# Anisotropic convolution surfaces

Alvaro Javier Fuentes Suárez, Evelyne Hubert, Cédric Zanni

▶ **To cite this version:**

Alvaro Javier Fuentes Suárez, Evelyne Hubert, Cédric Zanni. Anisotropic convolution surfaces. Computers and Graphics, 2019, 82, pp.106-116. 10.1016/j.cag.2019.05.018 . hal-02137325

**HAL Id: hal-02137325**
**https://inria.hal.science/hal-02137325**

Submitted on 22 May 2019

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Anisotropic convolution surfaces

Alvaro Javier Fuentes Suarez[1,*], Evelyne Hubert[1], and Cedric Zanni[2]

[1]Université Côte d'Azur, Inria, France
[2]Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

[*]Corresponding author: alvaro.fuentes-suarez@inria.fr

**Abstract**

Convolution surfaces with 1D skeletons have been limited to close-to-circular normal sections. The new formalism presented here allows for ellipsoidal normal sections. Anisotropy is prescribed on $\mathscr{G}^1$ skeletal curves, chosen as circular splines, by a rotation angle and the three radii of an ellipsoid at each extremity. This lightweight model creates smooth shapes that previously required tweaking the skeleton or supplementing it with 2D pieces. The scale invariance of our formalism achieves excellent radii control and thus lends itself to approximate a variety of shapes. The construction of a *scaffold* is extended to skeletons with $\mathscr{G}^1$ branches. It projects onto the convolution surface as a quad mesh with skeleton bound edge-flow.

*Keywords:* Convolution surfaces, implicit surfaces, medial and skeletal representations, curves and surfaces

## 1 Introduction

Skeletons, as a set of curves and/or surfaces centered inside a shape, provide a compact representation of the shape structure. Due to this property, skeletons have proved useful in many applications ranging from shape analysis to 3D modeling and deformation [46].

Convolution surfaces [9] associate radii information to the skeleton and provide a simple way for users to rapidly define a shape. A convolution surface is an implicit surface defined as a level set of a scalar field, the convolution field, that is obtained by integrating a kernel function over the skeleton. This technique allows to build a complex shape by modeling parts that assemble into a smooth surface, independently of the smoothness of the skeleton. They also represent a volume with the convolution surface as its boundary and can therefore be combined with other composition operators from implicit modeling frameworks [36, 54].

Skeleton-based implicit surfaces, have been extensively used to model a variety of smooth organic shapes, such as animals and trees, either by direct skeleton manipulation [43, 62, 60] or through sketch-based modeling [15, 6, 61, 52], or immersive modeling in virtual environments [63].

Standard convolution surfaces, and previous extensions, allows to model a large range of shape when used with a combination of both 1D and 2D skeletons. The latter are more cumbersome to manipulate during modeling, but using only 1D skeletons is restrictive in terms of the diversity of shape that can be generated. For instance, representing muscles for 3D animation with 1D skeletons require non-circular cross sections [42]. Current solutions for convolution surfaces consist in adding extra skeleton pieces which adds to the complexity, or using spatial warpings, which breaks the smoothness.

We therefore introduce *anisotropic convolution surfaces*, an extension that increases the modeling freedom, providing ellipse-like normal sections around 1D skeletons. We increase the diversity of shapes that can be generated from 1D skeletons, and diminish the need for 2D skeletons, while still retaining smoothness. We achieve anisotropy not just in the normal sections but also in the tangential direction. This allows sharper and steeper radius variation, and control of thickness at skeleton endpoints.

For anisotropic convolution a frame and three radii are associated to the points on the skeletal curves. These can be defined with few parameters along any continuous curve with continuous unit tangent ($\mathscr{G}^1$ *curves*). We thus favor *circular splines*, that is $\mathscr{G}^1$ curves piecewise composed of arcs of circle or line segments, as skeletal curves. Any spatial $\mathscr{G}^1$ curve can be approximated with a circular spline [45] and their Rotation Minimizing Frames [49] are easily computed. As compared to polylines, circular spline require less pieces to obtain a good approximation of skeletal curves with high curvature or torsion, or to obtain visual smoothness of the resulting convolution surface. This furthermore reduces the number of integrals to evaluate for the convolution field. It nonetheless retains a fast computation of the distance from a point to the curve.

We demonstrate the advantages of anisotropic convolutions in three applications, skeleton-based modeling, general implicit modeling, and shape approximation. For the first application we introduce a meshing technique based on a *scaffolding* method [19]. A *scaffold* is a coarse quad mesh that is built around a skeleton made of line segments. Here we extend the scaffolding method from line segments to $\mathscr{G}^1$ curves. We construct the mesh by first computing a refined scaffold that is then projected onto the surface. The projection step is done by ray shooting from the skeleton. This meshing technique can provide a coarse to fine quad mesh, that matches the topology of the skeleton and has good edge-flow. We aim to keep the number of evaluations of the convolution field reasonably low, at least comparable to Marching Cubes [31] and variants [21, 51].

Other than the simple and intuitive modeling of shapes, we also investigate the use of anisotropic convolution surfaces for shape approximation. Starting from a mesh of the shape and its skeletonization [30, 56], we propose a pipeline to best fit an anisotropic convolution surface. The new shape so obtained has a compact representation, and can be seen as a lossy compression of the original model. It is also smooth and can be obtained from occluded models (with holes). We use circular splines approximations to simplify the output of skeletonization algorithms (which usually requires a hefty post-processing [5]), and call on optimization to determine the parameters of the anisotropic convolution surface.

### 1.1 Related work

Several methods have been proposed for the generation of surfaces around 1D skeletons: sweep surfaces [40], offset surfaces [38], canal surfaces [37, 17], B-meshes [25], and convolution surfaces [8, 58] are some of them.

Convolution surfaces can be defined around any piecewise regular curve. To lower the complexity of the formulas of the integrals forming the convolution field one resorts to simpler curves, as line segments and arcs of circle [18, 22, 24, 28, 26, 32, 44, 58, 59, 61], or even quadratic curves [27]. More complex skeletal curves are approximated or warped [58]. Though line segments and arcs of circle support closed form formulas for convolution fields [24, 23, 27, 26, 44, 58], they can be daunting when varying the radius in a scale invariant way [18]. Further efforts in that direction are unproductive for the anisotropic formulation we present. Thus we resort to numerical integration [39], and this further motivates the development of a meshing technique requiring less evaluations of the the convolution field.

Weighted convolution and alternative formulations were introduced [26, 22, 59] to controllably vary the thickness along the skeleton, still with close-to-circular normal sections. Tai *et al.* [47] introduced general shaped normal sections with a modeling technique based on field remapping, at the cost of smoothness and complexity. Another approach for anisotropy is B-Meshes [25], where ellipse-like normal sections are achieved by interpolating ellipsoids along line segments. Nonetheless the orientation of the ellipsoid seems rigidly imposed by the system. The smoothness is handled on a post-processing step.

In [27] *planar* circular splines were first used for convolution surface skeletons. Our *spatial* circular spline approach takes advantage of biarcs theory [10]. We adapt some ideas on Rotation Minimizing Frames [10] for sweep surfaces [49] to the context of convolution surfaces. Biarcs-based circular spline ideas are also discussed in [33, 45]. To model organic shapes with few primitives [60] introduced helical primitives with warping, while our suggestion is to use approximation of helices with circular splines, removing the need for warping.

Motivated by the limitations of 1D skeleton, convolution surfaces around 2D skeletons have been developed [29, 23, 61, 43] for shape approximation and other applications. They are inherently more difficult to model and present more complex formulas. Our approach can be extended to 2D skeletons. Nonetheless the modeling capabilities of *anisotropic convolution* diminish the need of 2D skeletons.

Different scaffold constructions have been used in the literature [1, 4, 19, 25, 35, 48, 57]. The algorithm in [19] works for line segments skeletons of any topology and can respect their symmetries. We provide an adaptation of [19] for $\mathscr{G}^1$ curves.

### 1.2 General overview and contributions

*Anisotropic convolution* uses *frames* on the skeletal curves, which thus need to be $\mathscr{G}^1$ curves. We start by discussing *frames* for $\mathscr{G}^1$ curves, biarc theory, and circular splines in Section 2. An introduction to convolution surfaces, extensions, and our new formulation are discussed in Section 3. The meshing process that makes use of scaffolding is discussed in Section 4. In Section 5 we describe the numerical evaluation of the convolution fields. Section 6 describes three applications: skeleton-based modeling, general implicit modeling, and shape approximation.

Our contributions are: an extension to the modeling capabilities of convolution surfaces techniques; the introduction of a spatial circular spline skeletal description that allows for the definition of shapes in an organic way; the description of a meshing technique that uses the skeletal information coupled with a scaffold to improve the polygonization of the surfaces; and applications of anisotropic convolution to the modeling and approximation of a greater variety of shapes.

**Notation** Vectors are denoted by small bold letters and are assumed to be in column form. $\|\cdot\|$ denotes the Euclidean norm. Matrices are $3\times 3$ and represented by capital bold letters. Scalars are represented by non-bold letters. $\mathbf{I}$ denotes the identity matrix. $\mathbf{A}^i$ refers to the i-th column vector of the matrix $\mathbf{A}$. $\mathrm{SO}(3)$ and $Sym^+$, denote the group of orthonormal matrices, and the cone of positive-definite symmetric matrices, respectively. $\mathrm{Diag}(a,b,c)$ denotes the diagonal matrix with $a,b,c$ in the main diagonal. A piecewise-regular curve $\Gamma$ is said to be $\mathscr{G}^1$ if it has continuous unit tangent. We assume all curves to be parametrized by arc-length.

## 2 Preliminaries: circular splines and frames

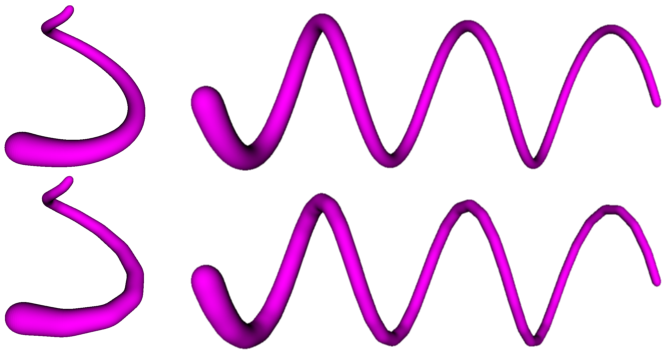
In this section we discuss preliminary concepts that are going to be used later in both theoretical and practical developments. Following [49], we briefly present the main components of *circular splines* approximation, and key properties of *frames* in the context of convolution.

