Generalized Sweep Templates for Implicit Modeling

Ryan Schmidt[*] Brian Wyvill[†]

University of Calgary, Canada

Abstract

A technique is presented for generating implicit sweep objects that support direct specification and manipulation of the surface with no topological limitations on the 2D sweep template. The novelty of this method is that the underlying scalar field has global properties which are desirable for interactive implicit solid modeling, allowing multiple sweep objects to be composed. A simple method for converting distance fields to bounded fields is described, allowing implicit sweep templates to be generated from any set of closed 2D contours (including "holes"). To avoid blending issues arising from gradient discontinuities, a general distance field approximation technique is presented which preserves sharp creases on the contour but is otherwise C^2 smooth. Flat endcaps are introduced into the 3D sweep formulation, which is implemented in the context of an interactive hierarchical implicit volume modeling tool.

1 Introduction

One of the strongest benefits of implicit modeling is that implicit volumes are trivial to compose using simple functions. In the context of hierachical implicit volume modeling frameworks such as the BlobTree [Wyvill et al. 1999], complex implicit models can be constructed by iteratively composing simpler volumes using solid modeling operations such as CSG and blending. A recent spatial caching technique for the BlobTree [Schmidt et al. 2005a] supports interactive modeling of complex implicit models.

Sweep surfaces, nearly ubiquitous in parametric modeling for the last 20 years [Requicha 1980], are a useful interactive modeling metaphor. However, sweep surface methods are very limited in the implicit domain [Bloomenthal 1997]. Existing implicit sweep representations compatible with the BlobTree are limited to star-shape sweep profiles defined by offset curves [Crespin et al. 1996]. Implicit sweep techniques that support the intuitive direct profile manipulation available in the parametric domain have not been previously developed for the BlobTree.

One of the key problems regarding implicit sweeps is the definition of a suitable 2D sweep template scalar field. The global properties of the scalar field must be considered. The BlobTree requires *bounded* scalar fields which have compact support. Gradient discontinuities in the field away from the surface must be avoided to prevent blending artifacts. Even when these conditions are met, the scalar field may still have subtle undesirable properties that can have a significant impact on blending [Barthe et al. 2003].

We describe a technique for generating 2D sweep template fields that can be used to create sweep volumes which have good blending properties and are compatible with the BlobTree (Section 3). First, we describe a simple technique for converting distance fields to bounded fields, allowing any set of closed 2D contours, including "holes", to be used to create the implicit sweep template. We then extend variational curve-fitting techniques to create a distance field approximation which is C^2 away from the surface, providing good blending properties, but preserves creases in the sweep template contour.

Our approach to 3D sweeps largely follows classical methods (Section 4). One issue which has not been previously discussed is the generation of flat sweep endcaps. We integrate a CSG intersection operation, based on the sweep trajectory parameter space, to create flat endcaps.

The goal of our sweep formulation is to create implicit sweep volumes compatible with the BlobTree. Within this framework, implicit sweeps can be composed with other implicit volumes (including other implicit sweeps). We have implemented these sweep primitives in an interactive BlobTree modeling environment and found them very useful for implicit solid modeling (Section 5). In interactive contexts, the ability to directly manipulate the surface contour is much more efficient than the previous offset-contour methods. The interaction techniques developed for parametric sweeps can be used without modification. Combined with their good blending behavior, our sweep primitives are an intuitive and expressive free-form addition to BlobTree modeling.

2 Related work

2.1 Hierarchical implicit volume modeling

Given a continuous scalar function $f : \mathbb{R}^3 \rightarrow \mathbb{R}_+$, we can define a volume $\mathcal{V}$:

$$\mathcal{V} = \left\{ \mathbf{p} \in \mathbb{R}^3 : f(\mathbf{p}) \geq v_{iso} \right\} \qquad (1)$$

where v_{iso} is called the *iso-value*. We call $\mathcal{V}$ an *implicit volume*. The surface $\mathcal{S}$ of this volume is defined by replacing the inequality in Equation 1 with an equality. This surface is referred to as an *implicit surface* [Bloomenthal 1997]. This definition also applies in 2D, where $\mathcal{S}$ is a contour and $\mathcal{V}$ is the enclosed area.

Two implicit volumes, defined by scalar functions f_1 and f_2, can be combined functionally using a composition operator $\mathcal{G}(f_1, f_2) \in \mathbb{R}_+$. Since $\mathcal{G}$ is also a scalar function, composition operators can be applied recursively. A variety of operators are available for performing Computational Solid Geometry (CSG), blending, space deformation, and more [Bloomenthal 1997].

Recursive application of composition operators results in a tree-like data structure with implicit volumes (*primitives*) at the leaves and composition operators at tree nodes. The final scalar field is evaluated at the root composition operator, which recursively evaluates its children, and so on. Using this framework, complex volumes

[*]e-mail: rms@cpsc.ucalgary.ca
[†]e-mail: blob@cpsc.ucalgary.ca

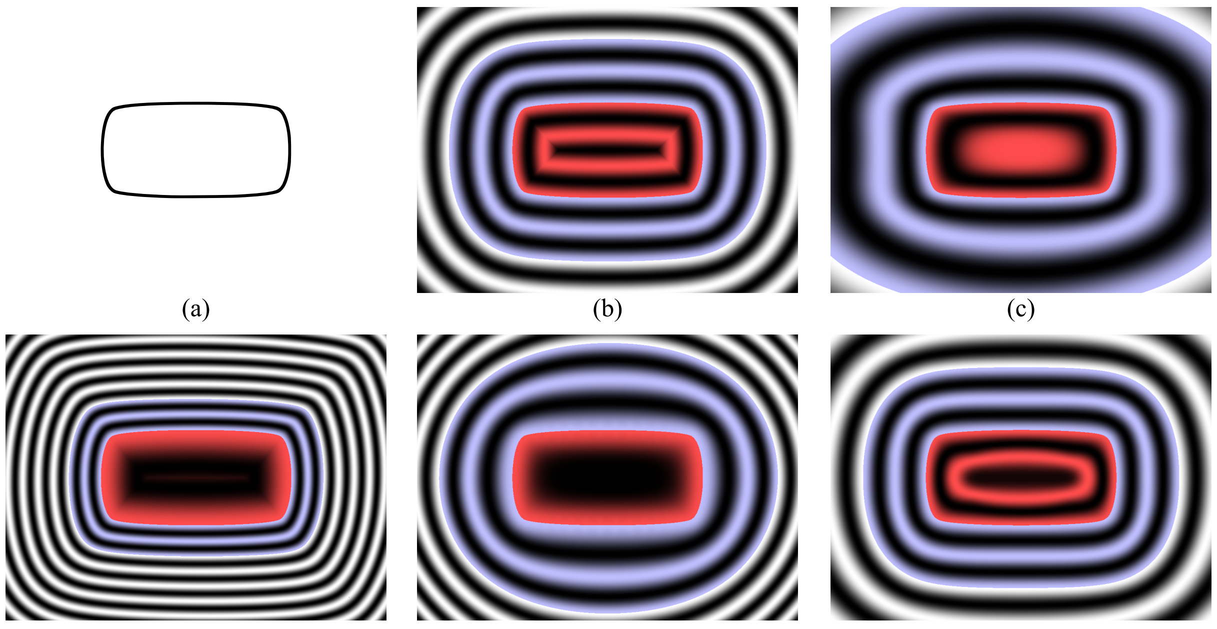


Figure 1: *Scalar fields generated from contour (a), with sin function applied before mapping to grayscale to emphasize iso-contours. Red highlight indicates region inside contour, blue highlight indicates bounded region after applying Equation 3. Exact distance field (b) has C^1 discontinuities inside contour. Variational approximation (c) using normal constraints provides a poor approximation to (b). Approximation with a C^1 implicit polygon (d) is sensitive to the tessellation of the curve, field values grow as tessellation level is increased. A normalized implicit polygon (e) is accurate near the contour, but the iso-contours quickly tend to a circular shape. Our variational approximation with boundary constraints (f) closely approximates (b) inside the falloff region but is globally C^2 continuous.*

can be incrementally constructed from simple components. This type of procedurally-defined hierarchical implicit volume model is often called a BlobTree [Wyvill et al. 1999].

