Implicit Modelling Using Subdivision-curves

Samuel Hornus, Alexis Angelidis, Marie-Paule Cani

Implicit Modeling using Subdivision-curves

Samuel Hornus, Alexis Angelidis and Marie-Paule Cani

iMAGIS[†]-GRAVIR, INRIA Rhône-Alpes
655 avenue de l'Europe, 38330 Montbonnot, France
{Alexis Angelidi, Marie-Paule.Cani, Samuel.Hornus}@imag.fr
http://www-imagis.imag.fr/Membres/

Categories and subject descriptors: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling–Curve, surface, solid and object representations;

Additional Key Words and Phrases: Convolution Surfaces, Levels of Detail, Implicit Surfaces.

Abstract

To remain an attractive model, skeleton-based implicit surfaces have to allow the design and display of shapes at interactive rates. This paper focuses on surfaces whose skeletons are graphs of interconnected curves. We present subdivision-curve primitives that rely on convolution for generating bulge-free and crease-free implicit surfaces. These surfaces are efficiently yet correctly displayed using local meshes around each curve that locally overlap in blending regions. Subdivision-curve primitives offer a practical solution to the unwanted-blending problem that ensures C^1 continuity everywhere. Moreover, they can be used to generate representations at different levels of detail, enabling the interactive display of at least a coarse version of the objects, whatever the performance of the workstation. We also present a practical solution to the unwanted blending problem, used to avoid blending between parts of the surface that do not correspond to neighbouring skeletal elements.

1 Introduction

Implicit surfaces are defined as the set of points P verifying an equation $F(P) = c$, where F is a scalar "field function", and c is a given iso-value. The advantages of this representation are well-known [4] : these surfaces surround a closed volume, whose in-out function eases the computation of intersections along rays or with other objects. Shapes of any topology can be easily created using several primitives that blend their field function contributions. Moreover, local and global deformations can be incorporated in the construction tree.

However, although they have been introduced in Computer Graphics for many years now [2, 16, 26], implicit surfaces are not as popular as parametric surfaces for performing modeling tasks. One of the main problems is the lack of parameterization which makes the interaction with implicit shapes very difficult. The second problem is the high cost, in the general case, of the calculation of the field function that generates the surface. This can be easily understood : the theory tells us that any 3D shape can be represented by a skeleton, defined as the locus of the centers of the maximal spheres included inside the object [1]. This skeleton is a graph of interconnected curves and surface patches. Generating implicit surfaces without unwanted creases and bulges from such complex skeletons requires the use of convolution, which is known to be computationally expensive [5, 21].

This paper presents a solution to the interactive modeling and display of implicit shapes whose skeletons are graphs of branching curves. As shown in [24], this restricted set of skeletons still covers a number of useful cases. Our solution relies on convolution surfaces and on levels of detail (LOD) for defining subdivision-curve implicit primitives that can be displayed at interactive rates. A previous version of this work appeared in [7]. In this new version, we improve the method by defining a kernel that provides a closed-from solution for convolution surfaces that have a varying radius along their skeleton. We also describe the use of our method in an interactive modelling software, where the user can create and edit implicit surfaces defined by subdivision curves, at interactive rates.

Section 2 discusses related works. Our first contri-

[†]iMAGIS is a joint project of CNRS, INRIA, Institut National Polytechnique de Grenoble and Université Joseph Fourier.

bution, described in Section 3, is the presentation of the subdivision-curve primitive, which uses convolution for allowing the definition of implicit shapes at different levels of detail. In the same section, we also present an efficient convolution kernel for fast rendering of the implicit surface. Section 4 discusses ways to compute surfaces with varying radius; the second solution we propose is a new closed-form convolution kernel that allow the exact computation of convolution surfaces with varying radius along a skeleton curve, which is a new contribution. Section 5 describes a simple method for the interactive display of blended subdivision-curve primitives, based on locally-overlapping meshes. Section 6 shows that the subdivision-curve representation offers a practical solution to the unwanted blending problem, which ensures C^1 continuity everywhere. Sections 7 presents the integration of these techniques in an interactive modelling software. Section 8 concludes and discusses the possible extensions of this work.

2 Related works

2.1 Modeling complex shapes with implicit surfaces

In implicit modelling, the choice of the most efficient way to model a given 3D shape is a recurring question : What is the most efficient way to model a given 3D shape ? Should we blend a few, complex primitives, or many simple ones ? Should we, rather, define the object using a sampled-field representation ?

2.1.1 Sampled-fields

A practical solution for avoiding the cost of blending many primitives consists in sampling the field function over a 3D grid. Interpolation can then be used for computing field values anywhere in constant-time. This approach was used in real-time sculpting systems, where the additional field contributions created by simple tool-primitives are progressively incorporated to the sampled-field [14, 11, 13, 18]. However, this representation is not well suited to deformations and animation, since objects would have to be resampled through the grid if they were moved or deformed.