### 2.1 Circular splines: approximation of general curves

Line segments may require many pieces to obtain a good approximation and a visually appealing convolution surface. Furthermore, a polyline approximation of a curve is continuous *at most*. Using arcs of circle and line segments one can get a $\mathscr{G}^1$ approximation of a skeletal curve in the form of circular splines [49]. Figure 1 compares the convolution of polylines and circular splines. We use biarc theory to construct a circular spline approximation from *Hermite data*.

Let $\mathscr{H} = (A_0, \mathbf{t}_0, A_1, \mathbf{t}_1)$ be the *Hermite data* defined by a pair of points $A_0, A_1 \in \mathbb{R}^3$, $A_0 \neq A_1$, along with their associated unit tangent vectors $\mathbf{t}_0, \mathbf{t}_1$. An *interpolating biarc* [10, 33, 49] of $\mathscr{H}$ is then defined as a $\mathscr{G}^1$ curve joining $A_0$ and $A_1$ formed by two arcs of circle such that one arc is passing through $A_0$ with unit tangent $\mathbf{t}_0$, and the other is passing through $A_1$ with unit tangent $\mathbf{t}_1$ (see Figure 2). Biarcs are simple but powerful enough for modeling spatial curves [45]. They allow interpolation of a set of Hermite data with a $\mathscr{G}^1$ curve, have an analytical formula for arc-length, and the distances from points in space are easy to compute [45].

On each Hermite data, the fitting biarcs form a one-parameter family [45]. There are several criteria [34, Chapter 3] to choose a particular biarc. In this work we use the biarc that gives equal tangent length for its two arcs ($l_0 = l_1$ in Figure 2). Our choice follows [45] where a circular spline approximation is computed from sampling points. We reuse the equal tangent property when computing a circular spline approximation of a polyline in Section 6 (Figure 21b).



(a) Top: 12 arcs of circle; bottom: 14 line segments.

(b) Top: 42 arcs of circle; bottom: 42 line segments.

Figure 1: Convolution surfaces around $\mathscr{G}^1$ circular spline (top), and polyline (bottom), approximations of (a) the spiral $(\frac{1}{2}t\cos t, \frac{3}{4}t\sin t, \frac{4}{5}t)$, $t \in [0, 2\pi]$, and (b) the elliptical helix $(2\cos t, 3\sin t, t)$, $t \in [0, 6\pi]$. All surfaces are smooth but the surfaces around a circular spline approximation are "visually" smoother.

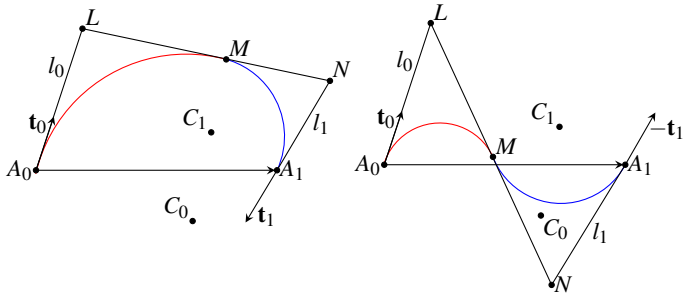



Figure 2: The biarc construction for $\mathscr{H} = (A_0, \mathbf{t}_0, A_1, \mathbf{t}_1)$, on the left; and $\mathscr{H} = (A_0, \mathbf{t}_0, A_1, -\mathbf{t}_1)$, on the right. Notice that each biarc has a natural tangential polyline: $A_0LMNA_1$.

Given the Hermite data set $\{\mathscr{H}_i = (A_i, \mathbf{t}_i, A_{i+1}, \mathbf{t}_{i+1})\}_{i=0}^{n-1}$, we can now construct an interpolating circular spline [45]. For each $\mathscr{H}_i$ we take the corresponding biarc, except when $\mathbf{t}_0 = \mathbf{t}_1 = A_1 - A_0$, where a line segment is used instead. The $\mathscr{G}^1$ curve piecewise defined by the biarcs and line segments is the interpolating *circular spline*. For a general $\mathscr{G}^1$ curve, a sampling is used to obtain the Hermite data set. The number of samples can be increased to get a more accurate approximation [45] (Figure 3). In the rest of the paper we assume the skeletal curves to be circular splines.

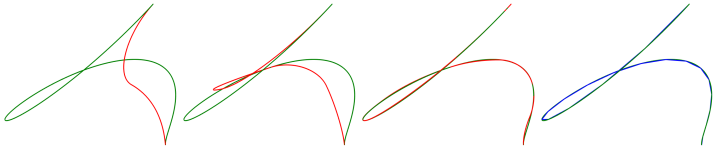


Figure 3: Biarcs interpolation (red) of a curve (green) with (from left to right) 2, 3, and 5 sampling points (2, 4, 8 arcs). The last picture on the right shows a polyline interpolation with 20 sampling points.

## 2.2 Frames for $\mathscr{G}^1$ curves

A *frame* is an orthonormal set of three vectors defined along a regular curve such that the first one coincides with the unit tangent. Formally, for a regular curve $\Gamma : [0, l] \to \mathbb{R}^3$, a *frame* is a continuous map $\mathbf{F} : [0, l] \to \mathrm{SO}(3)$ such that:

$$\mathbf{F}^1(s) = \Gamma'(s)^\mathsf{T} \quad \forall s \in [0, l]. \tag{1}$$

Framing a curve is a well-studied problem [7, 49] with the most common frame being the *Frenet* frame for $\mathscr{G}^2$ curves. Of particular interest for us are the *Rotation Minimizing Frames* (RMF) that can be defined for $\mathscr{G}^1$ curves and minimize the rotation around the tangent direction.

When $\Gamma$ is a $\mathscr{G}^1$ curve, piecewise composed of regular curves $\Gamma_h : [0, l_h] \to \mathbb{R}^3$ $(h = 1, 2, \ldots, d)$, we can define a frame $\mathbf{F}$ for $\Gamma$ by means of the frames $\mathbf{F}_h$ of $\Gamma_h$. Let $L_h = l_1 + l_2 + \cdots + l_h$ and $L_0 = 0$, then $\Gamma : [0, L_d] \to \mathbb{R}^3$ is parameterized as

$$\Gamma(s) = \Big\{ \Gamma_h(s) \quad L_{h-1} \le s < L_h \quad (h = 1, 2, \ldots, d)$$

and $\mathbf{F}$ is defined as

$$\mathbf{F}(s) = \Big\{ \tilde{\mathbf{F}}_h(s - L_{h-1}) \quad L_{h-1} \le s < L_h \quad (h = 1, 2, \ldots, d) \tag{2}$$

where $\tilde{\mathbf{F}}_h(s) = \mathbf{F}_h(s)\mathbf{R}_h$, with $\mathbf{R}_0 = \mathbf{I}$ and $\mathbf{R}_h \in \mathrm{SO}(3)$ is the rotation matrix that takes $\mathbf{F}_h(0)$ to $\tilde{\mathbf{F}}_{h-1}(l_{h-1})$ by rotating around $\mathbf{F}^1_h(0) = \mathbf{F}^1_{h-1}(l_{h-1})$, that is $\tilde{\mathbf{F}}_{h-1}(l_{h-1}) = \mathbf{F}_h(0)\mathbf{R}_h$ (see Figure 4).

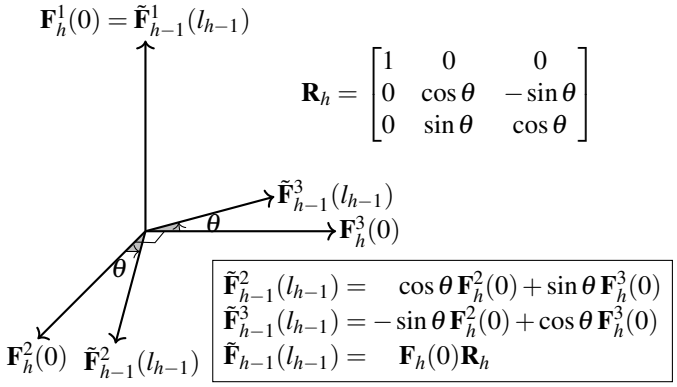



Figure 4: Rotation of $\mathbf{F}_h(0)$ into $\tilde{\mathbf{F}}_{h-1}(l_{h-1})$. Since $\Gamma$ is a $\mathscr{G}^1$ curve, the unit tangent vectors at $\Gamma_h(l_h)$ and $\Gamma_{h+1}(0)$ coincide, thus $\mathbf{F}^1_h(l_h) = \mathbf{F}^1_{h+1}(0)$ $(h = 1, 2, \ldots, d-1)$, and $\mathbf{F}_{h+1}(0)$ differs from $\mathbf{F}_h(l_h)$ by a rotation $R_h$ around the axis $\mathbf{F}^1_{h+1}(0) = \mathbf{F}^1_h(l_h)$. The rotation is needed in order to keep a continuous frame along $\Gamma$.

When $\mathbf{F}_h$ is a RMF of $\Gamma_h$, the frame defined in (2) is a RMF for the whole curve $\Gamma$ [7, 49]. For arcs of circles, the Frenet frame is a RMF. It consists of the unit tangent, a unit radial vector and a normal vector to the plane where the arc sits. Note that a RMF may not be continuous for closed curves. A corrective rotation is to be added along the curve to obtain a continuous frame [49]. We later show how this is handled in a natural way in anisotropic convolution ($\theta_1 \neq \theta_0$ in Equation (11)).

# 3 Convolution surfaces: from varying thickness to anisotropy

Generally speaking a *convolution surface* in $\mathbb{R}^3$ is the level set of a *convolution* field that results from the integration of a *kernel* function $K$ along a *skeleton* $\mathscr{E}$. A *kernel* is a function $K : \mathbb{R}^+ \to \mathbb{R}^+$ which is at least $\mathscr{C}^1$, and is strictly decreasing when nonzero.

In this paper the skeletons are collections of regular curves with finite arc-length that intersect only at endpoints. For a skeletal curve $\Gamma : [0,l] \to \mathbb{R}^3$, the *convolution field* $C^K_\Gamma$ is defined as

$$C^K_\Gamma(P) = \int_0^l K(\|P - \Gamma(s)\|)\,\mathrm{d}s \quad \text{for all } P \in \mathbb{R}^3. \tag{3}$$

For the skeleton $\mathscr{E}$, the *convolution field* $C_\mathscr{E}$ is defined as

$$C_\mathscr{E} = \sum_{\Gamma \in \mathscr{E}} \delta_\Gamma C^K_\Gamma(P), \tag{4}$$

with $\delta_\Gamma \in \mathbb{R}$ for all $\Gamma \in \mathscr{E}$. In the simplest case $\delta_\Gamma = 1$ for all $\Gamma \in \mathscr{E}$. When $\delta_\Gamma < 0$ a *soft carving* effect is achieved (like in [47], see Figure 10c for an example). Equation (3) can be written as a line integral, therefore the convolution field (4) is independent of the parameterization and skeleton subdivisions. Leibniz integral rule for differentiation under the integral symbol [16] guarantees that $C^K_\Gamma$ is also as smooth as $K$ for points outside of the skeleton (i.e. for $P \notin \Gamma([0,l])$).