A variety of benefits can be gained by requiring the scalar functions f which define volume primitives to be *bounded* [1]. A scalar field f is bounded if $f = 0$ outside some finite bounding volume. Operators $\mathcal{G}$ must also produce bounded fields. Bounded fields guarantee local influence, preventing changes made to a small part of a complex model from affecting distant portions of the surface. Local influence preserves a "principle of least surprise" that is critical for interactive modeling. Bounded scalar fields also reduce the cost of evaluating the BlobTree, since the field bounds at each node can be used to cull evaluations outside the non-zero region. In addition, bounded scalar fields are a pre-requisite for Hierarchical Spatial Caching [Schmidt et al. 2005a]. This approximation technique provides an order-of-magnitude reduction in visualization time, allowing complex hierarchical implicit volume models to be manipulated in real-time in interactive systems.

One type of implicit volume primitive with a bounded scalar field is the *skeletal primitive* [Bloomenthal and Wyvill 1990], defined by a geometric skeleton E (such as a point or line) and a one-dimensional function $g : \mathbb{R}_+ \to \mathbb{R}_+$. The scalar function f is then:

$$f_{\mathsf{E},g}(\mathbf{p}) = g \circ d_{\mathsf{E}}(\mathbf{p}) \qquad (2)$$

where d_{E} is a function that computes the minimum Euclidean distance from $\mathbf{p}$ to E. The shape of a skeletal primitive is primarily determined by E. We use the following function for g [Wyvill 2005]:

$$g_w(x) = (1-x^2)^3 \qquad (3)$$

where x is clamped to the range $[0,1]$. This polynomial smoothly decreases from 1 to 0 over the valid range, with zero tangents at each end. We choose 0.5 as the iso-value v_{iso}.

Finally we consider field continuity. At minimum, all scalar fields f must be C^0 continuous (implying that operators must preserve C^0 continuity). Under this condition it is impossible to create a scalar field that does not define a valid surface. C^1 (gradient) continuity is desirable because the gradient of the scalar field is used to calculate surface normals. C^1 discontinuities in the scalar field can result in the appearance of unexpected creases in blend surfaces (Figure 3).

2.2 Implicit sweep surfaces

Solid modeling by sweeping a 2D area (commonly referred to as a *sweep template*) along a 3D trajectory is a well known technique in computer graphics [Requicha 1980] [Foley et al. 1993]. Most early CAD systems [Requicha and Voelcker 1982] supported sweep surfaces as boundary representations (B-reps). These B-rep sweep solids lacked a robust mathematical foundation, self-intersecting sweeps were simply invalid. Recent work on the more general problem of sweeping a 3D volume has produced several general swept-volume envelope computation techniques which employ either the sweep-envelope differential equation [Blackmore et al. 1997] or Jacobian rank-deficiency conditions [Abdel-Malek and Yeh 1997]. Applications of these techniques are explored in a recent survey [Abdel-Malek et al. 2000a]. These methods operate

[1] We use the term *bounded* [Barthe et al. 2003], rather than *compact support*, in an attempt to draw an analogy to the concept of bounding boxes that is ubiquitous in computer graphics.

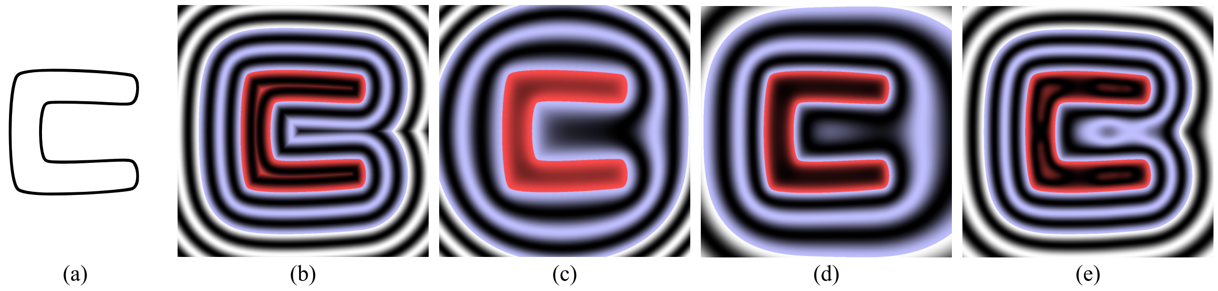


Figure 2: *Scalar fields generated using a non-convex curve (a). The exact distance field (b) has C^1 discontinuities inside and outside the curve. Approximation with a normalized implicit polygon (c) and variational approximation with normal constraints (d) provide poor approximations in concave region. Our approach (e) smoothly approximates the distance field away from the surface.*

on closed-form boundary representations which are symbolically manipulated. The volume boundaries can be defined using closed-form implicit surfaces [Abdel-Malek et al. 2000b]. However, the resulting swept volume is still a boundary representation, and not an implicit volume defined such as in Equation 1. [Sourin and Pasko 1996] describe a technique for sweeping implicit volumes. Evaluating the resulting function requires slow non-linear optimization algorithms.

There are essentially two approaches to creating implicit sweep surfaces. One method is to convert the 2D sweep template to a 2D scalar field, and then sweep this scalar field in 3D. Alternatively, a discretization of a 3D parametric sweep surface (such as a triangle mesh or point set) can be approximated with an implicit representation.

Creating an implicit representation of an arbitrary 3D surface is non-trivial. Some success in this domain has been achieved using *scattered-data interpolation* techniques, where an implicit surface is defined that passes through a set of 3D sample points. One approach, variational implicit surfaces [Turk and O'Brien 1999] [Savchenko et al. 1995], involves solving a dense matrix and hence does not scale to more than a few thousand sample points - insufficient to adequately represent a complex sweep surface. The cost of solving the linear system can be greatly reduced using Fast Multipole Methods, permitting millions of sample points to be used [Carr et al. 2001]. Unfortunately the multipole expansions necessary to implement this technique were not provided in [Carr et al. 2001] and have not been published. This algorithm is only available commercially, so we do not consider it an alternative. In addition, these approximation techniques cannot represent sharp creases or points, since the variational implicit surface is globally C^2 continous. Interpolation of point values is generalized in the concept of transfinite interpolation [Rvachev et al. 2001], which can interpolate arbitrary geometric curves and preserves discontinuities. However, this technique has only been demonstrated for a restricted set of closed-form 2D implicit contours.

Another class of scattered-data interpolation techniques have been developed that employ basis functions with local support [Morse et al. 2001]. The underlying linear systems are sparse and hence more efficient to solve. Partition of unity techniques such as MPU implicit surfaces [Ohtake et al. 2003] and the Partition of Unity Variational method [Reuter 2003] apply hierarchical approximation techniques to simplify surface fitting and can represent sharp features in certain configurations. However, these techniques are not guaranteed to define an implicit volume (Equation 1). Additional internal iso-surfaces may be produced as a result of blending local approximations with finite support [Reuter 2003]. Volume modeling operations such as CSG may expose these internal iso-surfaces.

The Implicit Moving-Least-Squares (IMLS) technique [Shen et al. 2004] converts triangle meshes into implicit volumes and hence could be used to define implicit sweep surfaces. However, the implicit surface interpolates the tessellation - the differential properties of the sweep template are lost, local curvature is always zero (or infinite on the creases between each triangle), and an adequate tessellation level must be determined.

[Sealy and Wyvill 1997] and [Winter and Chen 2002] approximate sweep surfaces with volume datasets. The sweep dataset is initialized by sampling a sweep template specified as a 2D image. Volume datasets cannot represent features smaller than a single voxel. Surface creases and sharp points are also lost in the conversion process. Adaptively-Sampled Distance Fields (ADFs) [Frisken et al. 2000] attempt to deal with these issues, but contain C^1 discontinuities in the reconstructed scalar field.