2.1.2 Blending simple primitives

Among the practical solutions used in implicit shape design, the most common one consists in blending a large number of very simple primitives such as point skeletons, which sum their field contributions. Controlling these points is quite intuitive [2, 16, 26]. However, the number of primitives needed for modelling a given shape can increase to an arbitrarily high value. Rectilinear or planar parts of objects will only be approximated. As a consequence, point skeletons have been generalized to curve and surface skeletons, which can be done in two ways :

2.1.3 Distance surfaces

The first way of generalizing point skeletons is the following. The field function generated by a skeleton element S at a given point p is defined as the field contribution of the closest point from S to p. In this way, any shape can be used as a skeleton, as long as the shortest distance from a point in space to this shape can be computed. However, non-convex skeletons will generate creases in concavities since the closest point on the skeleton suddenly ”jumps” from one position to another. Splitting complex skeletons into convex parts is not a solution, since bulges may appear near the junctions. A solution to these problems is provided by the use of convolution surfaces.

2.1.4 Convolution surfaces

Directly generating implicit surfaces from continuous, non-convex skeletons requires the use of convolution surfaces [5]. Defined using integrals of field contributions along primitives, convolution surfaces were long considered as very expensive models. The first implementation of convolution surfaces [5] relied on the pre-computation and storage of sampled field values along the skeleton, to be combined with distance information for the query point. This increases efficiency but is an approximate solution, and requires large volumes of memory. An alternative is to find closed-form solutions for integrals. Such solutions were recently given for a number of classical kernels, convolved with a restricted number of simple primitives such as line-segments, arc-curves, or triangles [21]. However, no kernel resulted into a closed-form solution when integrated along a free-form curve or surface.

2.1.5 Complex skeletons and varying radius

In theory, any 3D object can be entirely defined from a geometric skeleton. This skeleton is called the “medial axis”, and is defined as the locus of the centers of the maximal spheres included inside the object. The skeleton can be represented as a graph of interconnected curves and surfaces, along which the radius information is stored. Several methods have been proposed to compute either exact or approximate versions of the skeleton from a discrete representation of the object’s surface [1, 24].

Implicit modelling belongs to the reverse approach, since the aim is to generate surfaces by defining and editing their skeleton. As we just saw, there will be no restriction on the modelled shape if the skeleton is a graph of interconnected

curves and surface patches provided with radius information that describe the object's local thickness. The radius information is taken into account by modifying the local strength of the field function along the skeleton.

The method presented in this paper relies on convolution for generating implicit surfaces along such complex skeletons. However, the skeletons we study here are restricted to graphs of branching curves. The possible extension to surface elements will be discussed in Section 8. Defining varying radii hasn't yet been thoroughly studied for convolution surfaces. This paper improves the approximate solutions that were described in previous works [3, 7], by expressing exact convolution in presence of a varying radius.

2.2 Interactive display and levels of detail

2.2.1 Solutions for interactive display

The lack of parameterisation has long been an obstacle to the interactive display of implicit surfaces. When only local editing of the shape is performed, real-time visualization can be obtained through incremental polygonisation [11]. However, this yields high memory costs since a huge data-structure has to be stored.

In constructive approaches, alternate methods have been used for displaying very quickly approximated representations of the designed shape. The simplest solution is to display non-blended versions of the primitives [19] or offsets of the skeletal elements [20]. Another solution is to display particles attached to the implicit surface [6, 25].Desbrun [10], rather, generates a piece-wise polygonisation, defined by closed-meshes attached to each skeletal element. Each of the meshes samples the region of the implicit volume where the contribution of the associated element is the highest (this region is called the skeleton's *territory*). Unfortunately, this results in local "cracks" between parts of the surface sampled by different meshes. Moreover, the number of polygons linearly increases with the number of primitives, although lots of these polygons may be hidden inside the implicit volume. This makes the method impractical when a large number of skeletons are used.

Our interactive display method is inspired from this last work. However, it reduces the number of hidden surface nodes by using a single mesh for each curve segment or surface patch in the skeleton graph. Moreover, cracks are avoided by making local meshes locally overlap.

2.2.2 The LOD paradigm

Regardless of the representation, very complex objects will need an arbitrarily large number of polygons to be displayed. Parametric models cope with this problem using the level of detail (LOD) paradigm: a region of interest (e.g. high curvature areas) or an object which is close to the camera will be displayed at a finer resolution than others.

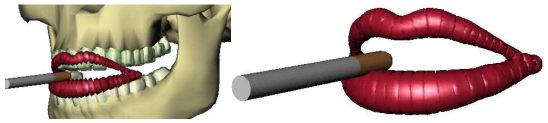

Figure 1. This figure, extracted from [8], illustrates the use of the interactive display method in [10], in the context of implicit lips made of a series of point-primitives positioned along a curve. Even with fine sampling rates (right), small cracks between local meshes are still visible.

To be better suited to interactive modelling, implicit surfaces should obey to this paradigm : at least a coarse version of a shape should be available for immediate interactive display, and refined versions should be available if needed.