The *convolution surface* $\mathscr{S}^c_\mathscr{E}$, for the level value $c > 0$, is formally defined as

$$\mathscr{S}^c_\mathscr{E} = \left\{ P \in \mathbb{R}^3 \;\middle|\; \mathscr{C}^K_\mathscr{E}(P) = c \right\}. \tag{5}$$

It is closed (in a topological sense) and smooth, provided $c$ is not a critical value of $C^K_\mathscr{E}$ [13, Section 2-2]. For simplicity if $\mathscr{E}$ has only one skeletal curve $\Gamma$ we identify $\mathscr{E}$ with $\Gamma$.

Among the kernel functions proposed in the literature [23, 24, 44, 43, 59, 55, 58] , we choose the compactly supported polynomial kernel:

$$K(x) = \begin{cases} \frac{35}{16}\left(1 - x^2\right)^3 & 0 \le x \le 1 \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

The integral of the kernel over its support is 1, that is $\int_0^1 K(x)\mathrm{d}x = 1$. Compactly supported polynomial kernels are advantageous because changes on the skeleton affect only locally the convolution surface while the kernel is still a simple function (piecewise polynomial). A similar type of kernel was recently used in the Brush2Model tool [63].

## 3.1 Varying thickness

Despite the freedom in the choice of the kernel and level set, the thickness around the skeleton in a convolution surface do not change very much along the supporting skeletal curve. A convolution surface around a line segment has a tubular structure with circular normal sections.

In order to control the thickness several alternatives have been introduced. One of the first ideas was to use a *weight* function. Polynomial weight functions were used in [24, 26, 27, 28]. Other alternatives, proposed by Hornus *et al.* [22], and SCALIS proposed by Zanni *et al.* [59], modify the distance from the point to the curve before evaluating the kernel, the difference is in the normalization factor introduced in SCALIS. A summary of the convolution surface variants is shown in Table 1. All variants have circular normal sections. Within this limitation, SCALIS offers a better blending behavior and control over small details on which we build.

| Variant | Formulation |
|---|---|
| WEIGHTED [28, 24] | $J^K_{\Gamma,w}(P) = \int_0^l w(s) K(\|P - \Gamma(s)\|)\,\mathrm{d}s$ <br> $w : [0,l] \to \mathbb{R}$ *weight function* |
| HORNUS [22] | $H^K_{\Gamma,\rho}(P) = \int_0^l K\left(\frac{\|P - \Gamma(s)\|}{\rho(s)}\right)\mathrm{d}s$ <br> $\rho : [0,l] \to \mathbb{R}^+$ *radius function* |
| SCALIS [59] | $Z^K_{\Gamma,\lambda}(P) = \int_0^l K\left(\frac{\|P - \Gamma(s)\|}{\lambda(s)}\right)\frac{\mathrm{d}s}{\lambda(s)}$ <br> $\lambda : [0,l] \to \mathbb{R}^+$ *scale function* |

Table 1: Convolution surface variants. Each alternative formulation uses a function that controllably varies the thickness along the skeleton.

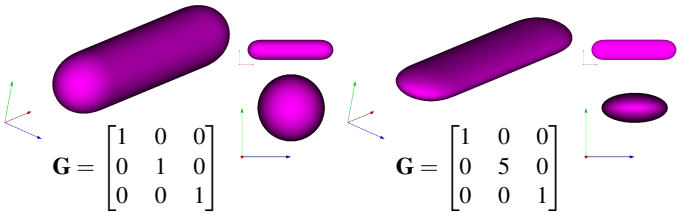
## 3.2 Anisotropic convolution

In order to increase modeling freedom we introduce anisotropy in the convolution surfaces. The idea is to change the way the distance to a point in the skeleton is computed before the kernel evaluation. Instead of using the standard Euclidean (isotropic) distance, we introduce an anisotropic distance. This is achieved with a scalar product defined by a *metric matrix*, related to a frame of the skeletal curve.

A matrix $\mathbf{G} \in Sym^+$ defines a scalar product $\langle \cdot, \cdot \rangle$ in $\mathbb{R}^3$ as $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^\mathsf{T}\mathbf{G}\mathbf{y}$ for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^3$. Thus $\mathbf{x} \mapsto \sqrt{\mathbf{x}^\mathsf{T}\mathbf{G}\mathbf{x}}$ defines a norm (and hence a distance) in $\mathbb{R}^3$. We refer to $\mathbf{G}$ as a *metric matrix*. The *anisotropic convolution field* is then defined as

$$C^K_{\Gamma,\mathbf{G}}(P) = \int_0^l K \circ g(\Gamma(s), P - \Gamma(s)) \cdot g(\Gamma(s), \Gamma'(s))\mathrm{d}s, \tag{7}$$

where $g(\mathbf{y}, \mathbf{x}) = \sqrt{\mathbf{x}^\mathsf{T} \cdot \mathbf{G}(\mathbf{y}) \cdot \mathbf{x}}$ for all $(\mathbf{y}, \mathbf{x}) \in \Gamma([0,l]) \times \mathbb{R}^3$, and $\mathbf{G} : \mathscr{E} \to Sym^+$ is a map that assigns a metric matrix to each point in the skeleton. In practice $\mathbf{G}$ is seen as a map from $[0,l]$ to $Sym^+$. If $\mathbf{G}(s) = \mathbf{I}$, then (7) reduces to (3). If $\mathbf{G}(s) = \lambda(s)^{-2}\mathbf{I}$, then (7) reduces to SCALIS formulation (Table 1). Other choices change the anisotropy of the surface. The normal sections are ellipses, and we can, for example, obtain a flattened volume (Figure 5). To vary the thickness we vary the metric matrix along the skeletal curve $\Gamma$.




(a) Isotropic metric. (b) Anisotropic metric.

Figure 5: Constant metric matrices along a line segment.

*Anisotropic convolution* has the same *scale invariant* behavior of SCALIS [59], that we state in the following proposition (proof in supplementary material).

**Proposition 1** (Scale invariance)**.** *For any scale* $\tau > 0$, $C^K_{\Gamma,\mathbf{G}}(X) = C^K_{\tau\Gamma,\tau\mathbf{G}}(\tau X)$ *for all* $X \in \mathbb{R}^3$, *with* $(\tau\Gamma)(s) = \tau\Gamma(\frac{s}{\tau})$ *and* $(\tau\mathbf{G})(s) = \frac{1}{\tau^2}\mathbf{G}(\frac{s}{\tau})$ *for* $s \in [0, \tau l]$.

We exploit the eigen-decomposition [2] of $\mathbf{G}$ to achieve control over the shape along $\Gamma$. Any matrix $\mathbf{G} \in Sym^+$ can be written as $\mathbf{G} = \mathbf{U}\mathbf{D}\mathbf{U}^\mathsf{T}$ where $\mathbf{U} \in \mathrm{SO}(3)$ has the eigen-vectors of $\mathbf{G}$ as columns, and $\mathbf{D}$ is a diagonal matrix with the eigen-values of $\mathbf{G}$ in the main diagonal. A visual representation is shown in Figure 6.

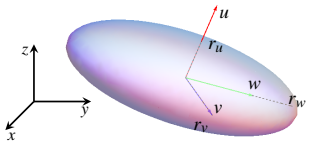



Figure 6: $\mathbf{G} = \mathbf{U}\mathbf{D}\mathbf{U}^\mathsf{T}$ as the ellipsoid $\mathbf{x}^\mathsf{T}\mathbf{G}\mathbf{x} = 1$. The eigenvectors $\mathbf{u}, \mathbf{v}, \mathbf{w}$ define the axes, the radii are given by the eigenvalues: $r_u = \alpha^{-\frac{1}{2}}, r_v = \beta^{-\frac{1}{2}}, r_w = \gamma^{-\frac{1}{2}}$.

Starting with a set of three orthonormal vector fields $\mathbf{u}, \mathbf{v}, \mathbf{w} : [0, l] \to \mathbb{R}^3$ and three positive functions $\alpha, \beta, \gamma : [0, l] \to \mathbb{R}^*_+$ we can define a metric matrix

$$\begin{aligned} &\mathbf{G}(s) = \mathbf{U}(s)\mathbf{D}(s)\left(\mathbf{U}(s)^\mathsf{T}\right) \quad \text{for } s \in [0, l], \text{ with} \\ &\mathbf{U}(s) = [\mathbf{u}(s)\ \mathbf{v}(s)\ \mathbf{w}(s)] \text{ and} \\ &\mathbf{D}(s) = \mathrm{Diag}(\alpha(s), \beta(s), \gamma(s)). \end{aligned} \tag{8}$$

Equation (8) decouples the magnitude ($\mathbf{D}$) and orientation ($\mathbf{U}$) of $\mathbf{G}$. Given a desired final shape, we construct a suitable map $\mathbf{G} : [0, l] \to Sym^+$ that controls the shape along $\Gamma$. To achieve this we take $\mathbf{U}$ to be a frame of $\Gamma$ (Section 2.2) and so we get a metric matrix that "follows" the skeletal curve. With this choice, $\mathbf{U}^1(s) = \Gamma'(s)$, and we get that (7) simplifies to

$$C^K_{\Gamma,\mathbf{G}}(P) = \int_0^l K \circ g(\Gamma(s), P - \Gamma(s))\sqrt{\alpha(s)}\mathrm{d}s. \tag{9}$$

The right hand side of (9) can be written as the line integral $\int_\Gamma K \circ g(\mathbf{y}, P - \mathbf{y})\sqrt{\alpha(\mathbf{y})}\mathrm{d}\mathbf{y}$, and we can deduce the same advantages as in (3). We discuss next how to define $\mathbf{U}(s)$ and $\mathbf{D}(s)$ given some design parameters.