Implicit sweeps can alternatively be defined by following the parametric approach of sweeping a 2D template. One key issue is the definition of a suitable implicit template. A 2D scalar field must be defined that represents the closed 2D template contours. This scalar field is then swept along the sweep trajectory.

[Crespin et al. 1996] describes implicit sweep primitives with bounded sweep template fields. Profile curves defined in polar coordinates are used to create an anisotropic distance field. The sweep surface is an offset surface from the swept profile curve, prohibiting direct surface specification. Profile curves are limited to "star-shapes" by the polar definition. [Grimm 1999] describes implicit generalized cylinders which have a similar limitation.

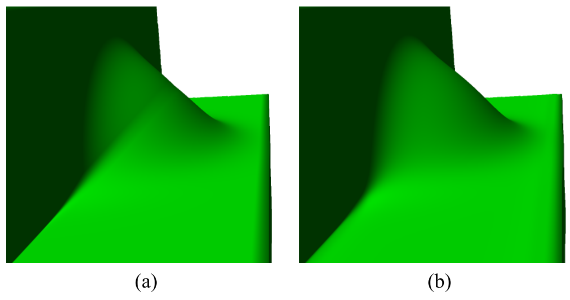


Figure 3: *An implicit sphere is blended with an L-shaped sweep primitive. A sweep template based on an exact distance field, which contains a C^1 discontinuity equidistant from the two flat surfaces, is swept in (a) and results in a crease on the blend surface. A sweep template created with our method is used in (b) and produces a smooth blend surface.*

Another approach is to approximate the contour with a polygon, and then construct an implicit representation of the polygon [Pasko et al. 1996b] [Barthe et al. 2003]. These methods require that non-convex polygons be carefully decomposed into pieces that can be properly combined using CSG intersection operations. This decomposition is non-trivial. The properties of the resulting scalar field are dependent on the tessellation level (Figure 1d). These issues can be mitigated using explicit normalization techniques [Biswas and Shapiro 2004]. A side-effect of this normalization is that offset iso-contours quickly converge to a circle (Figure 1e). While *Huygen's Principle* guarantees that all distance fields will converge to circular iso-contours in the limit, in this case it happens quickly and has an undesirable effect on blending.

A swept 2D implicit polygon results in a faceted 3D sweep surface. Curvature at the surface is not preserved, and gradient discontinuities exist along the sweep paths of each vertex in the 2D polygon. The 2D polygon can be smoothed at the vertices [Pasko et al. 1996b]. However, this "corner-cutting" reduces the accuracy of the approximation as the surface no longer passes through the accurate surface samples (the polygon vertices). An implicit representation of 2D curves has been described [Pasko et al. 1996a], but is based on an underlying "carrier polygon" that has only been developed for cubic polynomial splines. Variational interpolation can also be applied to approximate a 2D contour [Yngve and Turk 2002]. The standard technique involves constraining the variational solution only near the contour, the rest of the field is unconstrained and hence determining bounds is very difficult (Figure 1c).

2D implicit curves can be represented using level set techniques [Osher and Fedkiw 2002]. These algorithms are based on 2D pixel discretizations, and hence have the same issues as volume datasets regarding small and sharp features. Skeletonization techniques such as Medial Axis Transform [Blum 1967] can be used to compute distance fields which approximate curves but also contain C^1 discontinuities.

To summarize, several implicit sweep models have been proposed that support direct surface specification and manipulation. However, none of these models produce the bounded scalar fields necessary for interactive modeling of complex hierarchical implicit models [Wyvill et al. 1999] [Schmidt et al. 2005a].

3 2D implicit sweep templates

In our sweep formulation, and in hierarchical implicit modeling in general, the behavior of blending surfaces is largely determined by the global properties of the underlying scalar fields. For example, in Figure 3, two volumes are blended. In both cases the volumes are identical but the underlying scalar fields are different. One, based on a distance field, contains C^1 discontinuities in the scalar field away from the surface but inside the blending region, resulting in a crease on the blending surface [Barthe et al. 2003]. The other scalar field, created with our method, is continuous and blends smoothly.

In the following section we develop a 2D implicit sweep template that represents a closed 2D contour $\mathcal{C}$. First, we describe a technique for converting the distance field $d_{\mathcal{C}}$ to a bounded 2D field. This simple method allows BlobTree-compatible implicit sweep surfaces to be generated from any set of closed 2D contours. We then describe a method for creating a smooth approximation $\widetilde{d_{\mathcal{C}}}$ to the exact distance field $d_{\mathcal{C}}$ which does not contain undesirable C^1 discontinuities.. This approximation technique, which extends existing work on approximating curves with variational interpolation [Turk and O'Brien 1999], cannot reproduce sharp edges in $\mathcal{C}$. Hence, we describe a technique for re-introducing sharp edges into the sweep template, followed by a comparison of the distance field approximation with existing methods.

3.1 Bounding distance fields

In the parametric domain a sweep surface is generated by sweeping a 2D template curve $\mathcal{C}$ along a 3D trajectory $\mathcal{T}$. In the implicit domain, we must create a 2D template scalar field $f_{\mathcal{C}}$ that represents $\mathcal{C}$. Since we are creating 3D volumes, $\mathcal{C}$ must be a closed contour. Also, $f_{\mathcal{C}}$ should be bounded and continuous if the scalar field created by sweeping $f_{\mathcal{C}}$ is to be used in the BlobTree.

The approach used to create skeletal primitives (Section 2.1) - applying g_w to a distance field - can be adapted to create the sweep template field $f_{\mathcal{C}}$. However, in the distance field $d_{\mathcal{C}}$ the curve lies on the zero iso-contour, $d_{\mathcal{C}} = 0$. If g_w is applied directly to $d_{\mathcal{C}}$, the v_{iso} iso-contour in the bounded field will be offset from $\mathcal{C}$. To force this contour to lie on $\mathcal{C}$ the distance values must be shifted. We assume that $d_{\mathcal{C}}$ is signed, where a negative distance value indicates that the point is inside $\mathcal{C}$. The shifted distance $d'_{\mathcal{C}}$ is then:

$$d'_{\mathcal{C}} = min(g_w^{-1}(v_{iso}) + d_{\mathcal{C}}, 0) \tag{4}$$

The bounded 2D sweep template field $f_{\mathcal{C}}$ is then defined as

$$f_{\mathcal{C}} = g_w \circ d'_{\mathcal{C}} \tag{5}$$

This technique produces a scalar field which contains C^1 discontinuities (Figure 2a). As noted, these discontinuities will be swept in 3D and produce undesirable blending artifacts (Figure 3a). Hence it is necessary to develop an approximation to $d_{\mathcal{C}}$ that has a smooth field away from the surface.

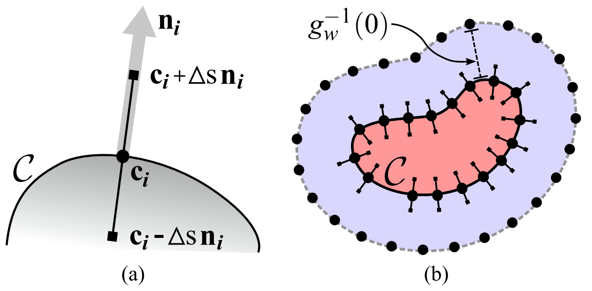
3.2 C^2 distance field approximation

An implicit approximation to a 2D contour $\mathcal{C}$ can be created using variational interpolation [Turk and O'Brien 1999]. A C^2 interpolating thin-plate spline is fit to a set of constraint points placed at samples of $\mathcal{C}$. To adequately constrain the solution, *normal constraints* [Carr et al. 2001] [Yngve and Turk 2002] are added. Inner and outer normal constraints at a sample point are added at short distances along the normal to $\mathcal{C}$ at the on-curve sample (Figure 4a).

