SCALe-invariant Integral Surfaces

Cédric Zanni, Adrien Bernhardt, Maxime Quiblier, Marie-Paule Cani

Volume xx (200y), Number z, pp. 1–13

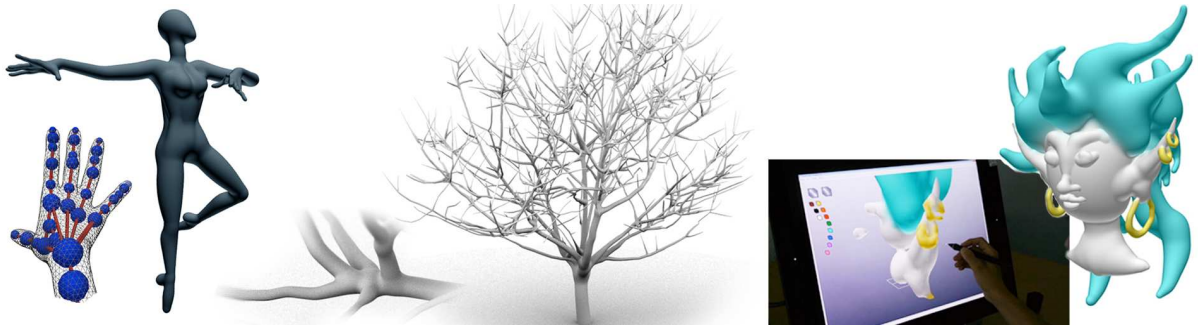
SCALe-invariant Integral Surfaces

author preprint

The definitive version is available at www.interscience.wiley.com

C. Zanni[1,2], A. Bernhardt[1,2], M. Quiblier[2], M.-P. Cani[1,2]

[1]Laboratoire Jean-Kutzmann, University of Grenoble, France
[2]INRIA Grenoble - Rhône-Alpes, France



(a) (b) (c)

Figure 1: *SCAle-invariant Integral Surfaces (SCALIS) makes interactive, skeleton-based modeling intuitive (a), provides a direct reconstruction of complex shapes from their medial axes (b) and is very convenient for sketch-based modeling (c).*

Abstract

Extraction of skeletons from solid shapes has attracted quite a lot of attention, but less attention was paid so far to the reverse operation: generating smooth surfaces from skeletons and local radius information. Convolution surfaces, i.e. implicit surfaces generated by integrating a smoothing kernel along a skeleton, were developed to do so. However, they failed to reconstruct prescribed radii and were unable to model large shapes with fine details. This work introduces SCALe-invariant Integral Surfaces *(SCALIS), a new paradigm for implicit modeling from skeleton graphs. Similarly to convolution surfaces, our new surfaces still smoothly blend when field contributions from new skeleton parts are added. However, in contrast with convolution surfaces, blending properties are scale invariant. This brings three major benefits: the radius of the surface around a skeleton can be explicitly controlled, shapes generated in blending regions are self-similar regardless of the scale of the model, and thin shape components are not excessively smoothed out when blended into larger ones.*

1. Introduction

Geometric skeletons, namely lower dimensional structures centered inside a shape, have been instrumental in shape analysis, shape matching or as a tool for shape deformation for many years [Lon98, BL99]. A typical example is the medial axis of a solid shape [SPB96], a non-manifold graph of curve and surface components, defined as the locus of the centers of maximal spheres included into the shape. Other examples are Reeb graphs used for extracting topological information, or centered graphs of curves used for automatically fitting animation skeletons [BGSF08, AHLD07]. With RigMesh, [BJD*12] introduced a sketch-based mod-

eling method that unifies modeling and rigging stages of the character animation pipeline, thus enabling the easy reposing and animation of the created shape.

Easing the reverse operation, i.e. the generation of smooth 3D shapes from skeletons, would bring many benefits. Firstly, it would enable to store solid models in a skeletal form, a highly compact representation compared to B-reps due to their lower topological dimension. Secondly, using skeletons as modeling primitives would ease conceptual shape design as well as subsequent deformation and animation due to the high-level vision they provide on geometry and topology. Lastly, being able to generate shapes from skeletons would ease the reconstruction of smooth solids from raw data such as point-sets or voxels, for which skeleton extraction techniques are already available.

Theoretically, skeletal structures are reversible shape representations when associated with radius information. However, generating smooth solid models from skeletal information is intricate. Convolution surfaces, namely iso-surfaces of scalar fields generated by convolving a skeleton with a smoothing kernel, are probably the most attractive model to do so. Individual primitives generated by each skeleton component seamlessly blend when their field contributions are summed, which makes modeling by parts straightforward. This model was successfully used in both animation and sketch-based modeling applications [Blo02, BPCB08]. Unfortunately, convolution surfaces also bring severe drawbacks which restrict their use: firstly, although the smoothing kernel can be weighted along the skeleton, there is no mechanism for prescribing the desired radii; secondly, modeling large shapes with fine details is difficult, as small sized components are typically smoothed out when blended into larger ones; even more surprisingly, convolution surfaces are not scale invariant. Depending of the scale of the workspace, blending produces quite different outputs for the same input shapes. These drawbacks make shape control very difficult and are one of the reasons why skeletons never spread as a modeling primitive.

We therefore introduce *SCAle-invariant Integral Surfaces* (SCALIS), a new implicit model aimed at making skeleton-based modeling usable. Based on a combination of convolution and space warps, SCALIS retains the advantages of convolution surfaces, namely making the resulting shape independent of the degree of subdivision of the skeleton and producing seamless blends (using a simple sum of fields) when new skeletal branches are added. However, in contrast with convolution surfaces, SCALIS avoids the three problems listed before: the radii of the generated shapes can be explicitly tuned; the level of smoothing (still controlled through the choice of a smoothing kernel) is self-similar at different scales and brings scale-invariant blending; small shape components are not smoothed out anymore when blended with larger shapes, which makes detailed modeling possible.

2. Background

2.1. Distance surfaces

Implicit surfaces, i.e. iso-surfaces $f(\mathbf{p}) = c$ of some smooth scalar field f, provide an easy way of creating smooth shapes of arbitrary topological genus since they blend when their fields are summed.

Among them, skeleton-based implicit surfaces use scalar fields defined as decreasing functions $k : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ of the distance from some geometric skeleton, typically points, line-segments or triangles. Distance surfaces, such as "Blobby Molecules" [Bli82] or "Soft-Objects" [WMW86], are defined by:

$$f_S(P) = k\left(\min_{\mathbf{g}\in S} \|\mathbf{g}-\mathbf{p}\|\right) \tag{1}$$

where S is a geometric skeleton. One of the advantages of these surfaces is the ease to prescribed radii around skeletons. Nevertheless they present a major drawback: the bulging effect. It occur whenever two skeletons are set in continuity and blended with a simple sum (see figure 2(a)). This is due to the fact that:

$$k\left(\min_{\mathbf{g}\in S_1\cup S_2} \|\mathbf{g}-\mathbf{p}\|\right) \neq k\left(\min_{\mathbf{g}\in S_1} \|\mathbf{g}-\mathbf{p}\|\right) + k\left(\min_{\mathbf{g}\in S_2} \|\mathbf{g}-\mathbf{p}\|\right)$$

2.2. Convolution surfaces

Convolution surfaces [BS91] were developed to avoid unwanted bulges when implicit primitives generated by two neighboring skeletons blend. They define f as the integral of field contributions from all the individual points $\Gamma(s)$ along the skeleton:

$$f(\mathbf{p}) = \int_{s\in S} k\left(\|\Gamma(s)-\mathbf{p}\|\right)\, ds \tag{2}$$

where k is a smoothing kernel, and S a geometric skeleton of parameterization Γ. The integral being linear, the generated shape is independent of the subdivision of the skeleton into components, and is therefore bulge-free. Independence from subdivision enables to use complex networks of curves and

(a) (b)

Figure 2: *Blending by summation of two primitives: distance primitives (a) create a bulge while convolution (b) seamlessly blend.*

surfaces as skeleton, the latter being split into adequate sets of line-segment and triangle skeletons for evaluation.