### 3.2.1 Modeling with anisotropic convolution

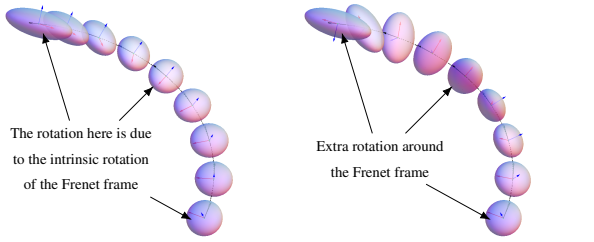
Anisotropic convolution allows for an intuitive modeling framework. First, to effectively describe the shape along the skeletal curve $\Gamma([0, l])$, we restrict $\mathbf{U}$ to be a frame of $\Gamma$. In practice $\mathbf{U}$ is defined as a rotation of an automatically computed frame $\mathbf{F}$ of $\Gamma$ (a RMF, or Frenet frame). At each extremity of $\Gamma$ the user choses the rotation and radii. Then the parameters (the rotation and radii) defining the matrix G are linearly interpolated along $\Gamma$. We model the surfaces as to not intersect the skeleton. For long enough skeletal curves, we have $C^K_{\Gamma,\mathbf{G}}(P) \in [1, 2]$ for all points $P$ in the skeleton, i.e. $P \in \Gamma([0, l])$. It follows that the level value $c$ must be in the interval $(0, 1)$ such that the whole skeleton is contained inside the surface.

We denote $\theta_i \in \mathbb{R}$ $(i = 0, 1)$ the angles associated to the initial $(i = 0)$ and final $(i = 1)$ endpoints of $\Gamma$ respectively. Then the frame $\mathbf{U}$ is defined as

$$\mathbf{U}(s) = \mathbf{F}(s)\mathbf{R}(s), \text{ with} \tag{10}$$

$$\mathbf{R}(s) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta(s) & -\sin\theta(s) \\ 0 & \sin\theta(s) & \cos\theta(s) \end{bmatrix}, \ \theta(s) = \frac{l - s}{l}\theta_0 + \frac{s}{l}\theta_1. \tag{11}$$

From user inputs, we determine initial and final eigenvalues $\alpha_i, \beta_i, \gamma_i \in R^+ (i = 0, 1)$ which are interpolated in order to define the positive eigen-values of $D(s)$ along the skeleton: $(\alpha(s), \beta(s), \gamma(s))$. We discuss this in Section 3.2.2.




(a) Anisotropy: $\alpha_0 = \alpha_1 = 1, \beta_0 = 1, \beta_1 = \frac{1}{4}, \gamma_0 = \gamma_1 = 2$. No rotation.

(b) Anisotropy: $\alpha_0 = \alpha_1 = 1$, $\beta_0 = 1, \beta_1 = \frac{1}{4}$, $\gamma_0 = \gamma_1 = 2$, $\theta_0 = 0, \theta_1 = \pi$.

Figure 7: Interpolation of the metric matrix $\mathbf{G}$ along the helix $\Gamma(t) = (5\cos t, 5\sin t, 3t)$, $t \in [0, \pi]$ with respect to the Frenet frame of $\Gamma$.

The eight parameters $(\theta_i, \alpha_i, \beta_i, \gamma_i)$ $(i = 0, 1)$ completely define $\mathbf{G} : [0, l] \to Sym^+$, and control the shape along the skeletal curve. Figure 7 illustrates the evolution of the parameters in (10) and (12) along the skeleton. $\beta$ and $\gamma$ describe the shape in the two normal directions of the frame along the curve, while $\alpha$ controls the distance from the tips of the surface to the extremities of the skeleton. Twisting $(\theta)$ and varying thickness $(\beta, \gamma)$ can be combined as shown in Figure 8. Varying $\alpha$ can be used to control the "bumping" in the blending area between two pieces, as shown in Figure 9.

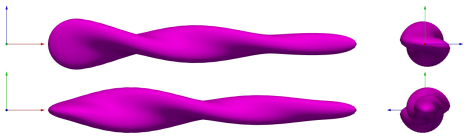
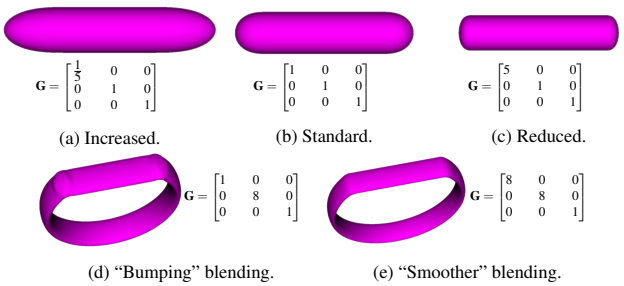


Figure 8: Anisotropic convolution around a segment with a combination of twisting and varying thickness: $\alpha_0 = \alpha_1 = 1, \beta_0 = \beta_1 = 10, \gamma_0 = 1, \gamma_1 = 6, \theta_0 = 0, \theta_1 = 2\pi$.

When $\mathbf{F}$ is a RMF of $\Gamma$, the frame $\mathbf{U}$ in (8) rotates minimally around $\Gamma$ outside the linearly prescribed rotation given by $\theta_0$ and $\theta_1$. This reflects the non-intrinsic user-defined rotation of the metric matrix so as to model some twisting (figures 8, 10b, 16 and 19), or to adjust a frame for continuity along a closed curve [49, 50] (Figure 18).

With anisotropic convolution one can mimic the effects of 2D skeletons. Figure 10a shows some shapes previously modeled in [61] with 2D polygonal skeletons. We can also obtain a variety of centrally symmetric normal sections by reusing the same skeleton with several metrics. Figure 10b shows cross-shaped normal sections. In Figure 10c we illustrate soft carving.

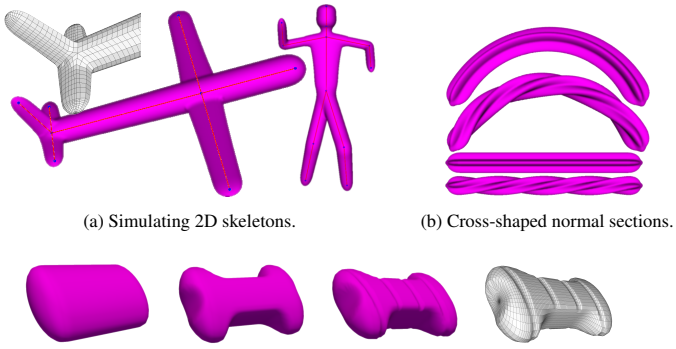
Since anisotropic convolution is scale-invariant (Proposition 1) we inherit the advantages of SCALIS [59] such as modeling at




(a) Increased. (b) Standard. (c) Reduced.

(d) "Bumping" blending. (e) "Smoother" blending.

Figure 9: Anisotropic convolution can be used to change the extremities (tips) of the surfaces (top row). An application to reducing "bumping" in the blending area is shown in (d-e).



(a) Simulating 2D skeletons. (b) Cross-shaped normal sections.

(c) Soft carving effect by subtracting anisotropic fields.

Figure 10: Extended modeling capabilities of anisotropic convolution.

different scales. Compared to SCALIS we gain anisotropy and twisting behavior. We can also control the radius in the tangent direction, hence the distance from the surface to the extremities of the skeletal curve; this was done before by modifying the skeletal curve [59]. Radii control for anisotropic convolution is discussed in the next section.

### 3.2.2 Controlling the radii

The eigenvalues and rotation of the metric matrix define the general shape of the surface. To gain a finer control over the actual radii we analyze what happens for a particular level set in the ideal case of a line segment skeleton with a constant metric matrix. We discuss next how the eigenvalue parameters must be chosen in order to attain a desired radius.

Let $\Gamma(s) = Q + s\mathbf{u}$ be a line segment skeleton, with $s \in [0,l]$, $\mathbf{u} \in \mathbb{R}^3$ the unit tangent vector, and $Q \in \mathbb{R}^3$. Since the tangent is constant along a line, we can take a constant frame $\mathbf{U} = [\mathbf{u}\ \mathbf{v}\ \mathbf{w}]$. For the level value $c \in (0,1)$, let $\omega_c$ be the only solution of $\omega_c - \omega_c^3 + \frac{3}{5}\omega_c^5 - \frac{1}{7}\omega_c^7 = \frac{16}{35}(1-c)$ in $(0,1)$, and let $\eta_c = \sqrt{1-(\frac{c}{2})^{\frac{2}{7}}}$. The next proposition gives the values for the eigenvalues that achieve precise radii on each direction around $\Gamma$. (See supplementary material for proof and derivations of the formulas for $\omega_c$ and $\eta_c$).

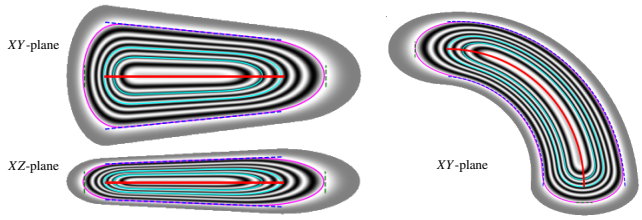
**Proposition 2** (Radii control)**.** *Given the radii $r_u, r_v, r_w$ such that $\frac{r_u}{\omega_c}(\frac{c}{2})^{\frac{1}{7}} \leq \frac{l}{2}$, let $\alpha = \frac{\omega_c^2}{r_u^2}$, $\beta = \frac{\eta_c^2}{r_v^2}$ and $\gamma = \frac{\eta_c^2}{r_w^2}$. Then $\{Q - r_u\mathbf{u}, Q + \hat{r}_u\mathbf{u} + r_v\mathbf{v}, Q + \hat{r}_u\mathbf{u} + r_w\mathbf{w}\} \subset \mathscr{S}^c_{\mathscr{E}}$ for $\hat{r}_u \in [\frac{r_u}{\omega_c}(\frac{c}{2})^{\frac{1}{7}}, l - \frac{r_u}{\omega_c}(\frac{c}{2})^{\frac{1}{7}}]$.*

From Proposition 2 we get several important conclusions. First, we can achieve precise control over the radius in the tangent direction (tip distances). Second, each eigenvalue is defined by $c$ (the level value) and the desired radius $(r_u, r_v, r_w)$, independently of the other two eigenvalues, i.e. changing one radius does not impact the others. Third, the radii in the normal directions are guaranteed in the interval $[\frac{r_u}{\omega_c}(\frac{c}{2})^{\frac{1}{7}}, l - \frac{r_u}{\omega_c}(\frac{c}{2})^{\frac{1}{7}}]$, depending only on the radius in the tangent direction and $c$. And last: by choosing a level set close to zero we can achieve the radii in the normal directions sufficiently close to the tips, within a valid interval along the curve, i.e. such that $\frac{r_u}{\omega_c}(\frac{c}{2})^{\frac{1}{7}} \leq \frac{l}{2}$. This follows from $\frac{1}{\omega_c}(\frac{c}{2})^{\frac{1}{7}}$ being an increasing function of $c$ with range $[0,+\infty]$ for $c \in [0,1]$. For $c = 0.1$, $\frac{1}{\omega_c}(\frac{c}{2})^{\frac{1}{7}} \approx 1.18654$. Note though, that choosing a particular level set $\tilde{c}$ for one skeletal curve $\Gamma \in \mathscr{E}$ is equivalent to change the constant $\delta_\Gamma$ to $\frac{c}{\tilde{c}}\delta_\Gamma$ in Equation (4). This way the overall convolution surface is still defined by the original level value $c$.