Although already studied in the sampled-field representation [23, 15, 13, 18], the LOD paradigm has never been used, to the authors knowledge, in the constructive implicit modelling framework. In the latter case, the only way to provide levels of detail in the field function definition is to progressively simplify (respectively refine) the skeleton that defines the surface, yielding a more efficient field evaluation (respectively, a more detailed shape). Finding implicit primitives that would provide approximations of a single limit shape throughout this simplification/refinement process is not easy. In particular, the classical distance surfaces cannot be used, since bulges would appear each time an element of the skeleton refines into several smaller elements.

The subdivision-based implicit surfaces presented in this paper are a first solution to this problem since they provide approximations of the modelled shapes at different LODs. The subdivision-curve implicit primitive, described next, was first introduced in our previous work [7]. This paper improves this primitive. Our solution relies on closed-form convolution for generating coherent results when the LOD changes.

3 The subdivision-Curve implicit primitive

This section explains how to generate implicit surfaces of varying radii along free-form skeleton curves and provides methods for computing them at different LODs.

3.1 Using subdivision-curves as skeletons

As stated earlier, generating an implicit representation at several LODs can be done by refining or simplifying the skeleton. Our basic idea consists of using a subdivision-curve as a skeleton. Subdivision-curves (see for instance [22]) are limit curves of a series of poly-lines, that are recursively computed using techniques such as "corner cut-

ting". These poly-lines provide the LOD representations of the curve we are looking for.

For instance, let us consider curves defined using Chaikin's subdivision mask : The user gives a control poly-line $(c^0) = (c^0_0, ..., c^0_{m-1})$. This poly-line can be used as a skeleton to generate a first, coarse approximation of the shape. If a finer representation is required, the line is recursively subdivided. This is done by using :

$$c^j_i = r_{-1}\ \dot{c}^j_{i-1} + r_0\ \dot{c}^j_i + r_1\ \dot{c}^j_{i+1}$$

where :

$$\begin{aligned} \dot{c}^j_i &= c^{j-1}_k \ \text{ if } i = 2k \\ \dot{c}^j_i &= \frac{1}{2}(c^{j-1}_k + c^{j-1}_{k+1}) \ \text{ if } i = 2k+1 \end{aligned}$$

If $(r_{-1}, r_0, r_1) = (0, \frac{1}{2}, \frac{1}{2})$, the limit curve is an uniform quadratic Bspline. If $(r_{-1}, r_0, r_1) = (\frac{1}{4}, \frac{1}{2}, \frac{1}{4})$, the limit curve is the cubic uniform B-spline.

In practice, the user may like to slightly modify this model by defining two kinds of joints in the initial poly-lines : those that should be smoothed, and those, called "sticking joints", that should remain unchanged. In consequence we are using a modified version of Chaikin's mask, where a control point is replaced by two points only when it is not marked as a sticking joint. Note that we don't split a point if the curvature of the poly-line at this point is below some threshold ϵ. See figure 2.

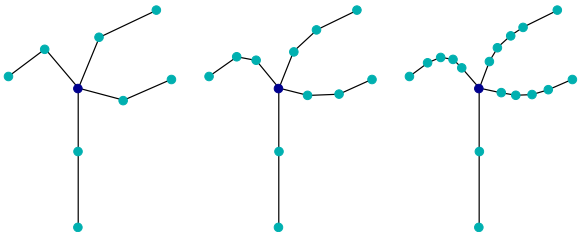

Figure 2. A subdivision-curve skeleton. The user defines a coarse version of the graph of curves (left). During refinement, the curves are smoothed (right), except at the "sticking joints", displayed darker, that the user has specified.

3.2 Convolution with an efficient infinite-support kernel

Maintaining a coherent shape for the implicit surface when a subdivision-skeleton refines requires the use of *convolution* for generating the surface. Indeed, thanks to the properties of the integral, convolution surfaces maintain a constant shape when a skeleton primitive breaks into pieces. This allows for an almost continuous variation of the shape during refinements and simplifications.

Subdivision-curve implicit primitives are computed by convolving a poly-line, which is the representation of the subdivision curve at a given LOD, with a well chosen kernel. In order to maintain a reasonable computational time, one may use kernels that provide closed-form formulae when convolved along line segments. The result of convolution with the poly-line is then the sum of contributions obtained from the different line-segments.

In addition to the kernels presented in [21], one may use the following convolution kernel, which provides a particularly simple and inexpensive closed-form solution :

$$f_S(P) = \frac{1}{d(P,S)^3}$$

where s is the skeletal point, P is the query point, and d denotes the distance. Once integrated along a segment S, this kernel gives[1] :

$$F_S(P) = \frac{\sin(\alpha_0) + \sin(\alpha_1)}{d(P,H)^2} \qquad (1)$$

where $d(P,H)$ is the distance between P and its projection point H on the line-segment support, and where the angles α_0 and α_1 are the signed angles $([PH],[PV_0])$ and $([PH],[PV_1])$, respectively (see Figure 3). In practive, the sine in the expression of F_S does not need to be computed since:

$$\sin(\alpha_0) = \frac{(H-P)^T.(P-V_0)}{d(P,V_0)^2}$$

Several of the examples shown in this paper have been generated using this kernel.