While the first convolution surfaces used *Gaussian* kernels, subsequent work provided kernels with closed-form expressions [MS98, She99] for both convolution along segment and triangle skeletons. These kernels can be expressed as:

- *Cauchy* of order i :

$$k(d) = \frac{1}{\left(1+\left(\frac{d}{\sigma}\right)^2\right)^{\frac{i}{2}}} \tag{3}$$

- *Inverse* of order i :

$$k(d) = \frac{1}{\left(\frac{d}{\sigma}\right)^i} \tag{4}$$

- *Compact Polynomial* of order i :

$$k(d) = \begin{cases} \left(1-\left(\frac{d}{\sigma}\right)^2\right)^{\frac{i}{2}} & if\ d < \sigma\,, \\ 0\ otherwise. \end{cases} \tag{5}$$

where σ is a resizing constant. Each kernel has its assets: *Cauchy* and *Inverse* are C^∞ but with a global support. *Inverse* also guarantees that the skeleton is totally included inside the generated surface (because of infinite field values over the skeleton). This comes with a drawback: this kernel is less convenient when digging operators such as Barthe's difference [BWdG04] are used, since these operations will have little effect near the skeleton. *Compact Polynomial* is "only" $C^{\lfloor \frac{i-1}{2} \rfloor}$, but it has the advantage of having a local support. This eases local shape control and enables efficient field queries.

2.3. Extension to non-constant radius

Weighted convolution using a weighting function τ along the skeleton, was introduced to capture convolution primitives of non-constant radius. Two different models were used:

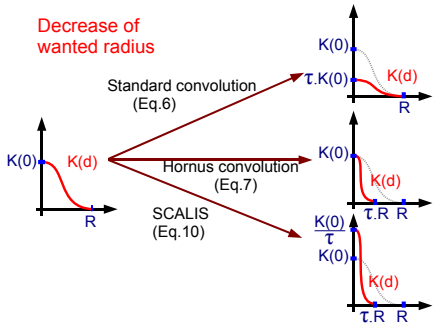
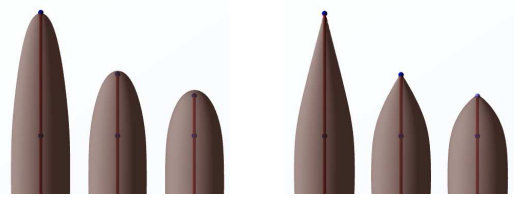


Figure 3: *Weighted models: kernel variations when the user decreases τ to get a thinner shape along a segment-skeleton.*



(a) (b)

Figure 4: *Shapes generated when the weight linearly falls to 0: (a) standard formulation, (b) SCALIS. Note the unwanted rounded extremities in (a).*

- The *standard formulation* developed by Jin [JTFP01, JT02b, JT02a] defines the field function as:

$$f(\mathbf{p}) = \int_{s\in S} \tau(s)\, k\left(\|\Gamma(s)-\mathbf{p}\|\right)\, ds \tag{6}$$

  This formulation was later re-used [AC02, ITZkF04, BPCB08], and general closed forms solutions were developed [HC12] for segment skeletons. Convolution on triangles was recently studied for *Compact Polynomial* kernels in [JTZ09, ZJLZ12, Hub12]. In this case sharp details can be obtained when the convolution radius is set to a small value.
- In contrast, Hornus [HAC03] used:

$$f(\mathbf{p}) = \int_{s\in S} k\left(\frac{\|\Gamma(s)-\mathbf{p}\|}{\tau(s)}\right)\, ds, \tag{7}$$

  A closed-form solution on segment-skeleton was only provided for the *Inverse* kernel of degree 2.

Figure 3 depicts the different strategies used by these two models to generate radius variations: while equation (6) changes the height of the kernel while keeping the width of the support unchanged, equation (7) generates a kernel of constant height, but of varying support-size. For comparison, our new solution SCALIS, scales both the width of the support and its height while keeping the area below the curve constant. This brings scale invariance.

Although they extended the range of shapes that could be modeled through convolution, the weighted convolution and Hornus models raised a number of issues:

1. **Lack of radius control:** There is no mechanism for prescribing a desired radius. This comes from the non-linear dependency between the weighting function τ and the radius of the iso-surface of interest (see figure 6(c)). In addition to preventing precise shape control, this makes modeling non-intuitive (see figure 4(a) and 12(a,b)).
2. **Blurring and vanishing details:** Small scale details are smoothed out when blended to a shape of larger size, especially using eq. (6) (see figure 5(a)). In addition, thin surfaces tend to collapse (figure 6(a)). As noted in [JT02a], the *Inverse* kernel is not affected by this problem due to the infinite field value on the skeleton.

3. **Scale-dependent blending:** The way two convolution surfaces blend when the associated fields are added does not depend only on their shapes, but also on the scale at which they were defined: as τ decreases, blending becomes smoother and smoother (see figure 7).

Recent work in implicit modeling has addressed some of these issues. Two sketch-based modeling systems based on convolution had to cope with radius control [ITZkF04, BPCB08]: either they used an optimization process to compute τ at skeleton vertices from 2D skeletons and radii extracted from the sketch, or relied on a sketching space of pre-defined size, combined with pre-set correspondence between weight and radius.

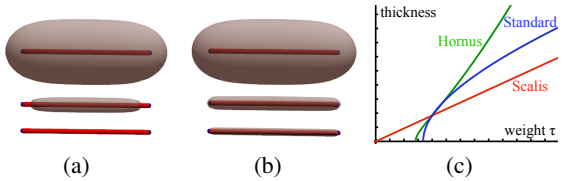


Figure 6: *Primitives of decreasing radius (Cauchy kernel): The standard formulation (a) causes the surface to collapse at the center of the skeleton and then vanish, contrary to SCALIS (b). (c) plots the thickness of the generated shape in function of τ for an infinite segment skeleton.*

2.4. Blobtree and blending

An alternative to convolution surfaces is a combination of implicit primitives within a Blobtree structure [WGG97], which enables one to create complex objects from simpler ones through a hierarchies of blending or warping nodes. As stated above, the most classical (smooth) blending operation is the sum:

$$f(\mathbf{P}) = \sum_i f_i(\mathbf{P}).$$

The Ricci operator [Ric73]:

$$f(\mathbf{P}) = \sqrt[n]{\sum_i f_i(\mathbf{P})^n},$$

provides some freedom in the choice of the blending behavior. Indeed it describes a family of blending that spans from the blending by sum ($n = 1$) to the union by maxima ($n = +\infty$).

Several methods have been introduced in this context to further improve blending. Among them, bounded blending [PPIK02, PPK05] enables to localize blending, either by using two control points on the surfaces or by setting an arbitrary bounding volume defined by an implicit surface. This method enables a variety of blending styles and users can tune it in order to control the smoothing of small details. Another method, presented in [dGWvdW09], uses a local restriction of the blending range to control blending and can be used to improve blending of small details into larger ones. By introducing a complex binary operator based on extra influence primitives generated at the intersection of the input surfaces, [BBCW10] was able to overcome the blurring details problem automatically. However, this operator is costly compared to the sum and it does not handle n-ary combinations. Furthermore, just like the previous ones, it would not insure shape independence from skeleton subdivision. We therefore develop a new model, as discussed below, which tries to overcome these limitations.

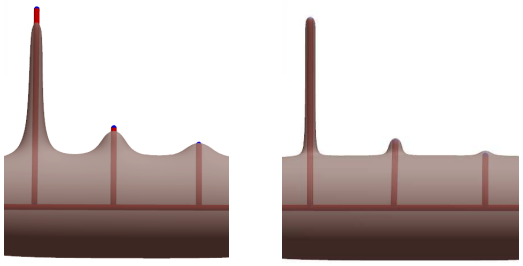


Figure 5: *Blending of thin primitives into a thick one. The details blur when the standard formulation is used (a), but not with SCALIS (b).*

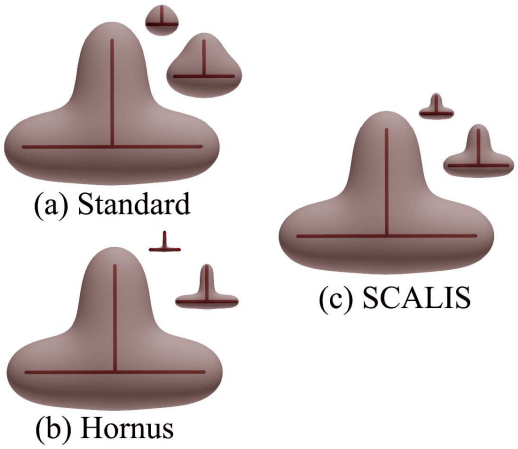


Figure 7: *Scaling applied to both convolution skeletons and τ: (a) Standard formulation, (b) Hornus formulation, (c) SCALIS. The first scaling factor is $\frac{1}{3}$ and the second is $\frac{1}{6}$, for a factor of $\frac{1}{10}$ the surface for Hornus formulation would only exist on a small vicinity of the intersection of the two skeletons.*

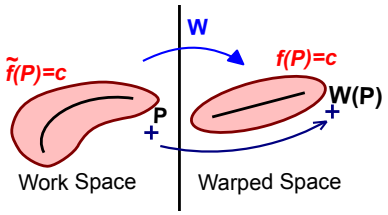


Figure 8: *Principle of warp-based implicit primitives.*

3. SCALIS: Scale-invariant Integral Surfaces

Since standard weighted convolutions do not generate a field that is a linear function of τ, they are not scale-invariant (see figure 7 (a)): both the relative radii of individual primitives and the way they blend change when models are scaled. In practice, this prevents the use of convolution primitives in general modeling systems, since shape resizing operations cannot be applied.