Normal constraints only ensure that the iso-contour passes through the sample points. The scalar field is unconstrained further from the surface, resulting in a poor approximation to the distance field (particularly in the case of non-convex $\mathcal{C}$, see Figure 2). In our case this is especially problematic because it is non-trivial to determine the bounds of the non-zero field values (after applying g_w) without resorting to a time-consuming spatial search.

Variational approximation produces an implicit curve, rather than polygon. Since only sample points are necessary, there are no restrictions on $\mathcal{C}$. However, normal constraints are insufficient to produce a variational solution which approximates a distance field. Our approach is to add *boundary constraints* that constrain the variational solution in regions further from the curve (Figure 4b).

Boundary constraint points are generated by sampling several iso-contours of the distance field. A reasonable distance field approximation can be obtained using one set of boundary constraints along the contour found at a distance of $g_w^{-1}(0)$. After applying g_w, this contour bounds the non-zero field values, and hence can be used to determine a bounding box. We also use two additional iso-contour



$\mathbf{n}_i$
$\mathbf{c}_i + \Delta s\, \mathbf{n}_i$
$\mathcal{C}$
$\mathbf{c}_i$
$\mathbf{c}_i - \Delta s\, \mathbf{n}_i$
(a)
$g_w^{-1}(0)$
$\mathcal{C}$
(b)

Figure 4: Normal constraints *(a) at a point* $\mathbf{c}_i$ *are added at short offset* $\triangle$ *s from the curve* $\mathcal{C}$*, along the curve normal* $\mathbf{n}_i$. Boundary constraints *(b) are placed at a constant distance from* $\mathcal{C}$ *to improve the distance field approximation and ensure that the field* $f_\mathcal{C}$ *is bounded within a known distance.*

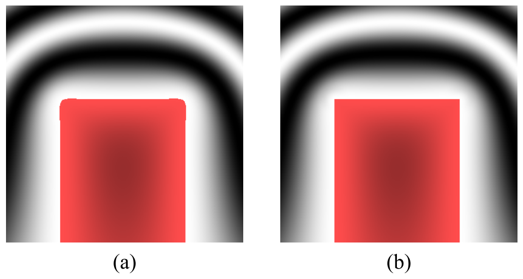


Figure 5: *The boundary-constrained variational method described in section 3.2 rounds out sharp edges (a) because it is globally* C^2 *continuous. Sharp features can be re-introduced (b) by blending between an implicit polygon and the variational field in the feature regions (see section 3.3).*

constraints, one at $g_w^{-1}(0.5 * v_{iso})$, which is approximately halfway between $\mathcal{C}$ and the zero-contour, and another at $g_w^{-1}(1.5 * v_{iso})$, which lies inside $\mathcal{C}$. The purpose of these extra constraints is to reduce error in the distance field approximation.

If $\mathcal{C}$ has high-frequency features, there may be higher error in the distance field approximation close to the curve. In implicit sweep applications, this error is not particularly critical. However, the situation can be identified automatically by comparing approximated distance values with exact distance values at small offsets from $\mathcal{C}$. If the measured error is unacceptable, additional sets of iso-contour constraints close to $\mathcal{C}$ can be inserted to further reduce approximation error.

Tracing iso-contours in the exact distance field $d_\mathcal{C}$ is non-trivial and computationally expensive. Instead we sample the iso-contour using a *distance transform* [Jain 1989] which approximates $d_\mathcal{C}$ on a discrete grid. We use the CSSED algorithm which runs in linear time in the number of image pixels [Cuisenaire and Macq 1999]. The distance transform is computed on a 512^2 pixel image and then discrete iso-contours are traced in the image. The resolution here is not critical, similar results have been found with 128^2 pixel images.

We have experimented with alternatives to this iso-contour-sampling constraint technique and found them lacking. Since the variational solution minimizes global curvature, constraints based on either random or regular sampling can result in unexpected oscillations. In addition, regular sampling introduces many spurious constraint points and if the sampling frequency is too high the discontinuities in the distance field are closely approximated. When these high-frequency areas are blended, *perceptual* discontinuities similar to those in Figure 3a can occur.

Adding boundary constraints to the variational solution results in a much more accurate approximation to the distance field, particularly in the case of non-convex curves (Figure 2). The variational scalar field is a globally C^2 continuous approximation to $d_\mathcal{C}$. The approximation accuracy can be improved by increasing the sampling rates used to generate constraint points.

3.3 Sharp features

The variational approximation to $\mathcal{C}$ is globally C^2 continuous. This implies that sharp features, which are C^1 discontinuities in $\mathcal{C}$, are not preserved in the variational approximation. Significant smoothing can be observed near creases, even with high sampling rates (Figure 5a). Crease approximation can be improved by using anisotropic basis functions [Dinh et al. 2001] but these basis functions do not create sharp C^1 discontinuities. Sharp features are preserved with implicit polygon techniques [Biswas and Shapiro 2004] [Barthe et al. 2003]. To introduce sharp features into our formulation, we take a constructive approach based on these existing techniques. We compute both our smooth distance field approximation $\widetilde{d_\mathcal{C}}$ and a crease-preserving approximation $\widehat{d_\mathcal{C}}$. These fields are then blended in sharp feature regions.

We assume that a set of crease positions $\mathbf{c}_i$ are given, as is a *feature radius* r_i. When evaluating $f_\mathcal{C}$ at a 2D point $\mathbf{u}$, we compute the distance k from $\mathbf{u}$ to the nearest feature point. The blended distance field approximation $d_\mathcal{C}^\star$ is then defined as follows:

$$d_\mathcal{C}^\star = \begin{cases} \widetilde{d_\mathcal{C}} & \text{if } k \le r \\ (1 - g_w(\frac{k}{r}))\widetilde{d_\mathcal{C}} + g_w(\frac{k}{r})\widehat{d_\mathcal{C}} & \text{if } r < k < 2r \\ \widehat{d_\mathcal{C}} & \text{if } k \ge 2r \end{cases} \tag{6}$$

See Figure 6 for a graphical representation of the blending regions. The use of g_w here is not significant, any smooth interpolating function can be used. This method is similar to the bounded-blending techniques used to control implicit volume composition operators [Pasko et al. 2002] [Galbraith et al. 2004]. An example is shown in Figure 5.

In our implementation we use the normalized implicit polygon approach [Biswas and Shapiro 2004] to define the crease-preserving field $\widehat{d_\mathcal{C}}$. The necessary equations are described in Appendix B. To locate crease points on $\mathcal{C}$ we use the rudimentary method of thresholding the angles between consecutive line segments in the polygonal approximation. This technique is not robust but works reasonably well.

This blending approach provides the benefits of boundary-constrained variational approximation while also locally preserving sharp features. Since our method provides a more accurate distance field approximation (Figure 1), it is beneficial even when the input contour is not curved. However, there are some drawbacks. In the blending regions around feature points the scalar field may not be monotonic, since the two fields being combined can be increasing at different rates. As previously noted, curvature in the feature region is not preserved by the polygonal approximation.