Thanks to Proposition 2, the user only has to define radii at the end-points of the skeletal curves, the eigen-values $\alpha_i, \beta_i, \gamma_i$ are then computed automatically. We perform eigen-value interpolation along the skeletal curves such that the radii of the metric matrix ellipsoid vary linearly. This is done using the following formula:

$$\chi(s) = \left(\frac{l-s}{l}\chi_0^{-\frac{1}{2}} + \frac{s}{l}\chi_1^{-\frac{1}{2}}\right)^{-2} \quad \text{for } \chi = \alpha, \beta, \gamma. \tag{12}$$

Figure 11 shows examples of radii control along the shape. We used Proposition 2 to compute the associated eigen-values given some desired radii at each extremity of $\Gamma$. Notice that, although Proposition 2 assumes $\Gamma$ to be a line segment and the radii to be constant, the eigenvalues computed with the same method works well for varying radii around both line segments and arcs of circle.

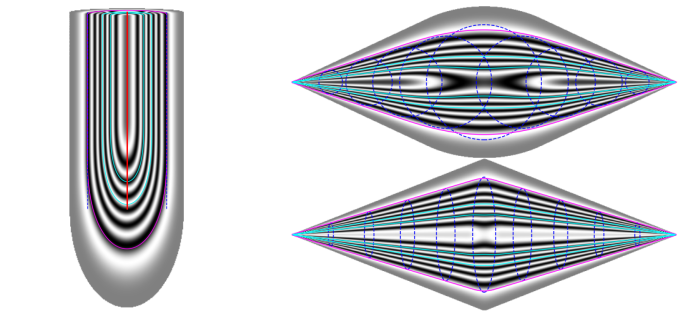



(a) Line segment with varying radii. (b) Arc of circle with varying radii.

Figure 11: Radii control in anisotropic convolution. We show the level sets for $c = 0.1$ (magenta), $c = 1.0$ and $c = 1.5$ (cyan). The blue dashed line shows the linear interpolation of the user-defined radii: from 1.5 to 1.0 in the $XY$-plane, from 0.7 to 0.5 in the $XZ$-plane. The green dashed line shows the expected radii in the tangent direction at extremities: from 0.6 to 1.2. Notice how the surface level set ($c = 0.1$) is very close to the expected radii interpolation even in the case of arcs of circle. In gray-scale we show the values of the convolution field composed with $x \to \sin(20x)$. The picture illustrates the boundary outside which the field value is zero. In red we show the skeletal curve.

As already illustrated in Figure 11, anisotropic convolution provides not only an ellipse-like normal section but also caps of independent depth at the extremities. Such a desirable feature would require tweaking the skeleton tips in [59]. If as in previous convolution formalism all radii are equal ($\alpha = \beta = \gamma$), the convolution can be seen as smoothing a union of sphere centered on the

skeleton. On the other end, if $\alpha \ll \beta, \gamma$, anisotropic convolution can be seen as smoothing of a union of discs normal to the skeleton (see Figure 12b). Even for circular normal sections ($\beta = \gamma$), this property allows to model shape features not amenable to previous convolution formalisms. The shape can easily be tapered at the extremities (Figure 12a) and its cross-section can vary more rapidly along the skeleton.



(a) Varying $\alpha$ with $\beta, \gamma$ constant. (b) Varying $\alpha, \beta, \gamma$.

Figure 12: Impact of tangential radii on the surface. Variation of $\alpha$ is only visible at endpoints (left). If the tangential radii is equal to orthogonal radii then the shape is similar to a union of sphere (right top) else if $\alpha$ tend toward zero, the shape is similar to a union of discs (right bottom).

# 4 Skeleton-driven meshing

In this section we introduce a meshing technique for convolution surfaces that exploits the skeleton structure. Our approach produces quad-dominant meshes that follow the skeleton, providing a good edge-flow, whether coarse or refined. We first compute a *scaffold* [35, 19] – a coarse quad mesh around the skeleton – adapted to $\mathscr{G}^1$ curves, that is then projected onto the surface. The mesh so obtained reflects the smoothness of the surface without post-processing. With this approach the evaluation of the convolution field is delayed until the projection step, which is done only once for each point in the scaffold. Our meshing technique is intended for surfaces that reflect the topology, and to some extent the geometry, of the skeleton.

Common choices for polygonization of implicit surfaces, Marching Cubes [31] and variants [21, Chapter 7], produce triangle meshes that may misrepresent the topology of the surface for coarse grids. We produce quad meshes with accurate topology and good edge-flow, independently of the coarseness of the mesh. With quads we can exploit the curvature behavior along the skeleton, having longer side-length along the skeleton (where the curvature is lower) than transversally (higher curvature). Our projection step also guarantees that the mesh has all its vertices *on* the surface.

In [19] a *scaffold* is a quad mesh built around an articulated skeleton made of line segments. Among other scaffolds-like constructions [35, 25, 4, 48, 57], [19] works for skeletons of any topology, including skeletons with cycles like in Figure 16, without introducing extra quads at the joints. It also provides scaffolds with the same number of quads around each line segment, and scaffolds respecting the symmetries of the skeleton. [19] constructs polygonal "tubes" around each line segment that meet at the joints. The "tubes" intersect the unit sphere centered at the joint creating a partition into convex spherical polygons. The number of vertices and edges on the spherical polygons depend on the topology and the geometry of the whole skeleton. The spherical polygons define a set of unit vectors at each extremity of the line segments that are *connected* by the quads (Figure 13).

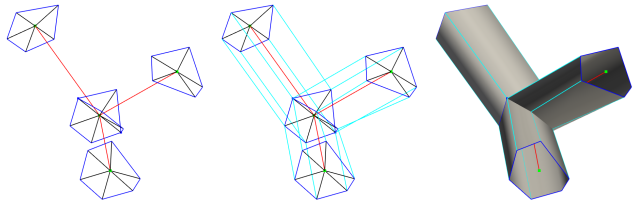


Figure 13: The scaffolding method constructs a quad mesh (right) from a line segment skeleton (red lines). This is done by constructing unit vectors (black lines) around each line segment extremities (green) that are then connected (cyan lines) in a bijective way. The vectors connected along each line segment define a polygon (blue) that encloses the corresponding segment at each extremity. Notice that some vectors (like in the middle joint) are connected along two or more line segments.

## 4.1 Scaffolds for circular splines

The scaffold in [19] is limited to line segment skeletons. To use it on circular spline skeletons we first take, for each skeletal curve, the polyline representation given by the collection of biarc tangents (Figure 2). We then modify the *connections* in the output of [19]: we drop the intermediate vectors of the polyline and connect directly the vectors at the endpoints of the skeletal curves (Figure 14). The scaffold is constructed around the topologically relevant skeletal curves. For example, in *soft carving* the pieces of the skeleton with $\delta_\Gamma < 0$ only provide details on surface and are not to support any pieces of the scaffold.

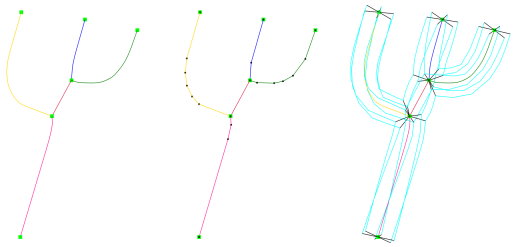


Figure 14: Meshing with a scaffold. From left to right: circular spline skeleton, tangential polyline discretization of the skeletal curves, and connection (cyan) of the vectors (black) along the curves. Notice that we *connect* directly the vectors at the endpoints of each curve

For a skeletal curve $\Gamma([0,l]) \in \mathscr{E}$, with frame $\mathbf{F}([0,l])$: let $m$ be the number of quads we want along $\Gamma$; let $\{\mathbf{f}_0^j\}_{j=0}^k$ and $\{\mathbf{f}_m^j\}_{j=0}^k$, be the sets of $k+1$ vectors computed by the scaffolding algorithm at $\Gamma(0)$ and $\Gamma(l)$ respectively. The indices $j$ represent the counterclockwise order, according to the frames $\mathbf{F}(0)$ and $\mathbf{F}(l)$ respectively, on the polygon generated around the extremities of $\Gamma$.

We connect the vectors by minimizing the rotation within the

frame $\mathbf{F}$ along $\Gamma$. That is, we connect $\mathbf{f}_0^j$ with $\mathbf{f}_m^{j+q}$ where

$$q = \underset{w=0\ldots k-1}{\operatorname{argmin}} \sum_{j=0}^{k} \|(\mathbf{F}(0))^\mathsf{T}\mathbf{f}_0^j - (\mathbf{F}(l))^\mathsf{T}\mathbf{f}_m^{k-j+w}\| \tag{13}$$

and $k-j+w$ is taken modulo $k$. For simplicity we relabel the vectors such that $\mathbf{f}_0^j$ is connected with $\mathbf{f}_m^j$.

To generate the quads along the skeletal curve we *transport* the vectors with varying local coordinates. Formally, given two unit vectors $\mathbf{f}_0, \mathbf{f}_m$ positioned at $\Gamma(0)$ and $\Gamma(l)$ respectively, we define the *transport* of $\mathbf{f}_0$ to $\mathbf{f}_m$ as

$$\mathbf{f}(t) = \mathbf{F}(t)\frac{\bar{\mathbf{f}}(t)}{\|\bar{\mathbf{f}}(t)\|}, \tag{14}$$

where $\bar{\mathbf{f}}$ is an interpolation from $\mathbf{F}(0)^\mathsf{T}\mathbf{f}_0$ to $\mathbf{F}(l)^\mathsf{T}\mathbf{f}_m$ that avoids the origin. In our setting we use the linear interpolation $\bar{\mathbf{f}}(t) = \frac{l-t}{l}\mathbf{F}(0)^\mathsf{T}\mathbf{f}_0 + \frac{t}{l}\mathbf{F}(l)^\mathsf{T}\mathbf{f}_m$ (Figure 15), valid when $\mathbf{F}(0)^\mathsf{T}\mathbf{f}_0 + \mathbf{F}(l)^\mathsf{T}\mathbf{f}_m \neq 0$.