In this work, we revise convolution surfaces in order to solve the three major problems we have identified while maintaining their most attractive features, namely n-ary blending using a simple sum and shape independence from skeleton subdivision. Our insight is that making convolution scale invariant is sufficient to solve the three issues listed in section 2.4. This section develops our solution in the case of primitives of constant weight τ. The extension of SCALIS to primitives of varying radii is presented in Section 4.

3.1. Warped-based implicit primitives

In the implicit modeling framework, space warps such as Barr's bending, twisting and tapering [Bar84] have been used for a long time for defining complex primitives from simpler ones [WO97]. The idea is to define the scalar field $\tilde{f}$ of the new primitive from a warping $w : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ and a simple primitive of field f, in the following way:

$$\tilde{f}(\mathbf{p}) = f(w(\mathbf{p}))$$

In other words, query points are transported through w to another space, where the new primitive warps into the simple one (see figure 8). We rely on this warping principle to define scale-invariant integral surfaces.

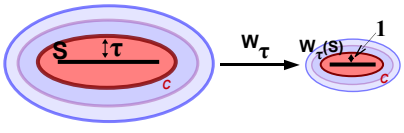


Figure 9: *Space warp used to get scale-invariant implicit primitives.*

3.2. Combining convolution with warping

To get two identical implicit primitives at different scales, not only in terms of shape but also blending behavior, their scalar fields should be scaled versions of each other. To get this invariance property, the idea is to compute field values in a warped space, where all skeletons have a unit weight, i.e. where the field is computed using the initial convolution formulation of equation (2). This is done by using a warped primitive, where the warping w is a simple scaling, applied to both the query point and the primitive parameters (skeleton S and weight τ). See figure 9. In summary, the new scalar field is defined by:

$$f_{\tau,S}(\mathbf{p}) = f_{1,w_\tau(S)}(w_\tau(\mathbf{p})) \tag{8}$$

where $f_{1,S}$ is the application of the basic convolution (equation (2)) along the skeleton S, and where w is defined by:

$$w_\tau(\mathbf{p}) = \frac{1}{\tau}\mathbf{p} \tag{9}$$

We can easily verify that this new field is scale-invariant:

$$f_{\alpha\tau,\alpha S}(\alpha\mathbf{p}) = f_{1,\frac{\alpha S}{\alpha\tau}}(\frac{\alpha\mathbf{p}}{\alpha\tau}) = f_{1,\frac{S}{\tau}}(\frac{\mathbf{p}}{\tau}) = f_{\tau,S}(\mathbf{p})$$

We also note that the favourable properties of convolution surfaces are maintained: if a segment-skeleton splits into pieces, the two parts transformed through the warp will still be two halves of a segment on which a standard convolution is computed. Therefore, SCALIS models are independent from the degree of subdivision of the skeleton, and smoothly blend when the fields are summed. As for convolution surfaces, this enables arbitrary graphs of curve-segments and of surface-patches to be used as skeletons, the latter being split into adequate sets of line-segments or of triangles for computing f. However, as for convolution surfaces, modeling general shapes will require SCALIS primitives of non constant radius, which we present next.

4. Enabling radius variations

For the sake of simplicity, we first extend SCALIS to non-constant radii in the case of curve-skeletons. Then, the solution is generalized to higher-order skeletons. Finally, we discuss the way to compute closed-form expressions in order to accelerate field queries.

4.1. Case of curve-skeletons

Let us suppose that a varying weight $\tau(s)$ has been defined along a curve-skeleton. The simple idea developed in the previous section, namely using τ as the control parameter of a space-warp (equation (8)) to get scale-invariance, cannot be applied anymore. However, we can extend the model by passing it to the limit, i.e. by reasoning about different local warps (each depending on the local weight value) applied to each infinitesimal arc-lengths of the skeleton.

Let us take a look at the field contribution of an infinitesimally small element ds of a curve-skeleton, as depicted in figure 10. The local space warp around s would modify distances by a factor $\frac{1}{\tau(s)}$. Thus the influence of the element ds is given by $k(\frac{1}{\tau(s)}\|\Gamma(s)-\mathbf{p}\|)$. Nevertheless, we should remember that the warp is also applied to the skeleton. Thus, the infinitesimally small arc length becomes $\frac{ds}{\tau(s)}$. Replacing the discrete sum over infinitesimal segments by an integral, it yields:

$$f(\mathbf{p}) = \int_{s\in S} k\left(\frac{\|\Gamma(s)-\mathbf{p}\|}{\tau(s)}\right)\frac{ds}{\tau(s)} \qquad (10)$$

Compared with the expression used by Hornus (equation (7)), our formulation consists of giving a density to the skeleton, this latter being inversely proportional to the local weight. Moreover, since integrating increases the degree of regularity of a function by one, our new formulation is C^∞ for *Inverse* and *Cauchy* kernels and $C^{\lfloor\frac{i+1}{2}\rfloor}$ for *Compact Polynomial* kernel.

4.2. Extension to skeletons of higher dimension

In the case of surface skeletons, the same method can be used. However, the resulting formulation is slightly different since scaling an infinitesimal surface element $dudv$ requires scaling it in both directions, by dividing it by $\tau(u,v)^2$. Therefore, the expression of SCALIS of non-constant radius on triangle-skeletons uses a factor $\frac{1}{\tau(u,v)^2}$ instead of $\frac{1}{\tau(s)}$ in eq. (10):

$$f(\mathbf{p}) = \int_{(u,v)\in S} k\left(\frac{\|\Gamma(u,v)-\mathbf{p}\|}{\tau(u,v)}\right)\cdot\frac{dudv}{\tau(u,v)^2} \qquad (11)$$

Similarly, the formulation would be easy to extend to elementary volume skeletons, as the ones used in [PGMG09], but with a division of the elementary volume by the third power of τ.

4.3. Closed-form solutions for SCALIS

Let us first stress that as for the convolution integral, providing closed-form solutions for the SCALIS integral (equations (10) and (11) is essential to get efficient field queries. The only alternative, when such expressions cannot be found, is numerical integration which implies choosing a trade-of between efficiency and accuracy.

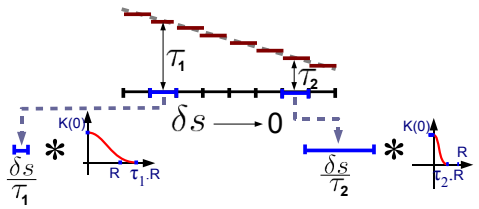


Figure 10: *Passing the space warping method to the limit.*

In the case of triangle-skeletons, a number of closed-form solutions have been developed for various kernels, but only for convolution with constant radius [She99, JTZ09, Hub12, ZJLZ12]. We directly derive our close-form SCALIS solution in the constant radius case from these expressions, using the warping formulation. In other cases, we use numerical integration. Figure 11 depicts two objects created using triangle primitives.

The case of segment-skeletons is easier: general, closed-form solutions have been developed for weighted convolution [HC12]. The integral used in SCALIS being very similar, we extend these solutions below in the case of linear weight variations along segment-skeletons: this is the most useful case in practice, since weights are generally assigned at vertices along poly-line skeletons and linearly interpolated in-between. If needed, subdivision can be used to get smoother variations [AC02].