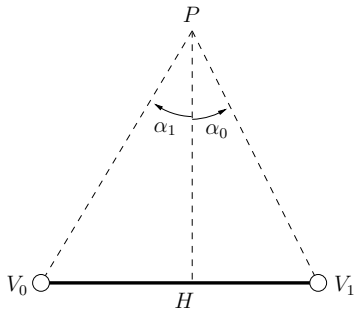

Figure 3. The field function that results from the convolution of $1/d^3$ along a line-segment skeleton is based on the distance between P and its projection point H, and on the signed angles α_0 and α_1.

[1] In this paper, we use the letter f for denoting a field value before integration, and F for denoting the convolved field obtained by integrating the previous one along a given skeleton.

3.3 Comparison of kernels

Computing the field value in equation (1) does not take too many operations, since sine values are easily computed using scalar products. In comparison with the number of operations given for the finite support polynomial kernel in [21], we get :

Number of operations	*	/	+ or -	sqrt
New kernel	20	3	23	2
Polynomial kernel	33	3	36	1

Numerical experiments showed that the new kernel has about the same performance as the polynomial one.

In practice, the convolution surfaces we use are generated by taking the iso-surface of iso-value $c = 1$ of the convolved field.

4 Convolution surfaces of varying radius

4.1 Specifying a varying radius along subdivision-curve

Associating a non-constant radius parameter along a subdivision-curve is easy : the user defines radius values r_i at the joints of the coarse control polygon that is used for defining the subdivision-curve. These values are interpolated for defining a radius $r(u)$ at any parameter u along the curve. When the skeleton curve subdivides, new radii are associated to the new vertices using the same subdivision mask as for vertex positions. In the last section, each subdivision curve had a constant radius all along. We now present two solutions to compute a convolution surface with varying radius. The solution presented in the next subsection was first presented in [7]. The one presented in subsection 4.3 is a new contribution.

4.2 An approximate solution

Classical methods for generating surfaces of varying radius modify the distance $d(P)$ used in the field function computation, for instance by dividing it by $r(u)$, where u is the parameter of the curve point that is the closest to P (see [3, 12]).

The closed-form field value at point P given in Equation (1) relies on the distance d to the closest point H on the segment skeleton. This allows us to generate a surface of a non-constant radius along the skeleton. As stated earlier, the user defines radii at each control point of the skeleton curve, so a radius $R(H)$ at point H can be interpolated from the radii at the segment extremities.

The surface is set to a varying radius by replacing $d(P,H)^2$ in Equation (1) by $D(P,H)^2$, where :

$$D(P,H) = \sqrt{2}\frac{d(P,H)}{R(H)}$$

With this expression, the surface will tend to have the radius $R(H)$ if P is near the center of a very long primitive, since sine values in Equation (1) then tend to 1. As in conventional convolution models, the local surface thickness is smaller than the specified value near the extremities of a primitive. Moreover, if the user has specified different radius values along a subdivision-curve primitive, the surface thickness will change accordingly along the primitive, since $R(H)$ changes with the projection point H. Note that to ensure C^1 continuity, we have to keep interpolating $R(H)$ when $R(H)$ is growing *and* H pass the end of the current line segment skeletal element. When decreasing, $R(H)$ is set to zero wherever interpolating it would results in a negative radius.

This solution has the advantage to be fast but does not provide an exact convolution surface : the radius first derivative is not continuous; see Figure 11. In the case of convolution surfaces, the new distance value should be integrated into the kernel *before convolution*, in order to preserve the good properties of convolution surfaces, such as their invariance under skeleton subdivision. Although it is not the case for the approximate solution proposed above, this solution is nevertheless a good approximation, and is useful because of its short computation time.

4.3 Exact convolution with varying radius

We now present a new kernel that allow exact convolution with a varying radius.

We managed to derive closed-form formulas in the varying radius case for two simple convolution kernels, namely $f_S(P) = r^2/d(P,S)^2$ and $f_S(P) = r^3/d(P,S)^3$. The varying radius r is defined along the segment using linear interpolation from the values r_0 and r_1 defined at the extremities. The best (i.e. the cheaper) of the two solutions is obtained using

$$f_S(P) = \frac{r^2}{d(P,S)^2}$$

Let the origin of parameterisation H along the skeleton be the projection of point P on it. Let V_0 and V_1, of respective coordinates a_0 and a_1, respectively be the farthest and the closest segment extremity from P (see Figure 4). If M is a point on the segment $[V_0V_1]$, we denote by u or $u(M)$ the signed distance between H and M, with $u(V_1) > 0$. Then the integral to compute is:

$$F_S(P) = \int_{-a_0}^{a_1} \frac{\frac{r_1-r_0}{a_1-a_0}u + \frac{r_0a_1-r_1*a_0}{a_1-a_0}}{d^2+u^2}du$$

which gives the closed-form solution:

$$F(S,P) = \frac{((d * C - D^2/d) * A + (r_0 - r_1) * D * B + (a_0 - a_1) * C)}{(a_0 - a_1)^2}$$

where:

$$\begin{aligned} A &= \arctan(\tfrac{a_1}{d}) + \arctan(-\tfrac{a_0}{d}) \\ B &= \log((a_1^2 + d^2)/(a_0^2 + d^2)) \\ C &= (r_0 - r_1)^2 \\ D &= r_0 * a_1 - r_1 * a_0 \end{aligned}$$

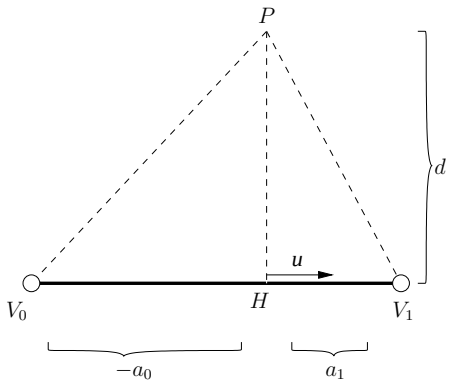

Figure 4. Field function computation when a varying radius is set.

An important feature of this field function is that it *preserves surface smoothness* when different radii are specified along a poly-line. But this kernel suffers from its more expensive computation time. Although a linear variation of the radius has been considered along each line segment, the convolution process, which consists into an integration, produces a surface that is at least C^1 everywhere. A result is shown in Figure 5. Experience shows that our first kernel can be used when we don't need high precision on the sampling surface, e.g. for the designing process, as the final shapes obtained with both kernel are comparable.

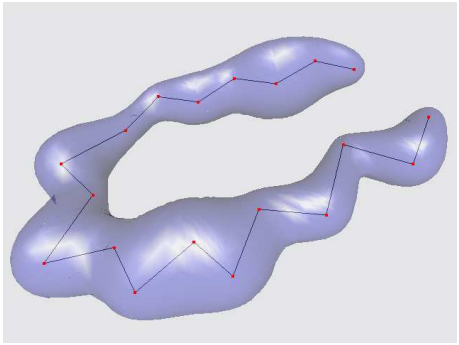

Figure 5. Example of surface with varying radius.

We do not compare the performance of this kernel with the previous one since it does not use the same class of numerical operations ($\arctan$, $\log$). In practice, this kernel is slower but still usable at interactive rates.

5 Interactive display

The method we use for the interactive display of subdivision-curve primitives is an extension of the adaptive sampling method introduced by Desbrun [10] : Sample-points are sent along given axes by each skeletal element in order to sample the element's "territory", i.e. the region where its field contribution is larger than the contribution of any other skeletal element. Sample-points may start from a previous position, then converge to the iso-surface, enabling the use of benefits gained from temporal coherence. This feature is essential for increasing efficiency during interactive modeling or animation sessions. This idea of temporal coherence is also very interesting in our subdivision/refinement framework : it will allow sample points to migrate from their previous position when a skeleton refines. However, two problems have to be solved for generating an efficient yet good quality display :

- Associating individual meshes with each skeleton primitive is inefficient for neighboring primitives (see Figure 1).
- Gaps between local meshes appear in Desbrun's method, since sample points are stopped at the limit of their associated primitive's territory.

Our solutions to these two problems are the following : first of all, we generate a single closed mesh around each subdivision-curve primitive. Mesh vertices are attached to specific points on the initial coarse poly-line, and converge to the iso-surface along fixed axes, as was done in [10]. When the subdivision-curve refines, we do not change the attachment of the vertices so that we can dissociate the refinement of the skeleton curves and of the implicit surface polygonisation. This increases the quality of results, as shown in Figure 6, and reduces the number of generated polygons compared to the use of several disconnected meshes as in Figure 1.

Secondly, we avoid discontinuities when several subdivision-curve primitives blend together by extending the region sampled by each mesh: instead of only sampling its primitive's territory, a mesh is set to sample the region of the implicit surface where the parent primitive's field value *plus a given constant* (0.5 in our implementation) is higher that any other contribution. This creates local overlapping regions between neighboring meshes, as shown in Figure 7, thus increasing the visual quality of the interactive display.

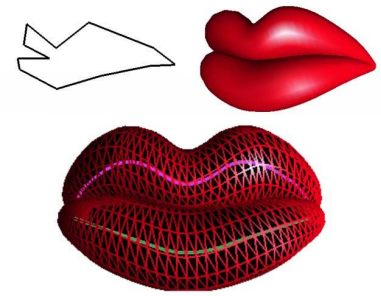

Figure 6. Interactive display of a lips model using the subdivision-curve methodology. The skeleton is made of two user-defined control-polylines whose associated implicit surfaces do not blend (left). Our display method associates a closed mesh to each of the curves (center and right). A non-constant surface radius is specified along the lines. The subdivided subdivision-curves are displayed on the right.