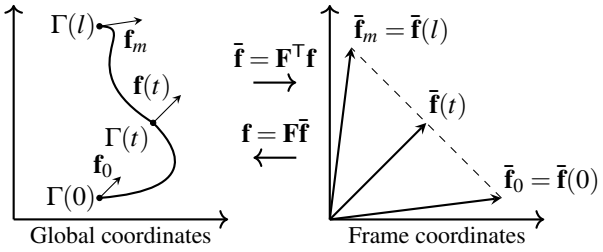



Figure 15: Transporting $\mathbf{f}_0$ to $\mathbf{f}_m$ along the curve $\Gamma$ with frame $\mathbf{F}$.

We then compute the *mesh lines* $\{\mathbf{f}_i^j\}_{i=0}^m$ $(j = 0,\ldots,k)$ where $\mathbf{f}_i^j = \mathbf{f}^j\left(\frac{i}{m}l\right)$ with $\mathbf{f}^j$ defined by the *transport* of $\mathbf{f}_0^j$ to $\mathbf{f}_m^j$ (14). The vertices $P_i^j = \Gamma\left(\frac{i}{m}l\right) + \mathbf{f}_i^j$ then define the quads

$$\mathscr{Q}_i^j = (P_i^j, P_{i+1}^j, P_{i+1}^{j+1}, P_i^{j+1}) \quad i = 0,1,\ldots,m-1 \quad j = 0,1,\ldots,k,$$

where the indices are considered modulo $k$.

To close the mesh at the tips one can choose to use polar-annular caps [3], which includes degenerate quads around a high valency irregular vertex. Or quad layout that includes several irregular vertices but with lower valency [53, Section 5.3]. In practice we use polar-annular caps.

### 4.2 Mesh projection onto the convolution surface

To obtain the final polygonization we project the mesh described in the previous section. The projection is done by ray-shooting from the skeleton onto the surface.

Given a point $T \in \Gamma([0,l])$, and a direction $\mathbf{u}$ with $\|\mathbf{u}\| = 1$, we find the first intersection of the ray $r(t) = T + t\mathbf{u}$ with the convolution surface around $\Gamma$. This means finding the minimal solution $t > 0$ of

$$C^K_{\Gamma,\mathbf{G}}(T + t\mathbf{u}) = c. \tag{15}$$

By definition, the point $P = T + t\mathbf{u}$ is on the convolution surface of the level set $c > 0$, for any $t > 0$ satisfying (15). We can exploit the decreasing nature of the field value when moving away from the skeleton to efficiently compute a solution of (15). Moreover, for compact support kernels the solutions can be bounded in an interval computed from the radius of the kernel. In our implementation we solve (15) with Brent's method [11] (available in GSL).

The quality of the obtained quad mesh can be appreciated in the wire-frames in Figures 10a, 10c, 16 and 19. In Figure 16 we show high-genus surface meshes.

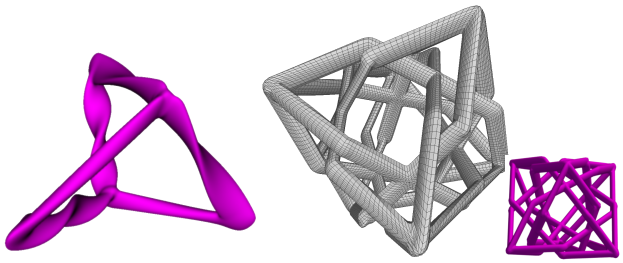


Figure 16: Abstract cage-like surfaces of positive genus (skeletons with cycles).

## 5 Implementation and Numerical integration

The meshing algorithm was prototyped in Python with call to a C library for field evaluation. Since the anisotropic convolution field requires a frame at each point of the skeletal curve, obtaining closed form formulas for anisotropic convolution (Equation (7)) is unrealistic. On the other hand quadrature methods for numerical integration provide accurate results up to a prescribed tolerance [39]. In our implementation we use the QUADPACK family of quadrature methods of the GNU Scientific Library (GSL) [20]. Each integral is evaluated using the *QAG adaptive integration method* [20] with 61 Gauss-Kronrod quadrature points, limited to a maximum of 100 subintervals, and absolute error of $10^{-8}$. When the radii are close to zero at one extremity, like in Figure 12b, the numerical accuracy of the field evaluation at a point $X \in \mathbb{R}^3$ is improved by placing more quadrature points in the vicinity of the closest point to $X$ on the skeleton. This can be effectively achieved by splitting the integral evaluation at the closest point of the skeleton into two integrals.

On all examples, timings are driven by the ray-shooting phase of the algorithm which is proportional to the number of vertices in the resulting mesh. For instance, in Figure 1a, the meshing time is 28.3s for the surface around the arcs of circle approximation, and 30.2s around line segments (using 4 core of an Intel Core i7-6600U CPU @ 2.60GHz). The total number of vertices in the mesh is 1160. The number of integral evaluations is more than 250k in both cases, with arcs of circle having in average less field evaluations per ray-shooting (22.75) than line segments (23.7). The average integral evaluation time is 0.34ms for an arc of circle, and 0.33ms for a line segment. Those numbers highlight two facts. First, it confirms that it is computationally more interesting to use circles for curve approximation. Secondly, an optimized vertex projection implemented on the GPU [61, 62] would improve timings.

## 6 Examples and Applications

We illustrate the new capabilities of anisotropic convolution

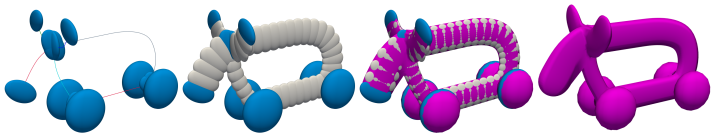
through three example applications: skeleton-based modeling, more general implicit modeling, and surface approximation.

## 6.1 Skeleton-based modeling

As with previous convolution surfaces, skeleton-based modeling is a direct application of our new formulation. The user only needs to define the circular splines with ellipsoids at the end-points. The current implementation does not provide real-time generation of the surface. However, it should be noted that due to the properties of anisotropic convolution surfaces, the implicit surface is close to the infinite union of interpolated ellipsoids along the splines. A dense enough subset of those ellipsoids can therefore provide a real-time preview to guide the user during modeling (see Figure 17).

In Figure 18 we show an example of surface meshes around a closed curve. The anisotropic surfaces are defined on a circular spline approximation of the smooth curve. We showcase the correction angle needed for RMFs around closed curves (sections 2.2 and 3.2.1).

In Figure 19 we show a salamander model. Anisotropy, tangential radii variation and RMF were instrumental for the tail and head of the salamander. We also show a wire-frame representation of the quad mesh.



(a) User inputs. (b) Interpolation. (c) Convolution. (d) Surface.

Figure 17: Elk model (10 biarcs, 15 line segments). Note how the interpolating ellipsoids provide an accurate preliminary visualization for the final convolution surface.

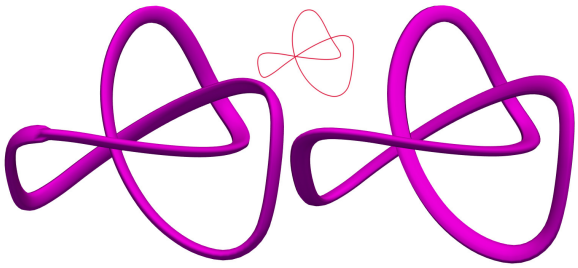


Figure 18: Convolution surface around a circular spline approximation of the knot $\Gamma(t) = (-10\cos t - 2\cos 5t + 15\sin 2t, -15\cos 2t + 10\sin t - 2\sin 5t, 10\cos 3t)$, $t \in [0, 2\pi]$. Anisotropic convolution parameters: $\alpha_0 = \alpha_1 = \gamma_0 = \gamma_1 = 1, \beta_0 = \beta_1 = 1/4$. On the left the surface without rotation: $\theta_0 = \theta_1 = 0$, on the right the surface with a corrective rotation: $\theta_0 = 0$ and $\theta_1 = -1.89$. In the middle we show the skeleton. We used 19 biarcs to approximate $\Gamma$.

## 6.2 Blobtree modeling

We also used the new formulation in a more general context, namely *Blobtree modeling* [54]. We follow the adequate inner bound methodology [12] in order to define a smooth carving operator from Ricci's blending operator [41]. For this purpose, we used the transfer function proposed in [41] to re-map field value to the expected range ($[0,1]$ with 0.5 as level value). Figure 20 depicts a cup modeled with this approach. All primitives used in this object rely on anisotropy, and noticeably on the anisotropy in the tangential direction. Note that when a large anisotropy in the tangent direction is used (as done on the foot of the cup) the maximal amount of blending that could be achieved by subsequent operations on the scalar field is reduced. We did not rely on our meshing approach for this object due to the large difference between the skeleton and the medial axis of the final surface (mainly due to the carving used to hollow the cup).

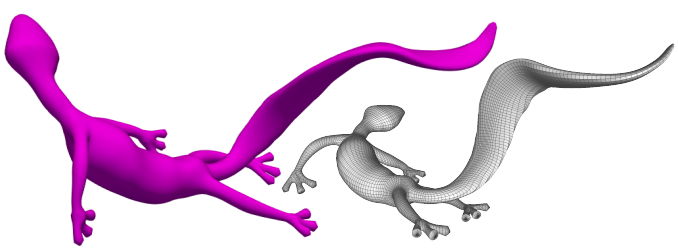


Figure 19: Salamander model (45 biarcs, 48 line segments). Left: surface computed with our meshing technique. Right: wire-frame showing the quads of the mesh.

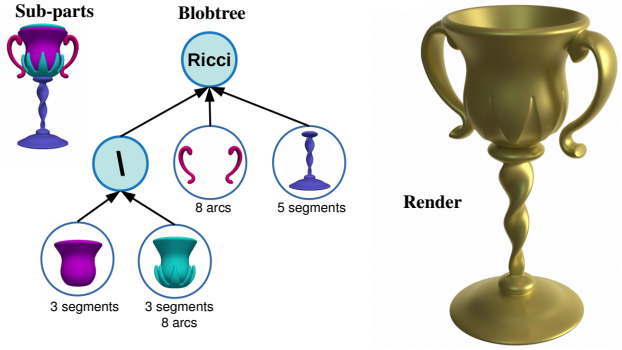



Figure 20: A cup model. Left: the *blobtree* used to design the cup, 27 primitives where used in total, including 11 for the carving. Right: rendering of a mesh extracted with Marching Cubes. Note the relevance of twisting and anisotropy, both in the normal and tangent directions, to model the different smooth parts by our new convolution with few parameters.