3.4 Analysis

A variety of implicit representations for general 2D curves have been developed (Section 2.2). The scalar fields generated by some

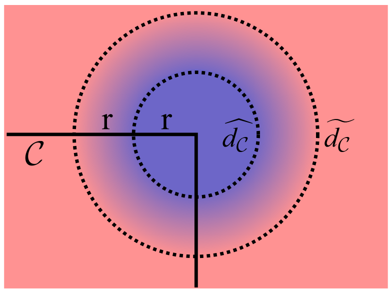


Figure 6: *A scalar field $\widehat{d_\mathcal{C}}$ that preserves sharp features in $\mathcal{C}$ can be combined with our smooth approximation $\widetilde{d_\mathcal{C}}$. Within a distance r from the sharp feature, $\widehat{d_\mathcal{C}}$ is used. Outside the radius* 2r*, $\widetilde{d_\mathcal{C}}$ is used. Between r and* 2r *(dashed circles in diagram), $\widehat{d_\mathcal{C}}$ and $\widetilde{d_\mathcal{C}}$ are smoothly blended.*

of these techniques are compared in Figures 1 and 2. In both of these examples it is clear that our technique more closely approximates a distance field than existing methods. Close approximation to a distance field is desirable because it allows easy computation of field bounds and results in more predictable blending behavior. The technique used to generate Figure 1d, for example, is sensitive to the tessellation level of the curve. If the tessellation level is increased, the iso-contour spacing will change and hence blend surfaces will also change. This is unacceptable. The normalized polygon approach [Biswas and Shapiro 2004] does not have this problem; however, the iso-contours produced approach a circular shape very quickly (Figure 2c). This results in undesirable blending behavior (Figure 7). In both of these cases, the scalar field represents a polygon, not a curve. The curvature of the input contour $\mathcal{C}$ is lost, and hence the differential surface properties of the implicit sweep do not reflect those that would be computed with a parametric sweep. Although it is still an approximation, our variational technique produces a curve and hence approximates the curvature of the accurate sweep surface.

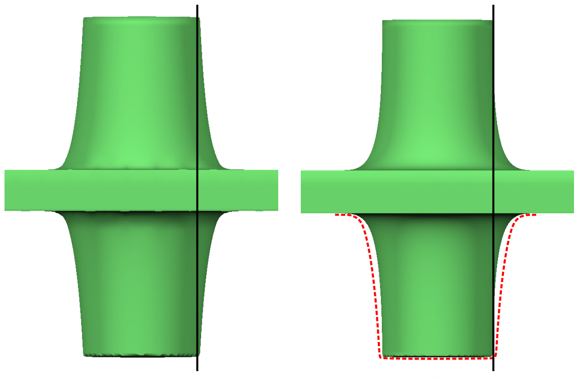


Figure 7: *Implicit sweep blended with cylinder. Sweep template is computed using normalized implicit polygon in (a), and our technique in (b). Black line shows edge of non-blended cylinder, dashed red line in (b) shows outline of (a). Blending in (b) is more localized and has a softer transition.*

Defining "goodness" measures for distance field approximation is challenging. One test is to compare the approximated field values with the exact distance field values. However, since we do not wish to reproduce the discontinuities found in distance fields, zero error is in fact undesirable. Still, we have performed this test with a variety of convex and non-convex curves, and found that, on average, our technique reduces the mean error by a factor of 25 over the normalized implicit polygon approach.

One characteristic of distance fields is that the magnitude of the gradient at all points is one (except along the discontinuity curves). This is related to the concept of normalization [Biswas and Shapiro 2004]. Again, if the goal is to smooth discontinuities in a distance field, minimizing the gradient magnitude error $|1-|\nabla f||$ is undesirable as it will produce high-frequency C^2 regions that are effectively "perceptual" discontinuities. However, we have compared our technique with the implicit polygon approach and found that it reduces the mean gradient error by a factor of 4.

Visually inspecting the gradient error map is instructive. Our technique is compared with the implicit polygon approach in Figure 8 by mapping the error values to grayscale. We see that in both cases the error is highest along what would be discontinuities in the exact distance field (inside $\mathcal{C}$, the discontinuities lie on the medial axis). In the implicit polygon approach (Figure 8a), the error is very low near $\mathcal{C}$ but increases rapidly away from the curve. Our approach has somewhat higher error near $\mathcal{C}$ but is much more accurate in the bounded approximation region further from the curve.

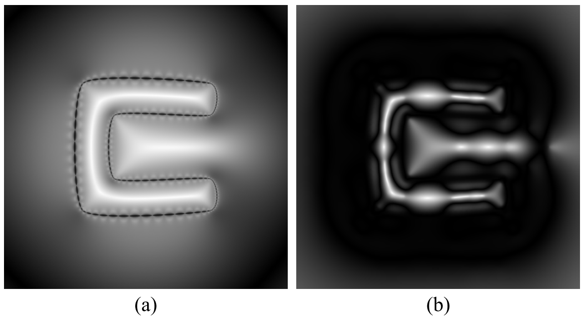


Figure 8: *Gradient images computed by mapping $|1-|\nabla f||$ to grayscale (white is maximum error). Normalized implicit polygon (a) has low error near surface, but higher error than our technique (b) in regions further from the surface.*

4 Sweep primitives

An implicit sweep primitive is defined by a 2D *sweep template field* $f_\mathcal{C}$ (Equation 5) and a 3D *sweep trajectory* $\mathcal{T}$. We will assume that $\mathcal{T}$ is a parametric function:

$$\mathcal{T}(s) = (t_x(s), t_y(s), t_z(s)), \qquad 0 \le s \le 1$$

The sweep surface $\mathcal{S}$ is defined by a 3D scalar field $f_\mathcal{S}$. Given a point $\mathcal{T}(s)$ on the sweep trajectory, we can define a geometric transformation $\mathbf{F}(s)$ that maps 2D points $\mathbf{u}$ to 3D points $\mathbf{p}$. We assume that $\mathbf{F}(s)$ is affine and maps the 2D origin to $\mathcal{T}(s)$, implying that points $\mathbf{F}(s)\cdot\mathbf{u}$ are co-planar (Figure 9a).

A possible definition for $f_\mathcal{S}$ is:

$$f_\mathcal{S}(\mathbf{p}) = f_\mathcal{C}(\mathbf{u}), \qquad \mathbf{F}(s)\cdot\mathbf{u} = \mathbf{p} \tag{7}$$

Unfortunately, given only a point $\mathbf{p}$ we cannot directly evaluate this equation because the parameter value s and 2D point $\mathbf{u}$ are unknown. Since $\mathbf{F}(s)$ is affine the inverse mapping $\mathbf{F}^{-1}(s)$ is:

$$f_\mathcal{S}(\mathbf{p}) = f_\mathcal{C}(\mathbf{u}), \qquad \mathbf{u} = \mathbf{F}^{-1}(s)\cdot\mathbf{p} \tag{8}$$

However, s is still unknown, and not necessarily unique (Figure 9).

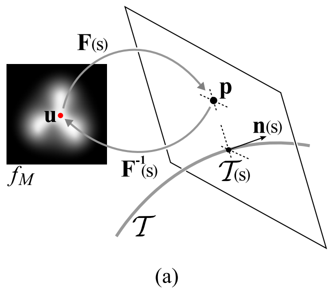
$\mathbf{F}(s)$ transforms the 2D plane into some 3D plane passing through $\mathcal{T}(s)$ with normal $\mathbf{n}(s)$ (Figure 9). Since there may be more than one plane that passes through $\mathbf{p}$ (Figure 9b), there exists a set of parameter values $\mathbb{S}(\mathbf{p}) = s_i$ such that $\mathbf{p}$ lies in the plane at s_i:

$$\mathbb{S}(\mathbf{p}) = \{s_i : (\mathbf{p} - \mathcal{T}(s)) \cdot \mathbf{n}(s) = 0\} \tag{9}$$

To compute the field value $f_{\mathcal{S}}(\mathbf{p})$, Equation 8 is evaluated for each parameter value $s_i \in \mathbb{S}(\mathbf{p})$. The resulting field values are combined with a composition operator $\mathcal{G}$, such as a CSG union operator [Ricci 1973], to produce a single field value. The final definition:

$$f_{\mathcal{S}}(\mathbf{p}) = \mathcal{G}\left(\left\{ f_{\mathcal{C}}(\mathbf{F}^{-1}(s_i) \cdot \mathbf{p}) \ : \ s_i \in \mathbb{S}(\mathbf{p}) \right\} \right) \tag{10}$$

can be evaluated for any trajectory where the parameter set $\mathbb{S}(\mathbf{p})$ is computable.