Segment-primitive with linear radius Assume that we have a segment of line **ab** on which a linearly varying weight $\tau(t) = \tau_0 + \Delta\tau\, t$ is defined (parameterized by $t \in [0,1]$) corresponding to the wanted radius around the segment. We introduce the following parameterization of the segment on $[0,1]$:

$$\Gamma(t) = \mathbf{a} + t\ (\mathbf{b}-\mathbf{a})$$

Then, we have (with $\mathbf{p}$ the query point):

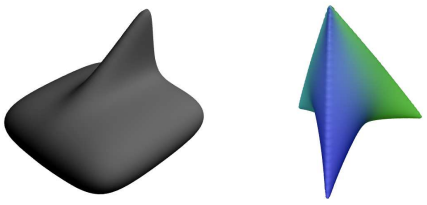
$$\|\mathbf{p}-\Gamma(t)\|^2 = \|\mathbf{b}-\mathbf{a}\|^2\, t^2 - 2(\mathbf{b}-\mathbf{a}).(\mathbf{p}-\mathbf{a})\, t + \|\mathbf{p}-\mathbf{a}\|^2$$

Injecting this relation into equation (10) yields to the formulae given in table 2 in the appendix. The antiderivatives we get are either of the form:

$$\int w(t)(at^2-2bt+c)^{\frac{i}{2}}\, dt \text{ with } w(t) \text{ polynomial and } i \in \mathbb{Z}$$

or

$$\int \frac{(at^2-2bt+c)^{\frac{i}{2}}}{(dt+e)^k}\, dt \text{ with } k \in \mathbb{N}^* \text{ and } i \in \mathbb{N}$$



(a) (b)

Figure 11: *Objects created using triangle primitives: (a) triangles with a constant thickness using a closed-form solution; (b) triangles with linearly varying thickness using numerical integration.*

Thus, we can obtain closed-form expressions of the scalar field by following the same principle as in [HC12, Hub12]: we find the closed-form associated with the first formula by directly applying the recurrence formula given in [HC12, Hub12]. If i is even, closed-form of the second integral can be easily found by expanding the numerator and then applying k integrations by parts. The recurrence formulae are given in table 4 and 5 of the appendix.

Note that applying the differentiation rule under the integral sign (Leibniz' rule) yields closed-form expressions for the gradient of the scalar field as well. Indeed, it is sufficient to know the differential of the following functions in order to obtain the expression of the gradient as an integral:

* $N : \mathbf{p} \mapsto \|\mathbf{p}-\mathbf{a}\|^2 \;\Rightarrow\; \vec{\nabla}N : \mathbf{p} \mapsto 2(\mathbf{p}-\mathbf{a})$
* $S : \mathbf{p} \mapsto (\mathbf{b}-\mathbf{a}).(\mathbf{p}-\mathbf{a}) \;\Rightarrow\; \vec{\nabla}S : \mathbf{p} \mapsto \mathbf{b}-\mathbf{a}$
* $R : \mathbf{p} \mapsto \dfrac{1}{h(\mathbf{p})^{\alpha}} \;\Rightarrow\; \vec{\nabla}R : \mathbf{p} \mapsto -\alpha\dfrac{\vec{\nabla}h(\mathbf{p})}{h(\mathbf{p})^{\alpha+1}}$
* $M : \mathbf{p} \mapsto (h(\mathbf{p}))^{\alpha} \;\Rightarrow\; \vec{\nabla}M : \mathbf{p} \mapsto \alpha\vec{\nabla}h(\mathbf{p}).(h(\mathbf{p}))^{\alpha-1}$

These integral formula are given in table 2. The same recurrence formula as for scalar fields can be used to find closed-form solutions of the gradients, which was never done before with previous models. In practice, as suggested in [HC12], we use Maple to unroll the recurrence and create optimized code for the evaluation of our functions. The number of required operations for each field or gradient query is given in table 3 of the appendix. Note that when both f and its gradient are needed, the number of operations can be optimized since many terms appear in both expressions.

Case of Inverse kernels: We note that the scale-invariant formulation associated to the *Inverse* kernel of order i corresponds to the use of the weight $(\Delta\tau.t+\tau_0)^{i-1}$ in the standard formulation (eq. 6).

Case of Compact Polynomial kernels: In order to compute the integral associated to *Compact Polynomial* kernels, one should first select the part of the skeleton that is inside the finite support of the kernel.

In order to do so, we search t such that $k\left(\frac{\|\Gamma(t)-\mathbf{p}\|}{\tau(t)}\right) > 0$, with k a decreasing function. This is equivalent to studying $1-\frac{\|\Gamma(t)-\mathbf{p}\|}{\sigma\tau(t)} > 0$, or $\|\Gamma(t)-\mathbf{p}\|^2 < \sigma^2\,\tau^2(t)$ with $\sigma^2\,\tau^2(t) = \sigma^2\Delta\tau^2\,t^2 + 2\sigma^2\Delta\tau\tau_0\,t + \sigma^2\tau_0^2$.

This yields a second degree relation:

$$\begin{array}{lcl} & (\|\mathbf{b}-\mathbf{a}\|^2 - R^2\Delta\tau^2) & t^2 \\ - & 2((\mathbf{b}-\mathbf{a}).(\mathbf{p}-\mathbf{a}) + R^2\Delta\tau\tau_0) & t \\ + & \|\mathbf{p}-\mathbf{a}\|^2 - R^2\tau_0^2 & \leq 0 \end{array}$$

We solve it by studying the sign of its discriminant. Note that from geometric considerations, we are ensured that the solution is always a connected part of $[0,1]$ (which is important since having to use two distinct intervals would double the evaluation cost).

An example of scalar field obtained is shown in figure 12(b), we can note that the kernel support smoothly changes along the skeleton making it easier to model shapes with a large variation in radius.

5. Achieving radius control

This section presents our last contribution to skeleton-based modeling: we enhance scale-invariant integral surfaces (SCALIS) with direct control of the surface radius along a skeleton. We first present a simple modification of the model enabling to directly use τ to set the desired radius. This radius is reached wherever the skeleton is long enough relative to the local width of the kernel. In Section 5.2, we improve the model in order to provide radius control in other cases as well for segment-skeletons.

5.1. Relating τ to the radius

Let us come back to the general expression for SCALIS on line-segments (equation (10)). We first remark that a slight modification of this expression can bring an important benefit: the model can be set so that $\tau(s)$ directly sets the radius of the resulting shape. This is done in the following way:

Let us consider a line (a segment-skeleton of infinite length) of constant weight τ, and define f_τ as the field value at a distance τ from the line. It happens that this value is independent from τ due to the scale invariance of our model (the value only depends on the convolution kernel used, see table 6). Let us now define $N_k(c) = \frac{c}{f_\tau}$, and then renormalize the scalar field by the inverse of this value. This sets the surface of interest, associated to the iso-value c around the skeleton, to the prescribed distance τ from the line. The normalization factor being independent from τ, we redefine SCALIS primitives generated by segment-skeleton as:

$$f(\mathbf{p}) = \frac{1}{N_k(c)} \int_{s\in S} k\left(\frac{\|\Gamma(s)-\mathbf{p}\|}{\tau(s)}\right) \frac{ds}{\tau(s)} \qquad (12)$$

with $N_k(c)$ being a normalization factor depending only on the iso-value of interest c. The same methodology can be used for higher dimension primitives.

This improved model provides an explicit control of the surface radius, directly controlled by the choice of τ. Moreover, for *Compact Polynomial* kernels, the prescribed value is exactly reached wherever there is a sufficient portion of segment-skeleton (respectively, of surface-skeleton) of similar weight near the point of interest which is due to their compact support. The required length L to get a radius τ is given by the Pythagore theorem (see figure 13): $\tau^2 + (\frac{L}{2})^2 = (\sigma\tau)^2$, so the needed length is $L = 2\tau\sqrt{\sigma^2-1}$. Note that it is proportional to τ, thus small and thin primitives can be set to a prescribed radius as easily as large and wide ones.