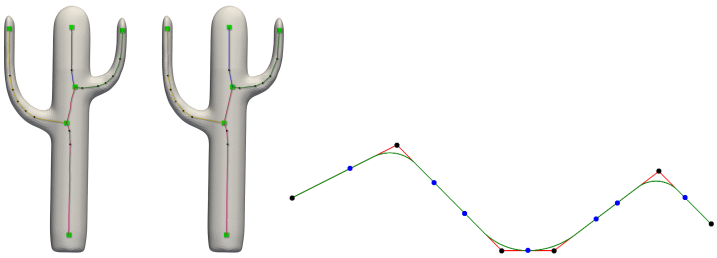
## 6.3 Shape approximation

Finally, we use anisotropic convolution in order to compute semi-automatically smooth and compact representation of surfaces representing volumes.

The steps of our approximation method are:

1. Compute skeleton consisting of short line segments.
2. Identify *branches* on the skeleton.
3. Simplify and approximate each branch.
4. Optimize the parameters of anisotropic convolution on each branch.

The first three steps give the set of curves defining the skeleton and are described in Section 6.3.1. The last step, computing the parameters along the skeletons, is described in Section 6.3.2.



(a) Branch simplification. (b) Circular spline from polyline.

Figure 21: Post-processing of skeletonization algorithms output. In picture (a), on the left, the output of a skeletonization algorithm; on the right, the simplified branch polylines. We highlight the branches of the skeleton in different colors. In black, we show the points selected as vertices of the simplified polyline. The *branching points* appear in bright green. In picture (b), we show the tangential circular spline, in green, obtained from the red polyline. The original points of the polyline are shown in black; in blue, the tangential points.

### 6.3.1 Skeleton approximation

Given the output $\mathscr{O}$ of a skeletonization algorithm, we define a *branch* as the portion of $\mathscr{O}$ connecting two *branching points*. In general a point of $\mathscr{O}$ is considered a *branching point* if it is connected with either only one or more than two other points in the skeleton. A point in $\mathscr{O}$ is identified as additional branching point if: a sudden change of distance from the model surface to the skeleton is detected, or the incident line segments have a sharp angle ($< \pi/2$). Some extra branching points may be identified by the user to better reflect features of the input model. To reduce the number of line segments, we simplify each branch with a polyline simplification algorithm [14] (Figure 21a). When the tips of the branches are not sufficiently close to the input model (along tangent direction), we extend the skeletons. Additional manual editing of the skeleton can be performed by the user (as illustrated in Figure 22). The collection of simplified polyline branches is denoted by $\mathscr{B}$. For each polyline $B \in \mathscr{B}$ we compute a tangential circular spline approximation $\Gamma_B$ that interpolates the endpoints of $B$. Figure 21b illustrates this process.

### 6.3.2 Parameters optimization along circular spline

To fit an anisotropic convolution on each spline (Step 4) we first associate each vertex in the input mesh with its closest branch. For each branch $B \in \mathscr{B}$, $\mathscr{P}_B$ denotes the set of the associated vertices.

Then we perform a non-linear least square optimization [21, Chapter 8] of the anisotropic convolution field (7). For each circular spline $\Gamma_B$ ($B \in \mathscr{B}$) we look for the parameters $\alpha_i, \beta_i, \gamma_i, \theta_i$ ($i = 0, 1$) of its corresponding anisotropic convolution field $C^K_{\Gamma_B}$ that minimize

$$\sum_{P \in \mathscr{P}_B} \left( w(C^K_{\Gamma_B, \mathbf{G}}(P)) - c \right)^2. \tag{16}$$

With $w(\cdot)$ a weighting function that rescales values outside the convolution surface. This is needed because field values inside the convolution function are in the range $(c, 2]$, while outside the range is $[0, c)$. For $c = 0.1$ (chosen level set) this would give an imbalance on the least squares optimization. The general expression for $w$ is

$$w(x) = \begin{cases} W(x-c)+c & 0 \le x \le c \\ x & \text{otherwise} \end{cases} \tag{17}$$

where $W \in \mathbb{R}$ is a constant that controls the additional weight for points outside the convolution surface. Experimental results show that a value of $W$ between 3 and 5 gives better fits for our chosen kernel (Section 3) and level set ($c = 0.1$).

In order to attain a valid solution the optimization parameters in (16) must be bounded. In practice we take $\alpha_i = 1$, $-\frac{\pi}{2} \le \theta_i \le \frac{\pi}{2}$, $0 < \beta_i, \gamma_i < e_B$, with $e_B$ taken such that all the points $P \in \mathscr{P}_B$ are inside the convolution surface defined by $\alpha_i = 1, \beta_i = \gamma_i = e_B, \theta_i = 0$ on the level set $c$. These values are also taken as the starting point in the optimization. They guarantee that the starting field is nonzero on every point $P \in \mathscr{P}_B$, i.e. all the points are inside the surface. For compact support kernels this is required to secure a non-zero gradient: outside the support the field is constantly zero.

### 6.3.3 Approximation experiments and discussion

In Figure 22 we show a semi-automatic anisotropic fitting resembling the fertility model. In this experiment the user modified the circular spline skeleton obtained from the output of a skeletonization algorithm (Section 6.3.1). Then the parameters defining the convolution surfaces were computed with our optimization step (Section 6.3.2).

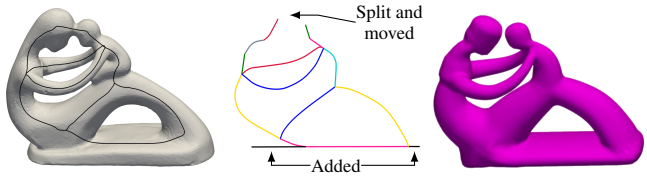



Figure 22: Semi-automatic approximation of the fertility model by anisotropic convolution surface. Left: Fertility model and output (black polyline) of the skeletonization algorithm in [56]. Middle: Circular splines after manual editing - adding the base (black lines) and splitting the top (red and green splines). Right: Result of the anisotropic convolution fitting process.

In Figure 23 we compare the results of automatic fittings to a cactus model with two skeletonization methods. The fitting is done on the original model, a randomly decimated model, and a model with holes. In the decimated models we reduced the number or points on the mesh to one fifth, reducing the computational time of the optimization accordingly. The original model has around 5200 vertices. In terms of Hausdorff distance, it is noticeable how the decimated model results in as good approximation as the original fitting. Similarly, for incomplete models (with holes) the fitting performs well.

## 7 Conclusions

We have extended the modeling capabilities of convolution surfaces with a method that controls the anisotropy of the surface

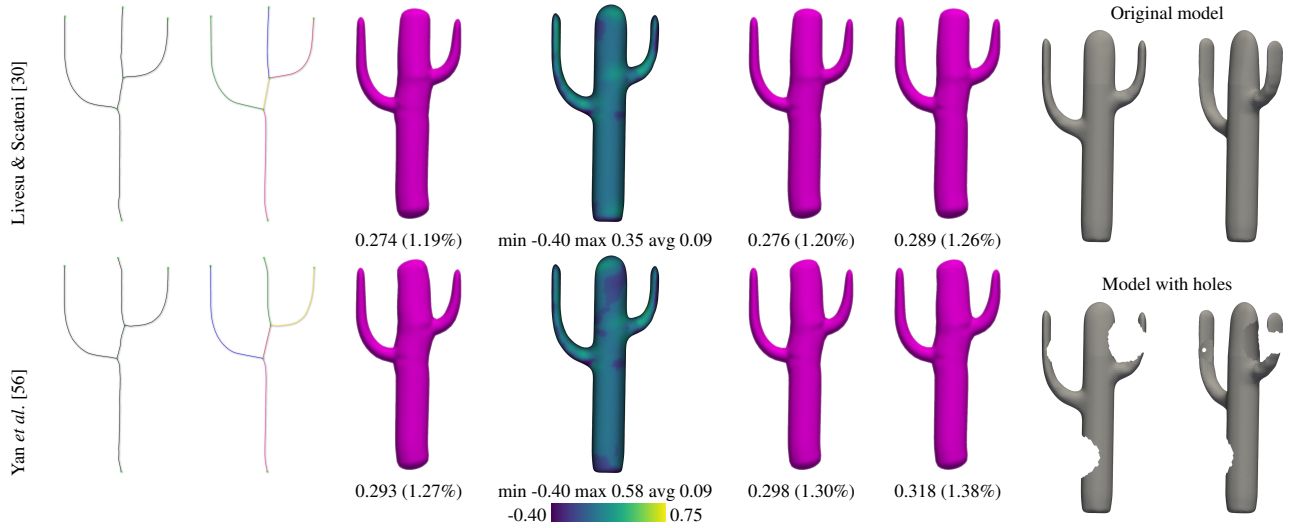



Figure 23: Surface approximation via skeletonization. On the left we show the output of the fitting process for the models on the right. Columns, from left to right: output of skeletonization algorithms, circular spline skeletons, fitted surfaces, weighted field value deviations from the level value ($c = 0.1$), fitted surface on a decimated model (one-fifth of points), fitted surface on a decimated model (one-fifth) with holes. Below the fitted surfaces we show the Hausdorff distances to the original model, in magnitude and as a percent of the bounding box diagonal (23.0 units).

around the skeleton. For $\mathscr{G}^1$ skeletal curves we introduced the use of biarcs approximation in order to use less pieces, while still retaining $\mathscr{G}^1$ continuity for the skeleton and smoothness in the final surface. The $\mathscr{G}^1$ skeleton allows to define a shape along the curve that is intuitive and easy to model, with few parameters that guarantee control over the radii around the skeleton. We developed a meshing technique for convolutions surfaces based on a scaffolding technique that generates a quad-dominant mesh that follows the structure of the skeleton. The variety of shapes and new modeling freedom is showcased in several examples. Mimicking shapes generated with 2D skeletons is another advantage of the new formulation. We also presented applications of anisotropic convolution to skeleton-based modeling, general implicit modeling, and surface approximation.

# References


[1] ANGELIDIS, A., AND CANI, M.-P. Adaptive implicit modeling using subdivision curves and surfaces as skeletons. *Proceedings of the seventh ACM symposium on Solid modeling and applications - SMA '02* (2002), 45.

[2] AXLER, S. *Linear Algebra Done Right*. Undergraduate Texts in Mathematics. Springer International Publishing, 2015.

[3] BÆRENTZEN, J. A., ABDRASHITOV, R., AND SINGH, K. Interactive shape modeling using a skeleton-mesh co-representation. *ACM Trans. Graph. 33*, 4 (2014), 1–10.