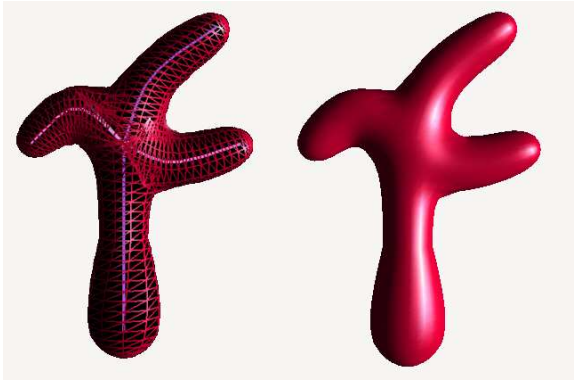

Figure 7. Implicit surfaces generated from the skeleton in Figure 2, made of two curves that sum their contributions. The local overlapping between the two closed meshes used for display yields quite a good quality visual result, without altering performances.

6 Avoiding unwanted blending with preserved smoothness

Modeling and animating complex shapes requires the ability to use a restricted blending graph. For instance, the loops of the snake's body in Figure 8 should not blend together.

Current solutions to the unwanted blending problem [17, 9], do not maintain the C^1 continuity of the shape everywhere, since the field at a given point is defined as the maximal contribution from groups of skeletons that blend together. This solution results in a union of volumes, possibly creating tangent discontinuities in regions where blending properties change.

Modeling with subdivision-curve primitives offers a practical solution to the problem: we use the skeleton, i.e. our graph of interconnected subdivision curves for defining blending properties, stating that a line-segment blends with its immediate neighbors, and that blending is locally transitive. Now, we have to avoid unwanted blending between segments that are very far away with respect to the topology of the graph, as the folds of the snake in Figure 8. To do so, we integrate the field at a point P by recursively adding the contribution of the segments that are nearest to P, and the contributions from their neighbors, but stopping along a curve as soon as a contribution can be neglected. Then, if a long curve loops, then folds back, the two folds will not blend together. This method yields C^1 continuity since a contribution is disregarded only when it is negligible. However, it should be noted that it only gives a partial solution to the problem : generating a surface that immediatly folds back onto itself and should not blend do not blend cannot be done using our method; see Figure 9.

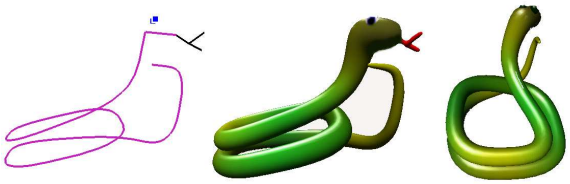

Figure 8. Unwanted blending is avoided while preserving C^1 continuity when this snake folds back onto itself.

7 Results

The methods described in this paper have been implemented in an interactive modelling system, where the user creates a new model by interting new vertex to the skeleton graph, and connecting them using segments. He/she may also select either a single vertex or groups of vertices in order to move them, to edit their local surface radius parameter, or to delete them. The implicit surfaces is updated after each operation, using the coarsest level of detail on the skeleton as long as the user does not push the refinment button. Three display modes are available : displaying both the skeleton and the implicit surface (using transparency), rendering the implicit surface only, and displaying it in wireframe.

Figure 10 depicts some steps of a modelling session, showing two of the different display modes.

The stylized kangaroo example in Figure 11 illustrates the kind of shape we can generate with our approach. The object is displayed at two different levels of detail. The left hand-side model only relies on 27 line-segment primitives for approximating the subdivision-curves that constitute the skeleton, while the right hand-side model uses 216 line-

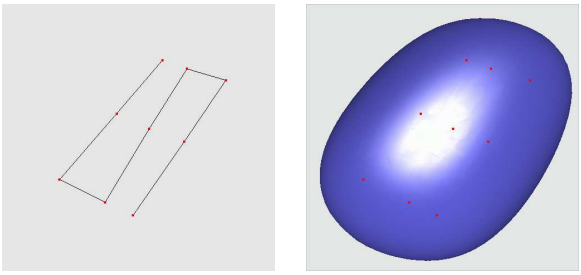

Figure 9. Unwanted blending misbehavior.

segments. Knowing that computing the field contribution of a line-segment takes a constant time, computing the surface on the left is about 8 times faster than computing the one on the right, if the same number of points are used to sample them.

In practice, there is usually no need to compute a fine mesh along a coarse version of a subdivision-curve as only a coarse representation of the object will be displayed. Figure 12 shows different representations of the same shape obtained by adapting the mesh resolution according to the resolution of the underlying skeleton. Figure 13 shows what kind of models can be designed using our modelling software. The coarse skeleton is made of 77 line-segments. Problems appear between the fingers of the toon because (in the computation of this picture only), vertices were not constrained to sample the territory of the line-segment they were attached to.