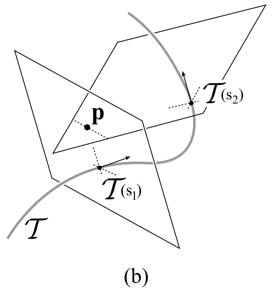


Figure 9: *Forward and inverse mapping (a) from sweep template field $f_{\mathcal{C}}$ to 3D space. Function $\mathbf{F}(s)$ maps a 2D point $\mathbf{u}$ to a 3D point $\mathbf{p}$ lying in plane defined by $\mathcal{T}(s)$ and $\mathbf{n}(s)$. The inverse map $\mathbf{F}^{-1}(s)$ maps from $\mathbf{p}$ to $\mathbf{u}$. In (b), Point $\mathbf{p}$ lies on the planes at points $\mathcal{T}(s_1)$ and $\mathcal{T}(s_2)$. A unique inverse mapping $\mathbf{F}^{-1}(s)$ does not exist.*

4.1 Sweep trajectories

Analytically solving Equation 10 for linear and circular trajectories is trivial [Winter and Chen 2002]. However, in the case of a general curved trajectory $\mathcal{T}(s)$, a closed-form solution is rarely possible. Assuming that the normal function $\mathbf{n}(s)$ (Equation 9) is the tangent vector $\mathcal{T}'(s)$, then the roots of the following equation must be found to compute the parameter value set $\mathbb{S}(\mathbf{p})$:

$$(\mathbf{p} - \mathcal{T}(s)) \cdot \mathcal{T}'(s) = 0 \tag{11}$$

For quadratic parametric curves such as a quadratic Bezier curve, this equation is a degree 3 polynomial which can be solved. However, these curves are planar and hence quite limited. The polynomial is degree 5 for cubic Bezier curves, precluding analytic solution. Numerical root-finding techniques [Schneider 1990] must be used to solve for the roots s.

Once $\mathbb{S}(\mathbf{p})$ is computed we can evaluate $\mathbf{F}_{\mathcal{T}}^{-1}(s)$:

$$\mathbf{F}_{\mathcal{T}}^{-1}(s) = \mathbf{Rot}\left[\mathbf{k}_1 \quad \mathbf{k}_2 \quad \mathcal{T}'(s)\right] \cdot \mathbf{Tr}\left[-\mathcal{T}(s)\right]$$

Care needs be taken when defining the vectors $\mathbf{k}_1$ and $\mathbf{k}_2$. Rotation-minimizing frames [Bloomenthal 1990] are suitable but cannot be computed analytically at an arbitrary parameter s. Instead we precompute a set of rotation-minimized frames along the curve at parameter values s_j, and then calculate the rotation-minimized frame at an arbitrary s based on the nearest stored frame at $s_j < s$. This technique is known as Bishop framing [Bishop 1975].

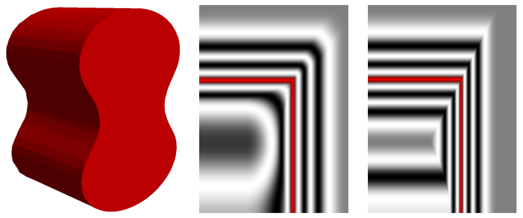


Figure 10: *Flat endcap with sharp transition. C^1 discontinuity on surface produces sharp crease (left) but does not propagate away from the surface with Barthe CSG intersection (middle). Ricci intersection produces C^1 discontinuity away from surface (right). Isocontour is marked in red.*

4.2 Endcaps

Open trajectories must be explicitly capped to ensure that a closed volume is defined. In previous works, sweeps were capped by interpolating the field to zero. This produces a variable-width endcap which may be undesirable. In theory, a flat endcap can be achieved in these systems by applying a CSG difference operation with a primitive such as an infinite plane or cube. However, there are many cases where it is likely that the CSG operation will have undesirable effects on other portions of the sweep. For instance, in Figure 11b it would be extremely difficult to define CSG operations that produced flat endcaps, if the endcaps were rounded.

We create flat endcaps by integrating a CSG intersection into the sweep formulation in parameter space. During each field evaluation, a CSG intersection with an infinite plane is applied to the computed field values. The field value for the infinite plane is not determined using the 3D position $\mathbf{p}$ but instead the parameter s. When s lies outside the range used to define the curve (typically [0,1]), then $\mathbf{p}$ is "outside" the infinite plane, otherwise it is inside. We use a CSG intersection operator that preserves C^1 continuity in the field away from the surface [Barthe et al. 2003], instead of the standard min/max CSG operations [Ricci 1973] which create C^1 discontinuities (Figure 10).

4.3 Self intersection

A key problem with sweep surfaces is that they are frequently self-intersecting (Figure 11). In the B-rep domain self-intersection is a critical problem [Requicha and Voelcker 1982], since the definition of 'inside' and 'outside' is often based on the sweep surface and hence is ambiguous inside self-intersections. These problems are not an issue in volumetric implicit modeling. Since a point in space has only one field value, there is no ambiguity as to whether or not it lies inside the volume.

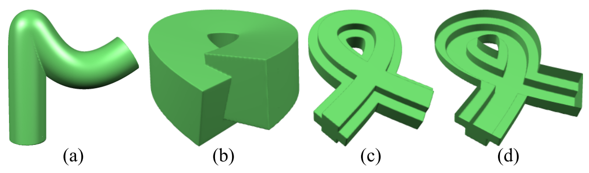


Figure 11: *Self intersection can occur near high-curvature regions of the sweep trajectory (a) as well as in overlapping regions (b), (c). Cutaway mesh in (d) shows manifold surface.*

The behavior of volumetric implicit sweeps in self-intersecting cases is entirely determined by the operator $\mathcal{G}$ (Equation 10). $\mathcal{G}$

should be applied to all field values produced with the solutions to Equation 11, as well as the endcap field values. The CSG Union operator of [Ricci 1973] can be used as $\mathcal{G}$. This produces a closed manifold surface but also gradient discontinuities in the scalar field (Figure 12). A C^1 continuous CSG union operator [Barthe et al. 2005] can be used to preserve both the surface creases and gradient continuity in the field away from the surface. Using this formulation, self-intersections are automatically handled. The final implicit volume is always defined as the outermost surface of the sweep.

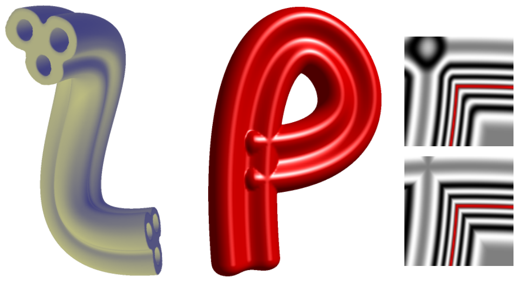


Figure 12: *Bezier curve sweeps. Holes in template are supported (left). Scalar field in self-intersecting cases (middle) is C^1 continuous using Barthe CSG union operator (right top). Ricci union operator produces C^1 discontinuities (right bottom).*

5 Modeling with sweeps

While a wide variety of volume primitives have been developed for hierarchical implicit volume modeling with BlobTrees, most are based on the skeletal primitive approach. Even with free-form skeletons, the user must model indirectly, manipulating an offset surface by changing the skeleton. The volumetric implicit sweep formulation we have presented allows the sweep profile to be directly specified, providing a useful free-form primitive for the BlobTree.

Visualizing BlobTree models is computationally expensive, and hence BlobTree modeling systems in the past have largely been offline, script-based systems. A recently-developed hierarchical spatial caching scheme [Schmidt et al. 2005a] allows BlobTree modeling to be performed interactively. We have implemented our implicit sweep primitives in this interactive system and found them extremely useful for implicit modeling. In an interactive context, the ability to directly specify the sweep profile can significantly reduce the time required to model a shape, since the trial-and-error involved with indirect manipulation is unnecessary. Our sweep surfaces, combined with interactive blending and CSG, can be used to quickly create complex 3D models (Figure 13).