[4] BÆRENTZEN, J. A., MISZTAL, M. K., AND WEŁNICKA, K. Converting skeletal structures to quad dominant meshes. *Computers and Graphics 36*, 5 (2012), 555–561.

[5] BARBIERI, S., MELONI, P., USAI, F., SPANO, L. D., AND SCATENI, R. An interactive editor for curve-skeletons: SkeletonLab. *Computers & Graphics 60* (2016), 23–33.

[6] BERNHARDT, A., PIHUIT, A., CANI, M.-P., AND BARTHE, L. Matisse: Painting 2D regions for Modeling Free-Form Shapes. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling* (2008), C. Alvarado and M.-P. Cani, Eds., The Eurographics Association.

[7] BISHOP, R. L. There is More than One Way to Frame a Curve. *The American Mathematical Monthly 82*, 3 (1975), 246–251.

[8] BLINN, J. F. A Generalization of Algebraic Surface Drawing. *ACM Transactions on Graphics 1*, 3 (1982), 235–256.

[9] BLOOMENTHAL, J., AND SHOEMAKE, K. Convolution surfaces. *ACM SIGGRAPH Computer Graphics 25*, 4 (1991), 251–256.

[10] BOLTON, K. Biarc curves. *Computer-Aided Design 7*, 2 (1975), 89–92.

[11] BRENT, R. P. *Algorithms for Minimization Without Derivatives*. Dover Books on Mathematics. Dover Publications, 2002.

[12] CANEZIN, F., GUENNEBAUD, G., AND BARTHE, L. Adequate inner bound for geometric modeling with compact field functions. *Computers & Graphics 37*, 6 (2013), 565–573.

[13] DO CARMO, M. P. *Differential geometry of curves and surfaces*. Prentice-Hall, New Jersey, 1976.

[14] DOUGLAS, D. H., AND PEUCKER, T. K. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization 10*, 2 (dec 1973), 112–122.

[15] ENTEM, E., BARTHE, L., CANI, M.-P., CORDIER, F., AND VAN DE PANNE, M. Modeling 3d animals from a side-view sketch. *Computers & Graphics 46* (2015), 221–230.

[16] FLANDERS, H. Differentiation under the integral sign. *American Mathematical Monthly 80*, 6 (1973), 615–627.

[17] FRYAZINOV, O., AND PASKO, A. Implicit variable-radius arc canal surfaces for solid modeling. *Computer-Aided Design and Applications 14*, 3 (2017), 251–258.

[18] FUENTES SUÁREZ, A. J., AND HUBERT, E. Convolution surfaces with varying radius: Formulae for skeletons made of arcs of circles and line segments. In *Research in Shape Analysis: WiSH2, Sirince, Turkey*, AWM. Springer, 2018, pp. 37–60.

[19] FUENTES SUÁREZ, A. J., AND HUBERT, E. Scaffolding skeletons using spherical Voronoi diagrams: Feasibility, regularity and symmetry. *Computer-Aided Design 102* (Sep 2018), 83–93.

[20] GALASSI, M., DAVIES, J., THEILER, J., GOUGH, B., JUNGMAN, G., ALKEN, P., BOOTH, M., ROSSI, F., AND ULERICH, R. *GNU Scientific Library*. No. Release 2.4. GNU, 2017.

[21] GOMES, A. J., VOICULESCU, I., JORGE, J., WYVILL, B., AND GALBRAITH, C. *Implicit Curves and Surfaces: Mathematics, Data Structures and Algorithms*. Springer London, 2009.

[22] HORNUS, S., ANGELIDIS, A., AND CANI, M.-P. Implicit modeling using subdivision curves. *The Visual Computer 19*, 2 (2003), 94–104.

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

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

[25] JI, Z., LIU, L., AND WANG, Y. B-Mesh: A modeling system for base meshes of 3D articulated shapes. *Computer Graphics Forum 29*, 7 (2010), 2169–2177.

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

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

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

[29] JIN, X., TAI, C.-L., AND ZHANG, H. Implicit modeling from polygon soup using convolution. *The Visual Computer 25*, 3 (2008), 279–288.

[30] LIVESU, M., AND SCATENI, R. Extracting curve-skeletons from digital shapes using occluding contours. *Visual Computer 29*, 9 (2013), 907–916.

[31] LORENSEN, W. E., AND CLINE, H. E. Marching cubes: A high resolution 3d surface construction algorithm. *ACM SIGGRAPH Computer Graphics 21*, 4 (1987), 163–169.

[32] MCCORMACK, J., AND SHERSTYUK, A. Creating and rendering convolution surfaces. *Computer Graphics Forum 17*, 2 (1998), 113–120.

[33] MEEK, D., AND WALTON, D. The family of biarcs that matches planar, two-point $G^1$ Hermite data. *Journal of Computational and Applied Mathematics 212*, 1 (2008), 31–45.

[34] NUTBOURNE, A. W., AND MARTIN, R. R. *Differential geometry applied to curve and surface design*. John Wiley & Sons, 1988.

[35] PANOTOPOULOU, A., ROSS, E., WELKER, K., HUBERT, E., AND MORIN, G. Scaffolding a skeleton. In *Research in Shape Analysis: WiSH2, Sirince, Turkey*, AWM. Springer, 2018, pp. 17–35.

[36] PASKO, A., ADZHIEV, V., SOURIN, A., AND SAVCHENKO, V. Function representation in geometric modeling: concepts, implementation and applications. *The Visual Computer 11*, 8 (1995), 429–446.

[37] PETERNELL, M., AND POTTMANN, H. Computing rational parametrizations of canal surfaces. *Journal of Symbolic Computation 23*, 2-3 (1997), 255–266.

[38] PHAM, B. Offset curves and surfaces: a brief survey. *Computer-Aided Design 24*, 4 (1992), 223–229.

[39] PIESSENS, R., DE DONCKER-KAPENGA, E., ÜBERHUBER, C. W., AND KAHANER, D. K. *Quadpack*. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 1983.

[40] REQUICHA, A. G. Representations for rigid solids: Theory, methods, and systems. *ACM Computing Surveys 12*, 4 (1980), 437–464.

[41] RICCI, A. A constructive geometry for computer graphics. *The Computer Journal 16*, 2 (1973), 157–160.

[42] ROUSSELLET, V., RUMMAN, N. A., CANEZIN, F., MELLADO, N., KAVAN, L., AND BARTHE, L. Dynamic implicit muscles for character skinning. *Computers & Graphics 77* (dec 2018), 227–239.

[43] SHERSTYUK, A. Interactive shape design with convolution surfaces. In *Proceedings Shape Modeling International 99. International Conference on Shape Modeling and Applications* (1999), IEEE, pp. 56–65.

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

[45] SONG, X., AIGNER, M., CHEN, F., AND JÜTTLER, B. Circular spline fitting using an evolution process. *Journal of Computational and Applied Mathematics 231*, 1 (2009), 423–433.

[46] TAGLIASACCHI, A., DELAME, T., SPAGNUOLO, M., AMENTA, N., AND TELEA, A. 3D Skeletons: A State-of-the-Art Report. *Computer Graphics Forum 35*, 2 (2016), 573–597.

[47] TAI, C.-L., ZHANG, H., AND FONG, J. C.-K. Prototype modeling from sketched silhouettes based on convolution surfaces. *Computer Graphics Forum 23*, 1 (mar 2004), 71–83.

[48] USAI, F., LIVESU, M., PUPPO, E., TARINI, M., AND SCATENI, R. Extraction of the Quad Layout of a Triangle Mesh Guided by Its Curve Skeleton. *ACM Transactions on Graphics 35*, 1 (2015), 1–13.

[49] WANG, W., AND JOE, B. Robust computation of the rotation minimizing frame for sweep surface modeling. *Computer-Aided Design 29*, 5 (1997), 379–391.

[50] WANG, W., JÜTTLER, B., ZHENG, D., AND LIU, Y. Computation of rotation minimizing frames. *ACM Transactions on Graphics 27*, 1 (2008), 1–18.

[51] WENGER, R. *Isosurfaces*. A K Peters/CRC Press, New York, 2013.

[52] WITHER, J., BOUDON, F., CANI, M.-P., AND GODIN, C. Structure from silhouettes: a new paradigm for fast sketch-based design of trees. *Computer Graphics Forum 28*, 2 (apr 2009), 541–550.

[53] WU, J., AND LIU, L. Generating quad mesh of 3d articulated shape for sculpting modeling. *Journal of Advanced Mechanical Design, Systems, and Manufacturing 6*, 3 (2012), 354–365.

[54] WYVILL, B., GUY, A., AND GALIN, E. Extending the CSG tree. Warping, blending and boolean operations in an implicit surface modeling system. *Computer Graphics Forum 18*, 2 (1999), 149–158.

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

[56] YAN, Y., SYKES, K., CHAMBERS, E., LETSCHER, D., AND JU, T. Erosion thickness on medial axes of 3D shapes. *ACM Transactions on Graphics 35*, 4 (Jul 2016), 1–12.

[57] YAO, C.-Y., CHU, H.-K., JU, T., AND LEE, T.-Y. Compatible quadrangulation by sketching. *Computer Animation and Virtual Worlds 20*, 2-3 (2009), 101–109.

[58] ZANNI, C. *Skeleton-based Implicit Modeling & Applications*. PhD thesis, Université de Grenoble, 2013.

[59] ZANNI, C., BERNHARDT, A., QUIBLIER, M., AND CANI, M.-P. SCALe-invariant integral surfaces. *Computer Graphics Forum 32*, 8 (2013), 219–232.

[60] ZANNI, C., HUBERT, E., AND CANI, M.-P. Warp-based helical implicit primitives. *Computers & Graphics 35*, 3 (jun 2011), 517–523.

[61] ZHU, X., JIN, X., LIU, S., AND ZHAO, H. Analytical solutions for sketch-based convolution surface modeling on the GPU. *The Visual Computer 28*, 11 (2011), 1115–1125.

[62] ZHU, X., JIN, X., AND YOU, L. Analytical solutions for tree-like structure modelling using subdivision surfaces. *Computer Animation and Virtual Worlds 26*, 1 (2015), 29–42.

[63] ZHU, X., SONG, L., YOU, L., ZHU, M., WANG, X., AND JIN, X. Brush2Model: Convolution surface-based brushes for 3D modelling in head-mounted display-based virtual environments. *Computer Animation and Virtual Worlds 28*, 3-4 (2017), e1764.

**Acknowledgment** This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 675789, and the project IMPRIMA (ANR-18-CE46-0004).