The radius of the generated primitive can still be smaller than τ at extremities of primitives, for short skeletons and

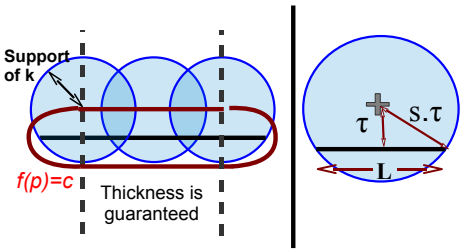


Figure 13: *Exact radius control for compact polynomial primitives. The portion of skeleton that influences a point in space are the ones that are at a distance smaller than the kernel radius. Therefore, a point at distance τ of the skeleton has to be in the area of influence of a length L of skeleton for the iso-surface to pass through.*

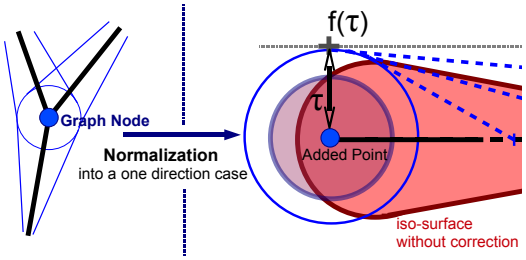


Figure 14: *Correction of the scalar field around a node of maximal radius in the skeleton graph. A "normalized" version of the node is used to compute the lacking field value.*

for extreme radii variations. A last extension of SCALIS that handles these cases is presented next.

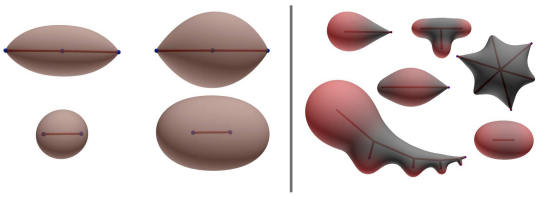
5.2. Guaranteeing radius in extreme cases

As mentioned in the previous section, the radius of a SCALIS primitive will be smaller than the prescribed value τ if a given minimal length of skeleton of similar radius is not available around the query point; this length is proportional to the prescribed radius. Thus, the shape typically gets thinner in a few typical situations: near the *end-point* of a skeleton and the *local maxima* of radius.

We develop a practical solution when the skeleton is a graph of poly-lines, which eases skeleton-based modeling and improves the reconstruction in sketch-based modeling applications (see Section 6). It consists of automatically adding the lacking length of integration at some graph nodes depending on their "1-ring" neighborhood.

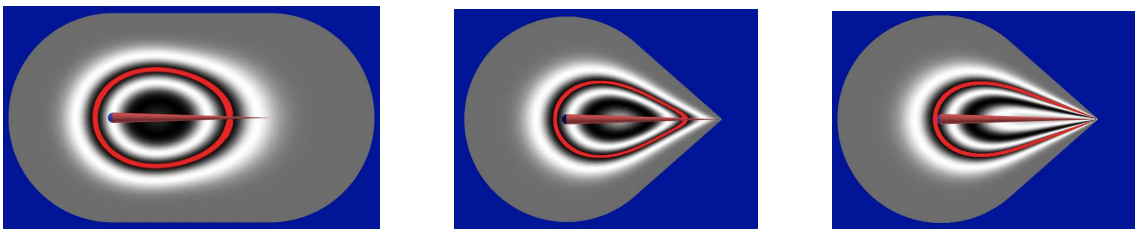
At the extremities of the skeleton, we add an extra segment-skeleton of constant weight in continuation of the ending skeleton branch set so that the desired radius τ is exactly reached at the end-point of the shape. Because of the scale-invariance property of SCALIS primitives, we only have to pre-compute the needed length l_1 for the case $\tau = 1$ and then set the length of extra skeleton to $\tau\, l_1$.

The second case (i.e. maximal radius) is more difficult to handle since there is no clear direction in which to add a segment-skeleton. For this reason we choose to concentrate all the needed length of skeleton at the point **s** of the skeleton with the maximal desired radius. This is equivalent to adding



(a) (b)

Figure 15: *(a) Comparison between SCALIS models before (left) and after (right) radius correction, (b) some simple correction of radii (appearing in red), just note that the "star" shape do note have correction since the blending of all leaving branch is sufficient to obtain the wanted radius in the "normalized" configuration.*



(a) (b) (c)

Figure 12: *Scalar fields generated integrating a compact support kernel along segment of linearly varying weight, with* 0 *value at one end: (a) Standard formulation (eq. 6), (b) Hornus formulation (eq. 7), (c) Scale-invariant formulation (SCALIS). As we can see, only SCALIS formulation enables the creation of the nice sharp point.*

a Dirac in the field integral or to adding a standard point-skeleton primitive to the skeleton. This added point uses the same kernel as SCALIS to compute its scalar field:

$$f(\mathbf{p}) = w\, k\left(\frac{\|\mathbf{s}-\mathbf{p}\|}{\tau}\right) \tag{13}$$

with τ the desired radius in S and w a weighting value computed as to get this required radius. The weight w is computed by looking at a "normalized" version of the surface behavior around this node of the skeleton graph: we consider that all the segments connected to this graph node point to the same direction and that they continue until their prescribed radius falls to zero (see figure 14). This was chosen to be independent from skeleton subdivision, and it enables to have only a local processing. In this configuration, the lacking field value at a distance τ is given by $c - f(\tau)$, where $f(\tau)$ is the field value of the normalized case at a distance τ in a direction orthogonal to the segments primitives (see figure 14). From this, we get $w = \frac{c-f(\tau)}{k(1)}$.

Note that this solution for insuring prescribed radii does not require any optimization step, which makes it fast and stable, and is coherent with the pleasant summation blend. Due to the addition of a skeleton point, the level of regularity of the resulting field is now the one of the kernel. Some base results are depicted in figure 15 and the correction is also used in more complex models (Figure 1 and 18). Multiple corrections in the same area could lead to unwanted bulging in some special cases. Even if those cases are unusual, it is possible to add safeguards by using a barycentric blending (with weights equal to individual field contribution) to blend the correctors together: this would prevent the added correction to be more important than the maximal one.

6. Results and discussion

6.1. Implementation details

Kernel choice: For all interactive applications, we used the *Compact Polynomial* kernel (equation (5)) of order $i = 6$. This choice is dictated by several consideration: thanks to its local support, it saves computational time while limiting blending at distance. The improvement of efficiency is due to the restriction of evaluation to tight bounding-boxes (with other kernel this would impact field regularity and create local artifact). As the degree i increase both the degree of continuity of the surface and computational cost increase. Thus we advice to use a degree $i = 6$ as a trade-off : it is the first one for which the Hessian of the field is continuous (ie - continuous curvature along the surface). The shape of figure 12(b) cannot be modeled with a support of constant radius (so, with standard convolution), since it has both a large smooth part and thin part.

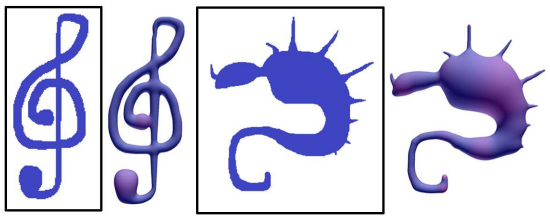


Figure 16: *Sketch based modeling examples, area affected by the radius correction method appear in purple.*

Rendering method: In order to provide interactive feedback during interactive modeling sessions, we used a local marching cube method [Blo94], enabling to recompute only the edited parts of the surface. Computation times for the whole meshes are given in Table 1. Just note that the efficiency is impacted by the needed resolution required to get small details. This can become really problematic when the difference between minimal and maximal radius become large. This is the case if we want to mesh the whole ant model with a resolution that enable us to obtain good detail quality that allows a close-up of the mandibles. For this reason, an improvement would be to use a dual marching cube method on an octree [JLSW02] whose construction would be guided by the BlobTree. First test shows that for a shape such as the ant, we can benefit from a speed-up factor of ten, despite the fact that the code used can still be further optimized.

6.2. Applications

We tested Scale-invariant Integral Surfaces (SCALIS) in three use cases: interactive constructive modeling with skeletons, shape reconstruction from medial axis data, and sketch-based modeling (see the associated video). In practice, we use our new implicit primitives in a BlobTree structure, enabling use of classical composition operators.

Skeleton-based modeling: To validate the fact that skeleton-based modeling is a good paradigm at conceptual stages of design, we set up a prototype of an interactive modeling system, where a user can create a shape from scratch by manipulating segment and triangle skeletons and prescribing the desired radii at the nodes of the resulting graph. Figure 1(a) depicts a dancer model created by a computer artist, novice to our system, in less than one hour. Note that this model includes parts of quite different scales (the whole body and the hand), validating the benefits of our approach. In practice, the hand model was designed at a larger scale and then scaled down to be blended with the arm still enabling further editing.We also used our system to model an ant with very sharp details (figure 18) validating the fact that thin details are not smoothed out with SCALIS. Furthermore, we can note on figure 17 that the thickness of the iso-surface match the radii given at skeleton vertices.