8 Conclusion

Subdivision-curve primitives define free-form implicit surfaces that can be displayed in real-time by adjusting the LOD at which the implicit surface is computed. This is done in a framework of structured modeling, where objects are edited by applying deformations to the skeleton curve. These two features make the model very well suited to interactive modeling systems where the user could first display a coarse version of a shape, and then adjust details by "zooming" (i.e. increasing the LOD) in the region where the skeleton curve is to be edited.

Subdivision-curve primitives would also be a good model for generating animation sequences, since the skeleton curves give an intuitive way to apply motion and deformations to objects. If these primitives were used in an interactive animation system, an automatic, error-driven, refinement/simplification procedure would have to be settled for automatically tuning LODs, using criteria such as the size of a primitive on the screen.

Interesting future work would consist in extending our

Figure 10. Modeling with skeleton curves.

methodology to surface primitives, thus offering the possibility to model any 3D shape. A closed-form convolution primitive already exists for triangles [21], and could be used for generating subdivision-surface primitives based on triangle subdivision schemes. Closed-form primitives could also be searched for quadrilateral primitives, since many subdivision schemes such as Catmull-Clark scheme [22] rely on quadrilateral elements. Our solution to the unwanted blending is far from perfect since it requires large loops to avoid blending. Other solutions should be proposed to suppress this constraint.

Acknowledgments: The authors would like to thank Andrei Sherstyuk for his helpful suggestions, and Pauline Jepp for carefully re-reading the early version of this paper.

References

[1] D. Attali and A. Montanvert. Computing and simplifying 2d and 3d semicontinuous sekeletons of 2d and 3d shapes. *Computer Vision and Image Understanding*, 67(3):261–273, Sept. 1997.

Figure 11. A stylized kangaroo modeled using subdivision curve primitives. Two different levels of detail are shown.

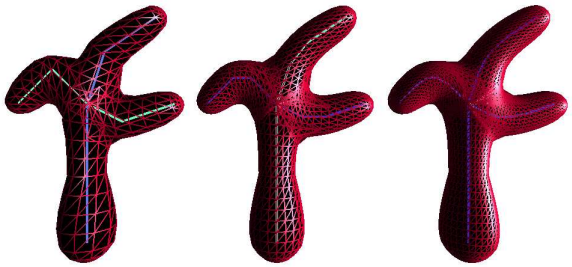

Figure 12. Representations of a surface at different LODs can be obtained by tuning both the resolution of the local meshes that sample the surface, and the resolution of its subdivision-curve skeletons. The left-hand-side model could be used when the object is far away, and the two other representations when it gets closer.

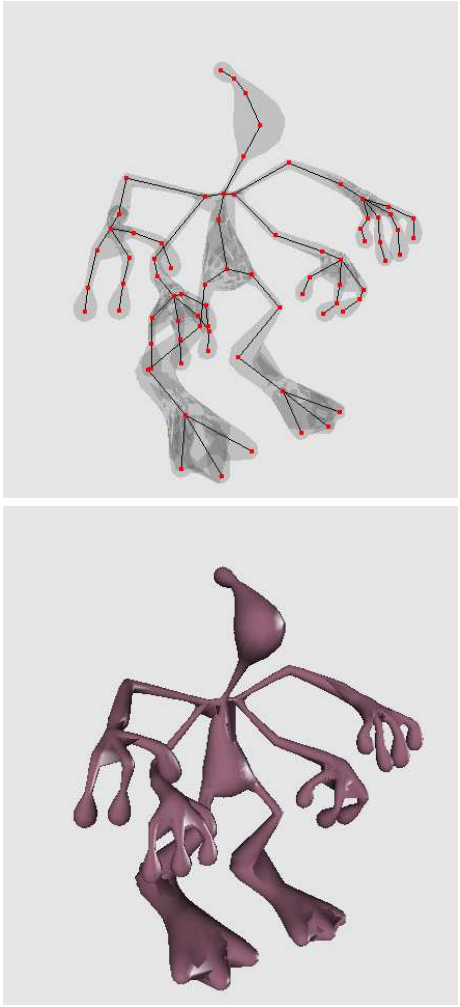

Figure 13. A toon created with our interactive modelling system.

[2] J. Blinn. A generalization of algebraic surface drawing. *ACM Transactions on Graphics*, 1(3):235–256, July 1982.

[3] J. Bloomenthal. Skeletal design of natural forms. *PHD Thesis*, The University of Calgary, Jan. 1995.

[4] J. Bloomenthal, C. Bajaj, J. Blinn, M.-P. Cani-Gascuel, A. Rockwood, B. Wyvill, and G. Wyvill. *Introduction to Implicit Surfaces*. Morgan Kaufmann, July 1997.

