaggy effect (see Figure 10(c)). Comparing with the Boolean operations, carving can produce smoother and softer results.

6.3. Designing surfaces with holes

Surfaces with handles or holes can be easily created using our system. One way to create a hole, such as the one in a torus, is by smoothly merging two components to meet at two locations. Another more direct way provided by our system is to let the user draw an inner silhouette loop within an outer silhouette curve. Our skeleton extraction algorithm in Section 5.1 can handle such cases. For convenience, instead of drawing two nested curves, we also allow the user to directly specify a closed curve, approximated by line segments, as the skeleton, and assign a radius to each skeletal point. The cylindrical part of the cup in Figure 12 is created in this manner.



(a) (b) (c)

Figure 10: *Carving vs. Boolean operation. (a) shows a whale shape and a carving tool. (b) and (c) are the results of soft carving and hard subtraction operation, respectively. All the whales are polygonalized at the same resolution.*

7. Design Examples

We demonstrate the capabilities of our system with examples in Figure 11 to 13. In these figures, the grey curves are the input silhouettes, the skels are drawn in green or blue lines, and the red lines indicate the β_y scaling direction of the enhanced convolution method. Table 1 lists some computation details for these objects. The chair shape contains some 2.5-dimension components, i.e., the rectangular seat and the four legs are designed using Equation 20. A variety of styles of chairs can be easily obtained by changing the parameters of the transform mapping $\mathcal{T}$. The cup shape is an example of objects with holes and handles. The cup body is designed by sketching the top circular curve as the skeleton and setting β_y to extrude the shape vertically. The rabbit and mouse in Figure 13 and 1 are examples of attractive prototype objects, which typically take a novice user about 10 minutes to design using our system.

Model	# Surface components	# Skel-groups	# Skels	MT time(sec) (0.5/0.25)
Chair	1	6	32	1.7 / 6.1
Cup	1	3	74	6.7 / 27.4
Dinosaur	6	16	126	1.1 / 4.1
Mouse	4	12	95	3.6 / 14.7
Rabbit	8	13	178	7.9 / 31.6

Table 1: *Model data. MT time refers to Marching Tetrahedra polygonization time. The two resolutions used are 0.5 and 0.25 cube-size. The timing results are obtained on a standard 2.4GHz Pentium IV PC with 512MB memory.*

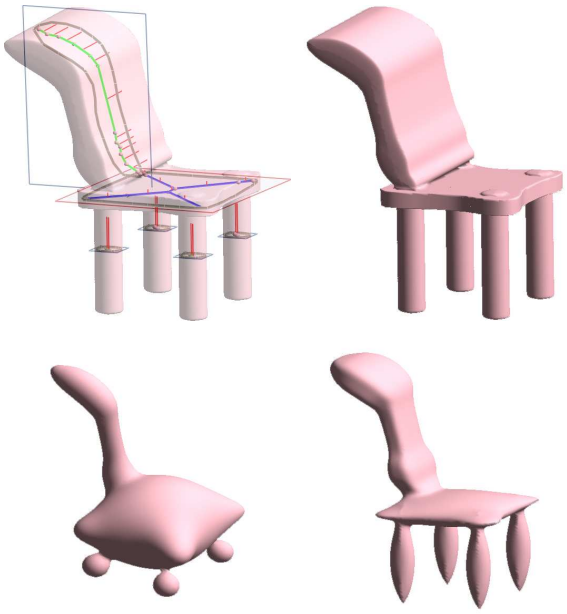


Figure 11: *Chair shape. Skeleton structure (top left), generic surface (bottom left), rigid-style produced by our enhanced convolution model (top right), another style obtained by tuning the parameters of $\mathcal{T}$ (bottom right).*

8. Conclusions and Future Work

We have presented a system based on convolution surface for designing meaningful freeform shapes from silhouette curves. The convolution surface model lends itself to intuitive, robust, and versatile design operations.