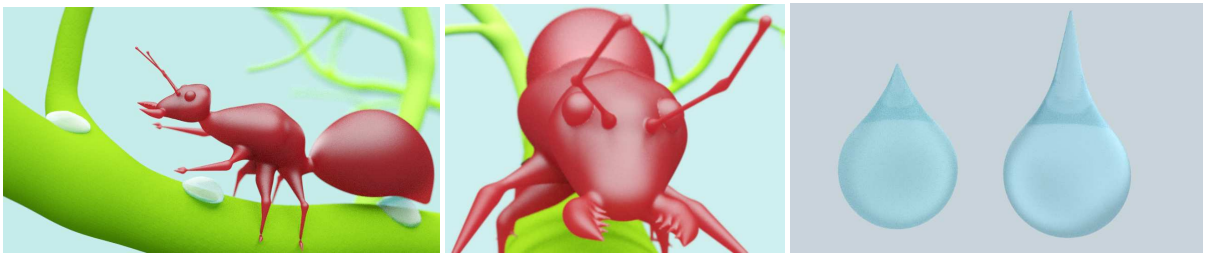
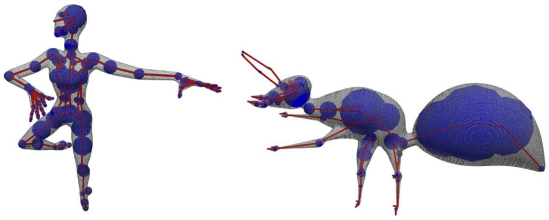


Figure 18: *Implicit ant on implicit branch. The zoom on the head of the ant show that the details on the mandible are not blurred (they still look to have sharp points). Right picture show 2 drops of water created from 3 segment scale-invariant primitives each and by applying the method of thickness correction.*

	Number of Triangles	Computation Times
Dancer 1(a)	93 984	0.32s
Elf 1(b)	214 364	2.64s
Ant (middle Res) 18(a)	190 058	0.72s
Ant (HiRes)	1 714 544	17.16s
Ant's Head (HiRes) 18(b)	166 568	0.73s
Ant (non-uniform)	97 772	1.28s

Table 1: *Computation time of a marching-cube on an Intel Core 2 Duo (2.4 GHz, 1 core used). Last row is the time for an octree-based dual marching cube.*

Sketch-based modeling: In practice, moving skeletons in a 3D space can be intricate for novice users: A good way to make interactive shape design even more intuitive is to set up a progressive sketch-based modeling system. SCALIS is particularly well adapted to this end, since it is able to generate 3D shapes that fit 2D contours directly, with no optimization step, and to seamlessly blend them into the current model using a simple sum of field contributions. Precise fitting of the sketched 2D contour is achieved by converting it to a medial axis representation, and to use it as the skeleton of a SCALIS model, as was done in [ITZkF04] for convolution surfaces. Since for 2D figures, the medial axis is just a graph of polylines with no surface parts, the methods we describe in Section 5 are sufficient to get a good fitting of the contour. See figure 16, and figure 1(c) as example.



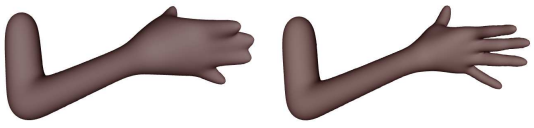
(a) (b)

Figure 17: *Skeletons of the dancer (a) and ant (b) with the model in wireframe.*

Reconstruction from medial-axis: Our last application is the off-line reconstruction of a model from a scanned object. We used SCALIS to directly generate the shape from a medial axis already extracted from scanned data of a real tree. SCALIS is a very good choice for this application: firstly, the field function enables to compute a pseudo-distance between data-points and the iso-surface of interest providing an error measure on the quality of reconstruction; in addition, the resulting model is very compact and can be easily edited through direct skeleton manipulation; lastly, the scale-invariant blending behavior of SCALIS produces smooth and self-similar branching at different scales, a very desirable behavior for a fractal-like self-similar object such as a tree (figure 1(b)).

6.3. Discussion

We already mentioned the advantages of our model over convolution, namely radius control, non-blurring & non-vanishing details and scale invariant blending. Based on these properties, SCALIS is able to capture smooth shapes with sharp features, as those of figure 18, 19 and 20 (see the ant's mandible), that could not easily be modeled with previous implicit modeling techniques. If we take a look at figure 19 with the hand of the dancer (in this case modeled with the *Cauchy* kernel), we can see that the shape we would have obtained using classic convolution is not acceptable: the fingers are nearly lost. Furthermore, ant's abdomen would have been difficult to design with standard convolu-



(a) (b)

Figure 19: *(a) Hand of the dancer modeled with standard convolution, note the unwanted blurring of the fingers, (b) hand modeled with SCALIS.*

tion model with *Compact Polynomial* kernel since there is a large difference between the maximal and the minimal radius.

One should also note that SCALIS works with any of the Kernel functions of equations (3), (4) and (5), and in particular for any choice of their parameter σ. Since our model maintains the prescribed radius around skeletons and that σ controls the based width of kernel support (hence the blending properties), our model can reconstruct shapes that tend toward an exact reconstruction of the medial axis data : this is achieved by making σ tends towards 1 for *Compact Polynomial* kernels. A last benefit of our approach is to provide closed form solutions for both the field values and their gradient, which is useful to accelerate tessellation or ray-tracing of the models.

SCALIS also suffers from a few drawbacks, discussed below:

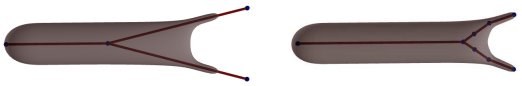
- SCALIS is more expensive to compute than former models (see Table 3 in the appendix for the comparison between SCALIS and standard weighted convolution). However, SCALIS is particularly well adapted to compact support kernels, which enable local field queries. This makes the use of our model still possible in interactive applications.
- Scale invariant blending may not be always desired: In SCALIS, when a thin segment is blended into a thick one, its shape is very well preserved, even in the blending area. In some cases, a smoother blending may be desired. A practical solution for the user is to split the thin segment and manually add the desired transition of radius.
- Loss of "superposition" property: One of the advantages of standard convolution surfaces (equation (6)) is that whenever we stack two skeletons with weight τ_1 and τ_2, then the resulting surface is equivalent to the one generated from a single segment of weight $\tau_1 + \tau_2$. This property made it easy to create "Y" branching junctions where a main branch subdivides into two thinner ones without suffering from any bulge. The loss of this property is to be mitigated: the quality of the shape produced highly depends on the kernel used and on its σ parameter (see figure 21 (a) versus figure 22(a)). In case the shape produced by the superposition property is intuitive as in figure 21 (a), SCALIS enables to model it using another rule : $\tau^2 = \tau_1^2 + \tau_2^2$ (see figure 21(c)). τ being the radius of the model, this new rule expresses the fact that the area of the cross section of the shape before and after the branching point is preserved. In other cases (see figure 22), we may need a different skeleton graph to get the same shape. Then our skeleton is closer to the medial axis of the shape and thus more intuitive to manipulate.

(a) (b)

Figure 20: *The positioning of two scale-invariant primitives in a "V" shape illustrates the good behavior of the SCALIS blending, which is the same at both extremities of the shape (we can note that the thin extremity is not blurred). (a) blending with a "max", (b) blending with a "+".*

(a) (b) (c)

Figure 21: *Comparison between two types of branching junction: (a) Standard convolution model with $\tau_1 + \tau_2 = \tau$, (b) SCALIS with $\tau_1 + \tau_2 = \tau$ (τ now controls the radius), (c) SCALIS with $\tau_1^2 + \tau_2^2 = \tau^2$.*



(a) (b)

Figure 22: *(a) Superposition property for standard convolution with Compact Polynomial kernel of degree 6 with $\sigma = 2.0$ and $\tau = 1.16$, (b) similar shape designed with scale-invariant integral surfaces. Note that the skeleton of (b) is much closer to the medial axis.*

7. Conclusion and Future work

Despite of the compactness of this representation and of their interest for conceptual shape design, skeleton implicit surfaces are not often used as a modeling primitive. In this work, we solved the problem of generating smooth, solid shapes from skeletal structures and radius information. This was done by introducing Scale invariant Integral Surfaces (SCALIS). Built on convolution surfaces, SCALIS inherited their favourable properties such as blending with a sum, and shape independence from skeleton refinement. But contrary to convolution surfaces, they bring radius control mechanisms and behave similarly at different scales. This allows reconstruction from medial data and yields intuitive editing.