[5] J. Bloomenthal and K. Shoemake. Convolution surfaces. *Computer Graphics*, 25(4):251–256, July 1991. Proceedings of SIGGRAPH'91 (Las Vegas, Nevada, July 1991).

[6] J. Bloomenthal and B. Wyvill. Interactive techniques for implicit modeling. *Computer Graphics*, 24(2):109–116, Mar. 1990. Proceedings of Symposium on Interactive 3D Graphics.

[7] M.-P. Cani and S. Hornus. Subdivision curve primitives: a new solution for interactive implicit modeling. In *Shape Modelling International*, Genova, Italy, may 2001. IEEE Computer Society Press.

[8] M.-P. Cani-Gascuel. Layered deformable models with implicit surfaces. In *Graphics Interface (GI'98) Proceedings*, Vancouver, Canada, June 1998. A K Peters. Invited paper.

[9] M.-P. Cani-Gascuel and M. Desbrun. Animation of deformable models using implicit surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 3(1):39–50, Mar. 1997.

[10] M. Desbrun, N. Tsingos, and M.-P. Cani-Gascuel. Adaptive sampling of implicit surfaces for interactive modeling and animation. *Computer Graphics Forum*, 15(5):319–327, Dec. 1996. A preliminary version of this paper appeared in *Implicit Surfaces'95*, Grenoble, France, may 1995.

[11] E. Ferley, M.-P. Cani, and J.-D. Gascuel. Practical volumetric sculpting. *To appear in the Visual Computer, 2000*, 2000. A preliminary version of this paper appeared in *Implicit Surfaces'99*, Bordeaux, France, sept 1999.

[12] E. Ferley, M.-P. Cani-Gascuel, and D. Attali. Skeletal reconstruction of branching shapes. *Computer Graphics Forum*, 16(5), Dec. 1997. An early version of this paper appeared in *Implicit Surfaces'96*, Eindhoven, The Netherlands, oct 1996.

[13] S. Frisken, R. Perry, A. Rockwood, and T. Jones. Adaptive sampled distance fi elds: A general representation of shape for computer graphics. *Computer Graphics*, July 2000. Proceedings of SIGGRAPH'00 (New Orleans, July 2000).

[14] T. Galyean and J. Hughes. Sculpting: An interactive volumetric modeling technique. *Computer Graphics*, 25(4):267–274, July 1991. Proceedings of SIGGRAPH'91 (Las Vegas, Nevada, July 1991).

[15] L. Grisoni and C. Schlick. Multiresolution representation of implicit objects. In *Implicit Surfaces'98—Eurographics and ACM-Siggraph Workshop*, pages 1–10, Seattle, USA, June 1998.

[16] H. Nishimura, M. Hirai, T. Kawai, T. Kawata, I. Shirakawa, and K. Omura. Objects modeling by distribution function and a method of image generation (in japanese). *The Transactions of the Institute of Electronics and Communication Engineers of Japan*, J68-D(4):718–725, 1985.

[17] A. Opalach and S. Maddock. Implicit surfaces: Appearance, blending and consistency. In *Fourth Eurographics Workshop on Animation and Simulation*, pages 233–245, Barcelona, Spain, Sept. 1993. Springer.

[18] R. N. Perry and S. F. Frisken. Kizamu: A system for sculpting digital characters. *Proceedings of SIGGRAPH 2001*, pages 47–56, August 2001. ISBN 1-58113-292-1.

[19] J. Shen and D. Thalmann. Interactive shape design using metaballs and splines. In *Implicit Surfaces'95—the First Eurographics Workshop on Implicit Surfaces*, pages 187–195, Grenoble, France, Apr. 1995.

[20] A. Sherstyuk. Interactive shape design with convolution surfaces. In *Shape Modeling International '99*, pages 56–65, Aizu-Wakamatsu, Japan, Mar. 1999. IEEE Computer Society Press.

[21] A. Sherstyuk. Kernel functions in convolution surfaces: a comparative analysis. *The Visual Computer*, 15(4), 1999.

[22] E. Stollnitz, T. Derose, and D. Salesin. *Wavelets for Computer Graphics*. Morgan Kaufmann, San Francisco, California.

[23] L. Velho and J. Gomez. Approximate conversion of parametric to implicit surfaces. *Computer Graphics Forum*, 15(5):327–337, Dec. 1996. A preliminary version of this paper appeared in *Implicit Surfaces'95*, Grenoble, France, may 1995.

[24] A. Verroust and F. Lazarus. Extracting skeletal curves from 3d scattered data. In *Shape Modeling International '99*, pages 194–201, Aizu-Wakamatsu, Japan, Mar. 1999. IEEE Computer Society Press.

[25] A. Witkin and P. Heckbert. Using particles to sample and control implicit surfaces. *Computer Graphics*, pages 269–278, July 1994. Proceedings of SIGGRAPH'94.

[26] G. Wyvill, C. McPheeters, and B. Wyvill. Data structure for soft objects. *The Visual Computer*, 2(4):227–234, Aug. 1986.