Comparing with single-resolution polygonal meshes, the implicit surface representation can be viewed as a continuous but compact representation. Meshes of arbitrary resolution can be extracted by the marching tetrahedra algorithm. As the implicit function evaluation is still the main bottleneck of our method, to achieve interactive response, we use

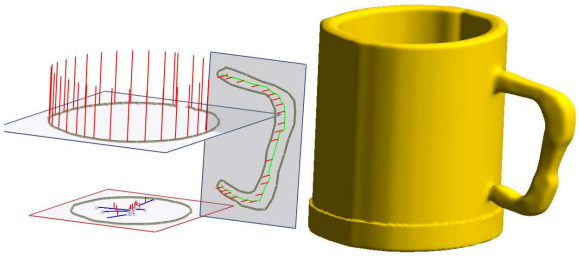


Figure 12: *Cup shape. Skeleton structure and resulting surface.*

a low sampling rate in the early stages of design, and only increase the sampling rate for the final output. Most of the skels in an object can be adequately represented by the original polynomial-weighted convolution model; only a handful of them requires the capability of the enhanced model. We add a condition switch to each skel so that the more expensive evaluation is only performed for the enhanced models. This strategy sped up the whole polygonization procedure by about 2-3 times in our experiments.

For simplicity and efficiency, we chose to use only line segment as the skeleton primitive. A weakness of convolving branching line segments is that bulges appear around junctions. We handled this bulging problem by scaling the field contributions at joints between line segments. Our simple and practical solution produces satisfactory surfaces in most of our experiments. The global fitting step also helps to solve the bulge problem. An alternative solution to the bulging problem is to include polygons in the skeleton and apply certain constraints[3], however, deriving a closed-form representation of polynomial weighted convolution surface in this case would be challenging and numerical method may be required. From the system implementation point of view, the tradeoff between additional primitive and increased capabilities also needs careful consideration. Nevertheless, a comparison study of these two branching methods is an interesting future work.

Due to the medial axis being sensitive to changes in input sample points, our skeleton extraction algorithm may produce unstable results for certain inputs. Future work may include exploring other skeletonization algorithms[1, 7] to extract alternative skeletons, possibly containing parabolic arcs[17] and polygons.

Acknowledgements

We wish to thank John Hughes for his constructive comments on an earlier version of the paper. We are also grateful to the anonymous reviewers for careful reading and comments which helped improve the exposition of the paper.

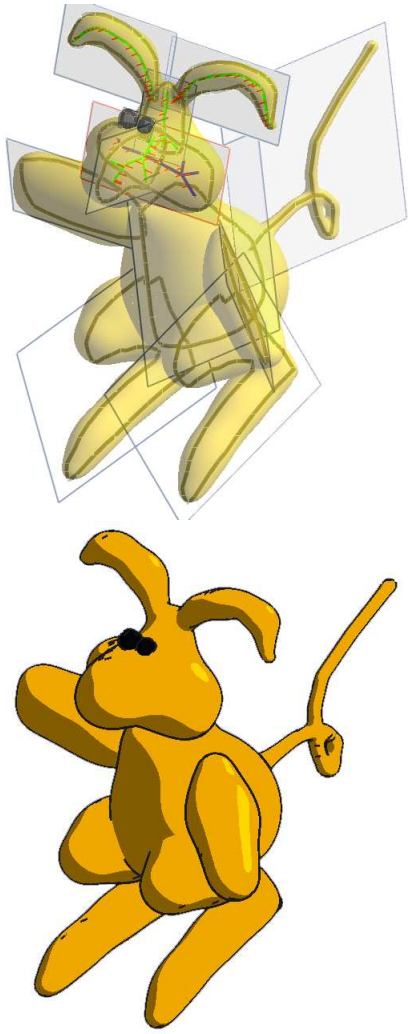


Figure 13: *Rabbit shape. Skeleton structure (upper), and the rabbit rendered with cartoon shading (lower).*

References

1. O. Aichholzer and F. Aurenhammer. Straight skeletons for general polygonal figures in the plane. In *Proc. 2nd Annu. Int'l Computing and Combinatorics Conf.*, pages 117–126, 1996. 10

2. J. Bloomenthal. An implicit surface polygonizer. In P.S. Heckbert, editor, *Graphics Gems IV*, pages 324–349. Academic Press, Cambridge, 1994. 3

3. J. Bloomenthal. Bulge elimination in convolution surfaces. *Computer Graphics Forum*, 16(1):31–41, 1997. 10

4. J. Bloomenthal and K. Shoemake. Convolution surface. *ACM SIGGRAPH 1991 Conference Proceedings*, pages 251–256, 1991. 3

5. D. Bourguignon, M.-P. Cani, and G. Drettakis. Drawing for il-

lustration and annotation in 3D. In *Computer Graphics Forum*, volume 20, pages 114–122, 2001. 2