Despite of the closed-form expressions we provided for f and for its gradient for segment skeletons, there is room for improvement concerning the efficiency of the model: avenues for future work include finding closed-form expressions for triangle with tri-linearly varying weight, or improving and parallelizing the evaluation of the scalar field and the meshing process.

Lastly, implicit surfaces are well known for their undesired blending at distance behavior. Although reduced with SCALIS, since the sharpness of the field automatically adapts to shape size, such behavior can still occur. An interesting direction for future work is therefore to guarantee that the surface has the same topology as its skeleton. We would like to do that while keeping a sum-like blending. Because of its scale invariance, we expect our model to ease the analysis of this problem. The scale parameter σ in the definition of the kernel now controls the sharpness of blending independently from the desired radius: it could be varied along the skeleton to prevent unwanted blending with other parts. This is not an easy solution, though, since methods for providing some smooth variations of σ need to be designed.

Acknowledgements

We also thank the anonymous reviewers for their useful comments and suggestions. This work was partially funded by the ERC advanced grant EXPRESSIVE.

References

[AC02] ANGELIDIS A., CANI M.-P.: Adaptive implicit modeling using subdivision curves and surfaces as skeletons. In *7th ACM Symposium on Solid Modeling and Applications, June, 2002* (Saarbrucken, Allemagne, June 2002), pp. 45–52.

[AHLD07] AUJAY G., HÉTROY F., LAZARUS F., DEPRAZ C.: Harmonic Skeleton for Realistic Character Animation. In *ACM-SIGGRAPH/Eurographics Symposium on Computer Animation* (2007), ACM.

[Bar84] BARR A. H.: Global and local deformations of solid primitives. *Special issue: SIGGRAPH84 - Comput. Graph. 18* (January 1984), 21–30.

[BBCW10] BERNHARDT A., BARTHE L., CANI M.-P., WYVILL B.: Implicit blending revisited. *Comput. Graph. Forum 29*, 2 (May 2010), 367–375.

[BGSF08] BIASOTTI S., GIORGI D., SPAGNUOLO M., FALCIDIENO B.: Reeb graphs for shape analysis and applications. *Theor. Comput. Sci. 392*, 1-3 (Feb. 2008), 5–22.

[BJD*12] BOROSÁN P., JIN M., DECARLO D., GINGOLD Y., NEALEN A.: Rigmesh: automatic rigging for part-based shape modeling and deformation. *ACM Trans. Graph. 31*, 6 (Nov. 2012), 198:1–198:9.

[BL99] BLOOMENTHAL J., LIM C.: Skeletal methods of shape manipulation. In *Proceedings of the International Conference on Shape Modeling and Applications* (1999), SMI '99, IEEE Computer Society.

[Bli82] BLINN J. F.: A generalization of algebraic surface drawing. *ACM Trans. Graph. 1*, 3 (July 1982), 235–256.

[Blo94] BLOOMENTHAL J.: Graphics gems iv. 1994, ch. An implicit surface polygonizer, pp. 324–349.

[Blo02] BLOOMENTHAL J.: Medial-based vertex deformation. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2002), SCA '02, ACM, pp. 147–151.

[BPCB08] BERNHARDT A., PIHUIT A., CANI M.-P., BARTHE L.: Matisse: Painting 2D regions for modeling free-form shapes. In *EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling, SBIM 2008, June, 2008* (Annecy, France, June 2008), Alvarado C., Cani M.-P., (Eds.), pp. 57–64.

[BS91] BLOOMENTHAL J., SHOEMAKE K.: Convolution surfaces. In *Proceedings SIGGRAPH '91* (New York, NY, USA, 1991), ACM, pp. 251–256.

[BWdG04] BARTHE L., WYVILL B., DE GROOT E.: Controllable binary csg operators for "soft objects". *International Journal of Shape Modeling 10*, 2 (2004), 135–154.

[dGWvdW09] DE GROOT E., WYVILL B., VAN DE WETERING H.: Locally restricted blending of blobtrees. *Computers & Graphics 33*, 6 (2009), 690 – 697.

[HAC03] HORNUS S., ANGELIDIS A., CANI M.-P.: Implicit modelling using subdivision curves. *Visual Comput. 19*, 2-3 (May 2003), 94–104.

[HC12] HUBERT E., CANI M.-P.: Convolution surfaces based on polygonal curve skeletons. *Journal of Symbolic Computation 47*, 6 (2012), 680 – 699.

[Hub12] HUBERT E.: Convolution surfaces based on polygons for infinite and compact support kernels. *Graphical Models 74*, 1 (2012), 1 – 13.

[JLSW02] JU T., LOSASSO F., SCHAEFER S., WARREN J.: Dual contouring of hermite data. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), SIGGRAPH '02, ACM, pp. 339–346.

[JT02a] JIN X., TAI C.-L.: Analytical methods for polynomial weighted convolution surfaces with various kernels. *Computer Graphics 26*, 3 (2002), 437–447.

[JT02b] JIN X., TAI C.-L.: Convolution surfaces for arcs and quadratic curves with a varying kernel. *The Visual Computer 18*, 8 (2002), 530–546.

[JTFP01] JIN X., TAI C.-L., FENG J., PENG Q.: Convolution surfaces for line skeletons with polynomial weight distributions. *J. Graph. Tools 6*, 3 (2001), 17–28.

[JTZ09] JIN X., TAI C.-L., ZHANG H.: Implicit modeling from polygon soup using convolution. *Vis. Comput. 25*, 3 (Feb. 2009), 279–288.

[Lon98] LONCARIC S.: A survey of shape analysis techniques. *Pattern Recognition 31* (1998), 983–1001.

[ITZkF04] LAN TAI C., ZHANG H., KIN FONG J. C.: Prototype modeling from sketched silhouettes based on convolution surfaces. *Computer Graphics Forum 23* (2004), 71–83.

[MS98] MCCORMACK J., SHERSTYUK A.: Creating and rendering convolution surfaces. *Computer Graphics Forum 17*, 2 (1998), 113–121.

[PGMG09] PEYTAVIE A., GALIN E., MERILLOU S., GROSJEAN J.: Arches: a Framework for Modeling Complex Terrains. *Computer Graphics Forum (Proceedings of Eurographics) 28*, 2 (2009), 457–467.

[PPIK02] PASKO G., PASKO A., IKEDA M., KUNII T.: Bounded blending operations. In *Proceedings of the Shape Modeling International 2002 (SMI'02)* (Washington, DC, USA, 2002), SMI '02, IEEE Computer Society, pp. 95–.

[PPK05] PASKO G. I., PASKO A. A., KUNII T. L.: Bounded blending for function-based shape modeling. *IEEE Comput. Graph. Appl. 25*, 2 (Mar. 2005), 36–45.

[Ric73] RICCI A.: Constructive geometry for computer graphics. *Computer Journal 16*, 2 (1973), 157–60.

[She99] SHERSTYUK A.: Kernel functions in convolution surfaces: A comparative analysis. *The Visual Computer 15*, 4 (1999), 171–182.

[SPB96] SHERBROOKE E., PATRIKALAKIS N., BRISSON E.: An algorithm for the medial axis transform of 3d polyhedral solids. *Visualization and Computer Graphics, IEEE Transactions on 2*, 1 (1996).

[WGG97] WYVILL B., GALIN E., GUY A.: The Blob Tree, Warping, Blending and Boolean Operations in an Implicit Surface Modeling System. *Submitted to The Visual Computer Journal* (July 1997). Submitted.

[WMW86] WYVILL G., MCPHEETERS C., WYVILL B.: Data structure for soft objects. *The Visual Computer 2*, 4 (Aug. 1986), 227–234.

[WO97] WYVILL B., OVERVELD K. V.: Warping as a modelling tool for csg/implicit models. In *Proceedings of the 1997 International Conference on Shape Modeling and Applications (SMI '97)* (Washington, DC, USA, 1997), IEEE Computer Society, pp. 205–214.

[ZJLZ12] ZHU X., JIN X., LIU S., ZHAO H.: Analytical solutions for sketch-based convolution surface modeling on the gpu. *The Visual Computer 28*, 11 (2012), 1115–1125.