We have used an earlier version of this sweep formulation to develop a linear sweep with rounded, variable-width endcaps. This primitive forms the basis of an interactive sketch-based BlobTree modeling tool [Schmidt et al. 2005b]. The more accurate distance field approximation presented here significantly improves blending control in this system. Figure 14b was created in this system by sketching linear sweeps and surfaces of revolution.

To visualize sweep surfaces in an interactive context, we approximate $f_{\mathcal{C}}$ with a discrete *field image*. This significantly reduces the computational expense of evaluating the scalar field for a sweep primitive. The approximation error that is introduced is acceptable for interactive visualization. More details can be found in an existing technical report [Schmidt and Wyvill 2005].

6 Conclusion

We have described techniques for generating 3D implicit sweep volumes that are compatible with the BlobTree hierarchical modeling system. Our main contribution is the development of a method for converting a 2D sweep template contour $\mathcal{C}$ (or set of contours, including "holes") into a bounded, continuous 2D scalar field which is used as an implicit sweep template. First, a general technique for converting 2D distance fields to bounded 2D scalar fields was presented. This was followed by the addition of *boundary constraints* to existing variational curve approximation techniques, which resulted in a C^2 distance field approximation. A constructive field-blending approach was described which preserves sharp features in the sweep template. Our approach to sweeping the 2D sweep template field largely follows existing methods, although the integration of flat endcaps into the sweep formulation is a necessary detail that has not been previously discussed.

Compatibility with BlobTree hierarchical implicit modeling is a key benefit of our sweep volumes. Existing BlobTree-compatible implicit sweeps [Crespin et al. 1996] required indirect sweep profile specification using offset contours. Other implicit sweep formulations which support direct profile specification are not usable with the BlobTree because they lack the necessary scalar field properties. Our sweep formulation combines the desirable attributes of these previous approaches. The BlobTree provides a rich infrastructure for shape modeling by procedural composition of implicit volumes, but has generally lacked free-form primitives that support direct surface specification. Particularly in an interactive context, our method provides an expressive, intuitive free-form primitive for the BlobTree. Complex shapes such as the set of pipes in Figure 14a can be quickly created by blending multiple sweep surfaces.

Preliminary analysis of our smooth distance field approximation has shown that it is significantly more accurate than existing techniques in regions away from the curve. For implicit sweeps, this is beneficial because blending behavior is more predictable and it is possible to determine a bounding box for the bounded scalar field without resorting to expensive spatial searching. Distance fields are used in many other applications, and some situations where a smooth distance field approximation may be desirable have been identified [Biswas and Shapiro 2004]. Further study of the properties of our distance field approximation is warranted to determine if it can be applied in these cases.

A limitation of the sweep technique we have described is that numerical root-finding is usually necessary in the case of curved trajectories. Root finding is expensive and requires an analytic expres-

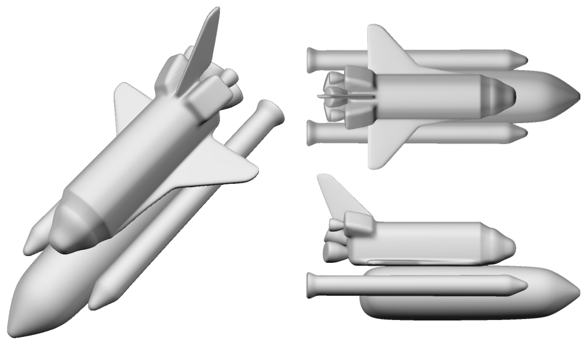


Figure 13: *Space shuttle model created by blending implicit sweeps and revolutions. This model was created in under an hour using our interactive system.*

sion for the curve tangent to function robustly. A key avenue for future work is a more general implicit sweep formulation that does not rely on root-finding. Since we are working in a volumetric context, a constructive approach based on approximation by union of linear segments can be employed. However, this technique breaks down in regions with high curvature. Approximation by union of simpler curved segments could provide superior results.

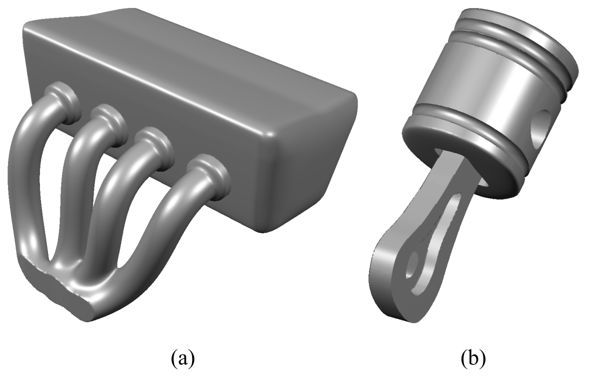


Figure 14: *Components of an engine modeled interactively using implicit sweep surfaces. Complex branching structures (a) can be created quickly by blending implicit sweeps. Hierarchical implicit CSG operations preserve sharp edges (b) and allow model components (including holes) to be easily manipulated or deleted.*

7 Acknowledgements

The authors would like to thank the reviewers for their valuable comments. This work was supported by the National Science and Engineering Research Council of Canada and iCore.

References

ABDEL-MALEK, K., AND YEH, H.-J. 1997. Geometric representation of the swept volume using jacobian rank-deficiency conditions. *Computer-Aided Design 29*, 6, 457–468.

ABDEL-MALEK, K., BLACKMORE, D., AND JOY, K. 2000. Swept volumes: Foundations, perspectives and applications. *submitted to International Journal of Shape Modeling*.

ABDEL-MALEK, K., YANG, J., AND BLACKMORE, D. 2000. Closed-form swept volume of implicit surfaces. In *Proceedings of ASME Design Engineering Technical Conferences*.

BARTHE, L., DODGSON, N. A., SABIN, M. A., WYVILL, B., AND GAILDRAT, V. 2003. Two-dimensional potential fields for advanced implicit modeling operators. *Computer Graphics Forum 22*, 1, 23–33.

BARTHE, L., WYVILL, B., AND DE GROOT, E. 2005. Controllable binary csg operators for soft objects. *International Journal of Shape Modeling*. (To Appear).

BISHOP, R. L. 1975. There is more than one way to frame a curve. *American Mathematical Monthly 82*, 3, 246–251.

BISWAS, A., AND SHAPIRO, V. 2004. Approximate distance fields with non-vanishing gradients. *Graphical Models 66*, 3, 133–159.

BLACKMORE, D., LEU, M., AND WANG, L. 1997. The sweep-envelope differential equation algorithm and its application to nc machining verification. *Computer-Aided Design 29*, 9, 629–637.

BLOOMENTHAL, J., AND WYVILL, B. 1990. Interactive techniques for implicit modeling. In *Proceedings of the 1990 symposium on Interactive 3D graphics*, 109–116.

BLOOMENTHAL, J. 1990. *Graphics Gems*. Morgan Kaufmann, ch. Calculation of references frames along a space curve, 567–571.

BLOOMENTHAL, J., Ed. 1997. *Introduction to Implicit Surfaces*. Morgan Kaufmann Publishers Inc.

BLUM, H. 1967. A transformation for extracting new descriptors of shape. In *Models for the Perception of Speech and Visual Form*, W. Wathen-Dunn, Ed. MIT Press, 362–380.

CARR, J. C., BEATSON, R. K., CHERRIE, J. B., MITCHELL, T. J., FRIGHT, W. R., MCCALLUM, B. C., AND EVANS, T. R. 2001. Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of ACM SIGGRAPH 2001*, 67–76.

CRESPIN, B., BLANC, C., AND SCHLICK, C. 1996. Implicit sweep objects. *Computer Graphics Forum 15*, 3 (Aug.), 165–174.

CUISENAIRE, O., AND MACQ, B. 1999. Fast and exact signed euclidean distance transformation with linear complexity. In *Proceedings of ICASSP 99*, vol. 6, 3293–3296.

DINH, H., SLABAUGH, G., AND TURK, G. 2001. Reconstructing surfaces using anisotropic basis functions. In *International Conference on Computer Vision*, 606–613.