6. J.C. Carr, R.K. Beatson, J.B. Cherrie, T.J. Mitchell, W.R. Fright, B.C. McCallurm, and T.R. Evans. Reconstruction and representation of 3D objects with radial basis function. *ACM SIGGRAPH 2001 Conference Proceedings*, pages 67–76, 2001. 2

7. F. Chin, J. Snoeyink, and C. Wang. Finding the medial axis of a simple polygon in linear time. *Discrete Computational Geometry*, 21(3):405–420, 1999. 5, 10

8. M. deBerg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational geometry–algorithms and applications*. Springer-Verlag, 2000. 6

9. D. A. D'Esopo. A convex programming procedure. *Naval Research Logistic Quarterly*, 6:33–42, 1959. 7

10. H. Q. Dinh, G. Slabaugh, and G. Turk. Reconstructing surfaces using anisotropic basis functions. In *International Conference on Computer Vision 2001*, pages 606–613, Vancouver, Canada, July 2001. 3

11. L. Eggli, C. Hsu, B. Bruderlin, and G. Elber. Inferring 3D models from freehand sketches and constraints. *Computer-Aided Design*, 29(2):101–112, 1997. 1, 2

12. T. Galyean and J. F. Hughes. Sculpting: an interactive volumetric modeling technique. *ACM SIGGRAPH 1991 Conference Proceedings*, pages 267–274, 1991. 1

13. C. M. Grimm. Implicit generalized cylinders using profile curves. In *Implicit surface 1999*, 1999. 2

14. I. J. Grimstead and R.R. Martin. Creating solid models from single 2D sketches. *Proc. of the Third Symposium on Solid Modeling and Applications*, pages 323–337, May 1995. 2

15. T. Igarashi and J. F. Hughes. Smooth meshes for sketch-based freeform modeling. In *ACM Symposim on Interactive 3D Graphics*, pages 139–142, Monterey, California, April 2003. 1, 3

16. T. Igarashi, S. Matsuoka, and H. Tanaka. Teddy: A sketching interface for 3D freeform design. *ACM SIGGRAPH 1999 Conference Proceedings*, pages 409–416, 1999. 1, 2, 6

17. X. Jin and C. L. Tai. Analytical convolution surfaces for quadratic curve skeletons with polynomial weight distributions. *Visual Computer*, 18(8):530–546, 2002. 3, 10

18. X. Jin, C. L. Tai, J. Feng, and Q. Peng. Convolution surfaces for line skeletons with polynomial weight distributions. *Journal of Graphics Tools*, 6(3):17–28, 2001. 3

19. O. Karpenko, J. F. Hughes, and R. Raskar. Free-form sketching with variational implicit surfaces. *Computer Graphics Forum*, 21(3), 2002. 1, 3

20. L. Markosian, J. M. Cohen, T. Crulli, and J. F. Hughes. Skin: A constructive approach to modeling free-form shapes. *ACM SIGGRAPH 1999 Conference Proceedings*, pages 393–400, 1999. 1

21. J. McCormack and A. Sherstyuk. Creating and rendering convolution surfaces. *Computer Graphics Forum*, 17(2):113–120, 1998. 3

22. P. Michalik, D. Kim, and B. Bruderlin. Sketch- and constraint-based design of B-spline surfaces. In *Solid Modeling 2002*, pages 297–304, 2002. 1, 2

23. R. Ogniewicz and M. Ilg. Voronoi skeletons: theory and applications. *Proc. Computer Vision and Pattern Recognition*, pages 63–69, June 1992. 5

24. S. Owada, F. Nielsen, K. Nakazawa, and T. Igarashi. *Lecture Notes in Computer Science*, volume 2733, chapter A Sketching Interface for Modeling the Internal Structures of 3D Shapes, pages 49–57. Springer-Verlag Heidelberg, 2003. 2

25. L. Prasad. Morphological analysis of shapes. *CNLS Newsletter*, 139:1–18, July 1997. 6

26. D. Pugh. Designing solid objects using interactive sketch interpretation. In *ACM Symposium on Interactive 3D Graphics*, pages 117–126, March 1992. 2

27. S.F. Qin, D. K. Wright, and I.N. Jordanov. From on-line sketching to 2D and 3D geometry: a system based on fuzzy knowledge. *Computer-Aided Design*, 32(14):851–866, 2000. 1

28. A. Sherstyuk. Kernel functions in convolution surfaces: A comparative analysis. *The Visual Computer*, 15(4):171–182, 1999. 3