Kernel	Formula
Cauchy	$\mathcal{C}_{g,i}(\mathbf{p}) = \|\mathbf{b}-\mathbf{a}\| \int_0^1 \frac{(\Delta\tau\, t+\tau_0)^{i-1}}{\left((l^2+\Delta\tau^2)t^2-2(uv-\Delta\tau\,\tau_0)t+(d^2+\tau_0^2)\right)^{\frac{i}{2}}}\, dt$ $\vec{\nabla}\mathcal{C}_{g,i}(\mathbf{p}) = \frac{i}{\sigma^2}\|\mathbf{b}-\mathbf{a}\| \int_0^1 \frac{\mathbf{h}(t)(\Delta\tau\, t+\tau_0)^{i-1}}{\left((l^2+\Delta\tau^2)t^2-2(uv-\Delta\tau\,\tau_0)t+(d^2+\tau_0^2)\right)^{\frac{i+2}{2}}}\, dt$
Inverse	$\mathcal{P}_{g,i}(\mathbf{p}) = \|\mathbf{b}-\mathbf{a}\| \int_0^1 \frac{(\Delta\tau\, t+\tau_0)^{i-1}}{\left(l^2\, t^2-2\, uv\, t+d^2\right)^{\frac{i}{2}}}\, dt$ $\vec{\nabla}\mathcal{P}_{g,i}(\mathbf{p}) = \frac{i}{\sigma^2}\|\mathbf{b}-\mathbf{a}\| \int_0^1 \frac{\mathbf{h}(t)(\Delta\tau\, t+\tau_0)^{i-1}}{\left(l^2\, t^2-2\, uv\, t+d^2\right)^{\frac{i+2}{2}}}\, dt$
Compact Polynomial	$\mathcal{R}_{g,i}(\mathbf{p}) = \|\mathbf{b}-\mathbf{a}\| \int_{l1}^{l2} \frac{\left((\Delta\tau^2-l^2)t^2-2(-\Delta\tau\,\tau_0-uv)t+(\tau_0^2-d^2)\right)^{\frac{i}{2}}}{(\Delta\tau\, t+\tau_0)^{i+1}}\, dt$ $\vec{\nabla}\mathcal{R}_{g,i}(\mathbf{p}) = \frac{i}{\sigma^2}\|\mathbf{b}-\mathbf{a}\| \int_{l1}^{l2} \mathbf{h}(t) \frac{\left((\Delta\tau^2-l^2)t^2-2(-\Delta\tau\,\tau_0-uv)t+(\tau_0^2-d^2)\right)^{\frac{i-2}{2}}}{(\Delta\tau\, t+\tau_0)^{i+1}}\, dt$

Table 2: ***Integrals for SCALIS segments with linear variation of radius, we use simplified notations :*** $l^2 = \frac{\|\mathbf{b}-\mathbf{a}\|^2}{\sigma^2}$, $d^2 = \frac{\|\mathbf{p}-\mathbf{a}\|^2}{\sigma^2}$, $uv = \frac{(\mathbf{b}-\mathbf{a}).(\mathbf{p}-\mathbf{a})}{\sigma^2}$ ***et*** $\mathbf{h}(t) = (\mathbf{b}-\mathbf{a})\, t-(\mathbf{p}-\mathbf{a})$

	f				∇f				$f, \nabla f$			
	$+$	$*$	$/$	$(\sqrt{},atan,\ln)$	$+$	$*$	$/$	$(\sqrt{},atan,\ln)$	$+$	$*$	$/$	$(\sqrt{},atan,\ln)$
SCALIS Cauchy 4	29	38	4	(1,2,1)	38	62	6	(1,2,0)	47	81	6	(1,2,1)
SCALIS Inverse 3	25	33	5	(5,0,1)	32	58	5	(2,0,0)	40	71	5	(5,0,1)
SCALIS Comp. Polynom. 4	43	82	3	(0,0,1)	37	67	3	(0,0,0)	50	102	3	(0,0,1)
SCALIS Comp. Polynom. 6	71	144	3	(0,0,1)	62	125	3	(0,0,0)	83	172	3	(0,0,1)
Standard Inverse 3 Linear	18	20	4	(2,0,0)	26	46	4	(2,0,0)	29	50	4	(2,0,0)
Standard Cauchy 4 Linear	21	26	4	(1,2,0)	29	40	6	(1,2,0)	32	58	6	(1,2,0)
Standard Cauchy 4 Cubical	27	34	4	(1,2,1)	36	59	6	(1,2,0)	45	77	6	(1,2,1)

Table 3: ***Number of required operations for the evaluation of the closed-form solutions associated to standard convolution and SCALIS, for different kernels.***

$I_{k,i}(a,b,c) = \int \frac{t^k}{(at^2-2bt+c)^{\frac{i}{2}}}\,dt$ with $k \in \mathbb{N}$ and $i \in \mathbb{Z}$
Base cases
$I_{0,1}(a,b,c) = \frac{1}{\sqrt{a}} \ln\left(\frac{at-b}{\sqrt{a}} + (at^2-2bt+c)^{\frac{1}{2}}\right)$ $I_{0,2}(a,b,c) = \frac{1}{\sqrt{ac-b^2}} \arctan\left(\frac{at-b}{\sqrt{ac-b^2}}\right)$
Recurrence for $k=0$
$(i-2)(ca-b^2)I_{0,i} + a(3-i)I_{0,i-2} = \frac{at-b}{(at^2-2bt+c)^{\frac{i-2}{2}}}$
Recurrence for $k=1$
$I_{1,i} - \frac{b}{a}I_{0,i} = \begin{cases} \frac{1}{2a}\ln(at^2-2bt+c) & \text{if } i=2 \\ \frac{1}{a(2-i)} \frac{1}{(at^2-2bt+c)^{\frac{i-2}{2}}} & \text{otherwise} \end{cases}$
Recurrence for $k \geq 2$
$aI_{i-1,i} - bI_{i-2,i} - I_{i-3,i-2} = \frac{1}{2-i} \frac{t^{i-2}}{(at^2-2bt+c)^{\frac{i-2}{2}}}$ if $k = i-1$ $a(i-k-1)I_{k,i} + b(2k-i)I_{k-1,i} - c(k-1)I_{k-2,i} = -\frac{t^{k-1}}{(at^2-2bt+c)^{\frac{i-2}{2}}}$ otherwise

Table 4: ***Recurrence formula for computation of closed-form solution of the first type of integral encountered. Note that first recurrence is only valid if*** $at^2-2bt+c>0$***, for more detail refers to [HC12, Hub12].***

$G_{k,i}(d,q) = \int \frac{t^k}{(dt+q)^i}\,dt$ with $k \in \mathbb{N}$ and $i \in \mathbb{N}^*$
Base cases
$G_{0,i}(d,q) = \begin{cases} \frac{1}{d}\ln(dt+q) & \text{if } i=1 \\ \frac{-1}{(i-1)d} \frac{1}{(dt+q)^{i-1}} & \text{otherwise} \end{cases}$ $G_{k,1}(d,q) = (-1)^k \frac{q^k}{d^{k+1}} \ln(dt+q) + \sum_{l=1}^{k} \left(\frac{(-1)^{k+l}}{l} \frac{q^{k-l}}{d^{k-l+1}} t^l\right)$
Recurrence
$G_{k,i}(d,q) = \frac{1}{(i-1)d}\left(k\, G_{k-1,i-1}(d,q) - \frac{t^k}{(dt+q)^{i-1}}\right)$

Table 5: ***Recurrence formula for computation of closed-form solution of the second type of integral encountered.***

Kernel	Formula
Cauchy	$f_{\tau,2} = \pi\frac{\sigma^2}{\sqrt{\sigma^2+1}}$, $f_{\tau,3} = 2\frac{\sigma^3}{\sigma^2+1}$ and $f_{\tau,i} = \frac{\sigma^2}{\sigma^2+1}\frac{i-3}{i-2}f_{\tau,i-2}$
Inverse	$f_{\tau,2} = \pi\sigma^2$, $f_{\tau,3} = 2\sigma^3$ and $f_{\tau,i} = \sigma^2\frac{i-3}{i-2}f_{\tau,i-2}$
Compact Polynomial	$f_{\tau,0} = 2\sigma\sqrt{1-\frac{1}{\sigma^2}}$ and $f_{\tau,i} = \frac{i}{i+1}(1-\frac{1}{\sigma^2})f_{\tau,i-2}$

Table 6: ***Normalization factor for classic kernel.***