DUCHON, J. 1977. *Constructive Theory of Functions of Several Variables*. Springer-Verlag, ch. Splines minimizing rotation-invariant semi-norms in Sobolev spaces, 85–100.

FOLEY, J. D., VAN DAM, A., FEINER, S. K., HUGHES, J. F., AND PHILLIPS, R. 1993. *Introduction to Computer Graphics*. Addison-Wesley.

FRISKEN, S., PERRY, R., ROCKWOOD, A., AND JONES, T. 2000. Adaptively sampled distance fields: A general representation of shape for computer graphics. In *Proceedings of SIGGRAPH 2000*, 249–254.

GALBRAITH, C., MUENDERMANN, L., AND WYVILL, B. 2004. Blobtree trees. *Computer Graphics Forum 23*, 3, 351–360.

GRIMM, C. 1999. Implicit generalized cylinders using profile curves. In *Implicit Surfaces 99*, 33–41.

JAIN, A. 1989. *Fundamentals of Digital Image Processing*. Prentice-Hall, ch. 2.

MORSE, B., YOO, T., RHEINGANS, P., CHEN, D., AND SUBRAMANIAN, K. 2001. Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. In *Proceedings of Shape Modeling International*, 89–98.

OHTAKE, Y., BELYAEV, A., ALEXA, M., TURK, G., AND SEIDEL, H.-P. 2003. Multi-level partition of unity implicits. *ACM Transactions on Graphics 22*, 3, 463–470.

OSHER, S. J., AND FEDKIW, R. P. 2002. *Level Set Methods and Dynamic Implicit Surfaces*. Springer.

PASKO, A., SAVCHENKO, A., AND SAVCHENKO, V. 1996. Implicit curved polygons. Tech. Rep. University of Aizu, Japan, Technical Report 96-1-004.

PASKO, A., SAVCHENKO, A., AND SAVCHENKO, V. 1996. Polygon-to-function conversion for sweeping. In *Proceedings of Implicit Surfaces 96*, 163–171.

PASKO, G., PASKO, A., IKEDA, M., AND KUNII, T. 2002. Bounded blending operations. In *International Conference on Shape Modeling and Applications*, 95–103.

REQUICHA, A. A. G., AND VOELCKER, H. B. 1982. Solid modeling: a historical summary and contemporary assessment. *IEEE Computer Graphics and Applications 2*, 2, 9–24.

REQUICHA, A. A. G. 1980. Representations for rigid solids: theory, methods and systems. *Computing Surveys 12*, 4, 437–464.

REUTER, P. 2003. *Reconstruction and Rendering of Implicit Surfaces from Large Unorganized Point Sets*. PhD thesis, LABRI - Universite Bordeaux.

RICCI, A. 1973. A constructive geometry for computer graphics. *Computer Graphics Journal 16*, 2, 157–160.

RVACHEV, V. L., SHEIKO, T. I., SHAPIRO, V., AND TSUKANOV, I. 2001. Transfinite interpolation over implicitly defined sets. *Computer Aided Geometric Design 18*, 4, 195–220.

SAVCHENKO, V., PASKO, A., OKUNEV, O., AND KUNII, T. 1995. Function representation of solids reconstructed from scattered surface points and contours. *Computer Graphics Forum 14*, 4, 181–188.

SCHMIDT, R., AND WYVILL, B. 2005. Implicit sweep surfaces. Tech. Rep. 2005-778-09, University of Calgary.

SCHMIDT, R., WYVILL, B., AND GALIN, E. 2005. Interactive implicit modeling with hierarchical spatial caching. In *Proceedings of International Conference on Shape Modeling and Applications (SMI 2005)*, 104–113.

SCHMIDT, R., WYVILL, B., SOUSA, M. C., AND JORGE, J. A. 2005. Shapeshop: Sketch-based solid modeling with blobtrees. In *Proceedings of the 2nd Eurographics Workshop on Sketch-Based Interfaces and Modeling*. to appear.

SCHNEIDER, P. J. 1990. *Graphics Gems*. Morgan Kaufmann, ch. A Bezier curve-based root finder, 409–415.

SEALY, G., AND WYVILL, G. 1997. Representing and rendering sweep objects using volume models. In *Computer Graphics International 1997*, 22–27.

SHEN, C., O'BRIEN, J. F., AND SHEWCHUK, J. R. 2004. Interpolating and approximating implicit surfaces from polygon soup. In *Proceedings of ACM SIGGRAPH 2004*, 896–904.

SOURIN, A., AND PASKO, A. 1996. Function representation for sweeping by a moving solid. *IEEE Transactions on Visualization and Computer Graphics 2*, 1, 11–18.

TURK, G., AND O'BRIEN, J. 1999. Shape transformation using variational implicit functions. In *Proceedings of SIGGRAPH 99*, 335–342.

WINTER, A. S., AND CHEN, M. 2002. Image-swept volumes. *Computer Graphics Forum 21*, 3, 441–450.

WYVILL, B., GUY, A., AND GALIN, E. 1999. Extending the csg tree. warping, blending and boolean operations in an implicit surface modeling system. *Computer Graphics Forum 18*, 2, 149–158.

WYVILL, G., 2005. Wyvill function. personal communication.

YNGVE, G., AND TURK, G. 2002. Robust creation of implicit surfaces from polygonal meshes. *IEEE Transactions on Visualization and Computer Graphics 8*, 4, 346–359.

Appendix A Variational Implicit Curves

One useful technique for generating a 2D scalar field is to interpolate a set of 2D field value samples $(\mathbf{m}_i, v_i)$, where v_i is the desired field value at point $\mathbf{m}_i$. We use a variational interpolation scheme based on thin-plate splines which is globally C^2 continuous. This scattered data interpolation technique has been used define implicit curves and surfaces [Turk and O'Brien 1999].

The thin-plate spline $f(\mathbf{u})$ is defined in terms of points $(\mathbf{m}_i, v_i)$ weighted by coefficients w_i, and a polynomial $\mathcal{P}(\mathbf{u}) = c_1\mathbf{u}_x + c_2\mathbf{u}_y + c_3$:

$$f(\mathbf{u}) = \sum w_i \|\mathbf{u} - \mathbf{m}_i\|^2 \ln(\|\mathbf{u} - \mathbf{m}_i\|) + \mathcal{P}(\mathbf{u}) \tag{12}$$

The weights w_i and coefficients c_1, c_2, and c_3 are found by solving a linear system defined by evaluating Equation 12 at each known solution $f(\mathbf{m}_i) = v_i$. These coefficients determine a variational solution which is guaranteed to interpolate all sample points $(\mathbf{v}_i, v_i)$ with C^2 continuity while minimizing global curvature [Duchon 1977].

Appendix B Normalized Implicit Polygons

A 2D closed contour $\mathcal{C}$ can be represented by an implicit polygon defined using the following approach (reproduced from [Biswas and Shapiro 2004]). First, $\mathcal{C}$ is approximated with a set of line segments. Then for a segment $((x_1, y_1), (x_2, y_2))$, with $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$, a scalar field can be defined:

$$f(x,y) = \frac{1}{d}\left((x - x_1)(y_2 - y_1) - (y - y_1)(x_2 - x_1)\right) \tag{13}$$

A circular scalar field is also defined for the line segment:

$$t(x,y) = \frac{1}{d}\left[\left(\frac{d}{2}\right)^2 - (x - \frac{x_1 + x_2}{2})^2 - (y - \frac{y_1 + y_2}{2})^2\right] \tag{14}$$

This circular field is used to normalize the line segment field, defining a new scalar field h:

$$h = \sqrt{f^2 + 0.25 * (|t| - t)^2} \tag{15}$$

These normalized line segments are then combined using the R-conjunction operator:

$$F(h_1, h_2) = h_1 + h_2 - \sqrt{h_1^2 + h_2^2} \tag{16}$$

The resulting field is unsigned, but since $\mathcal{C}$ is closed a polygon point-containment test can be used to determine whether a distance value should be positive or negative.