29. J.R. Shewchuk. Triangle: engineering a 2D quality mesh generator and delaunay triangulator. *First Workshop on Applied Computational Geometry Proceedings*, pages 124–133, 1996. 6

30. O. Tolba, J. Dorsey, and L. McMillan. A projective drawing system. In *ACM Symposium on Interactive 3D Graphics*, pages 25–34, March 2001. 2

31. G. Turk and J.F. O'Brien. Modeling with implicit surfaces that interpolate. *ACM Transactions on Graphics*, 21(4):855–873, October 2002. 2

32. G. Wyvill, C. Mcpheeters, and B. Wyvill. Data structures for soft objects. *The Visual Computer*, 2(4):227–234, 1986. 8

33. R.C. Zeleznik, K.P. Herndon, and J.F. Hughes. Sketch: an interface for sketching 3D scenes. *ACM SIGGRAPH 1996 Conference Proceedings*, pages 163–170, 1996. 1

Appendix A: Approximating a sketched profile by polar-form Bezier curves

Let $PS = \{(R_j, \theta_j) | j = 1, 2, \ldots, p\}$ be a point sequence sampled from the input profile. We want to fit an n-degree polar-form Bezier curve $b(\theta)$. Assume that the end control radii (r_0, θ_b) and (r_n, θ_e) are known. We minimize the following function

$$f_{PS}(r_1, \ldots, r_{n-1}) = \sum_{j=1}^{p} (b(\theta_j) - R_j)^2 \tag{27}$$

Let $\mathbf{r} = [r_1, \ldots, r_{n-1}]^\top$ and $\mathbf{B}_i = [B_{n,i}(t_1), \ldots, B_{n,i}(t_p)]^\top$ (with $t_i = \frac{\theta_i - \theta_b}{\theta_e - \theta_b}$) be the i-th column of the Bezier coefficient matrix $\mathbf{B} = [\mathbf{B}_1, \ldots, \mathbf{B}_{n-1}]$. We rewrite Equation 27 into matrix form

$$\min_{\mathbf{r}} \|\mathbf{B} \cdot \mathbf{r} - \tilde{\mathbf{R}}\|_2^2 \tag{28}$$

where

$$\tilde{\mathbf{R}} = [R_1, \ldots, R_p]^\top - [\mathbf{B}_0\ \mathbf{B}_n] \begin{bmatrix} r_0 \\ r_n \end{bmatrix}.$$

Since it is a typical least-squares optimization problem, the solution is as follows:

$$\mathbf{r} = (\mathbf{B}^\top \mathbf{B})^{-1} \mathbf{B}^\top \tilde{\mathbf{R}}.$$

Appendix B: Derivative property of polar-form Bezier curves

Consider the following transformation of the polar-form Bezier curves

$$\begin{cases} x(\theta) &= R(\theta)\cos\theta, \\ y(\theta) &= R(\theta)\sin\theta. \end{cases} \tag{29}$$

By the chain rule, the first-order derivatives are

$$\begin{cases} x' &= R'(\theta)\cos\theta - R(\theta)\sin\theta, \\ y' &= R'(\theta)\sin\theta + R(\theta)\cos\theta. \end{cases} \tag{30}$$

From Equation 15

$$R'(\theta) = \frac{n}{\theta_e - \theta_s} \sum_{i=0}^{n-1} \triangle r_i B_{n,i}\left(\frac{\theta - \theta_s}{\theta_e - \theta_s}\right), \tag{31}$$

where $\triangle r_i = r_{i+1} - r_i$ is the forward difference operator. Then at the endpoints,

$$R'(\theta_s) = \frac{n}{\theta_e - \theta_s} \triangle r_0, \qquad R'(\theta_e) = \frac{n}{\theta_e - \theta_s} \triangle r_{n-1}$$

To achieve tangent continuity for two consecutive j and $j+1$ curve segments, we must satisfy

$$(x,y)|_{\theta_{j+1}-} = (x,y)|_{\theta_{j+1}+}, \qquad (x',y')|_{\theta_{j+1}-} = \alpha (x',y')|_{\theta_{j+1}+},$$

where $\alpha > 0$. By choosing $\alpha = 1$, we obtain the following sufficient condition:

$$\triangle r_{j,n-1} = \triangle r_{j+1,0}, \tag{32}$$

which is equivalent to Equation 17.