Computational Visual Media
DOI 10.1007/s41095-017-0086-4 Vol. 3, No. 3, September 2017, 199–215

Research Article

Smooth shapes with spherical topology: Beyond traditional modeling, efficient deformation, and interaction

D. Schmitter[1] (✉), **P. García-Amorena**[1], **and M. Unser**[1]

Abstract Existing shape models with spherical topology are typically designed either in the *discrete* domain using *interpolating* polygon meshes or in the continuous domain using *smooth* but *non-interpolating* schemes such as subdivision or NURBS. Both polygon models and subdivision methods require a large number of parameters to model smooth surfaces. NURBS need fewer parameters but have a complicated rational expression and non-uniform shifts in their formulation. We present a new method to construct deformable closed surfaces, which includes exact spheres, by combining the best of two worlds: a *smooth, interpolating* model with a continuously varying tangent plane and well-defined curvature at every point on the surface. Our formulation is considerably simpler than NURBS and requires fewer parameters than polygon meshes. We demonstrate the generality of our method with applications including intuitive user-interactive shape modeling, continuous surface deformation, shape morphing, reconstruction of shapes from parameterized point clouds, and fast iterative shape optimization for image segmentation. Comparisons with discrete methods and non-interpolating approaches highlight the advantages of our framework.

Keywords shape modeling; spherical topology; parametric surfaces; splines; differential geometry

1 Biomedical Imaging Group, École Polytechnique Fédérale de Lausanne (EPFL), 1015 Lausanne, Switzerland. E-mail: D. Schmitter, daniel.schmitter@epfl.ch (✉); P. García-Amorena, pablo.garcia-amorenagarcia@epfl.ch; M. Unser, michael.unser@epfl.ch.

Manuscript received: 2017-01-24; accepted: 2017-04-23

1 Introduction

1.1 Background

The representation of shapes with spherical topology has been an ongoing research topic in computer graphics for more than three decades. The principal reason is the massive demand for closed genus-zero surfaces in industrial, architectural, and animation design as well as in biomedical imaging. Designing spherical-topology models that are simultaneously optimal with respect to several different shape characteristics still remains a challenge. Depending on whether an application involves user interaction, shape deformation, or optimization schemes, different aspects of a model are more important than others.

In user-interactive applications, a fundamental requirement is the ability to intuitively manipulate the shape. Typically, this requirement presupposes an easy way to interact directly with the surface as well as to control shapes locally. The surface deformation should be stable: a small perturbation of the surface should result in a small change of the shape. Numerical stability is crucial too: a theoretical model must remain useful in practice. On the other hand, an application might involve shape deformation as an optimization process. For example, in real-time shape recognition, approximation, and segmentation, the fast evaluation of derivative- and integral-based quantities in iterative settings is required. Further, the smoothness of the surface and the number of parameters involved can also play important roles. Usually, it is impossible to find a model that is optimal with respect to all of these requirements. In practice, a compromise is

made favoring the most important needs of a specific application. Existing models are based on polygon meshes, NURBS, or subdivision.

1.2 Overview and contribution

This article presents the full theory of a model for constructing deformable shapes with spherical topology, along with applications. This work was first introduced in a condensed form as a SIGGRAPH Asia technical brief in 2016 [1]. Our framework is based on interpolating control points, similar to polygon meshes, while at the same time providing a smooth surface, formulated in the continuous domain as for NURBS. The resulting surface allows local control, is everywhere differentiable, and has a continuously varying tangent plane at every point on the surface as well as a well-defined Gaussian curvature. A major contribution is an explicit formulation of necessary conditions for the poles of the sphere to remain closed and smooth when deforming. We illustrate the use of our method with several applications. (1) User-interactive shape modeling: as the basis functions are interpolating, the control points lie directly on the surface of the object, which facilitates intuitive shape modeling. The basis is also finitely supported, enabling local surface control; it allows us to model a broad range of shapes by deforming a single spherical surface patch. (2) Smooth surface reconstruction from parameterized point clouds: if the underlying spherical parameterization of the samples is known, they can be easily interpolated with our model to reconstruct a smooth surface. (3) Efficient surface deformation: by exploiting the affine invariance of our model, we illustrate how a fast implementation of minimum-energy deformation algorithms can be achieved. (4) Fast iterative optimization of deformation algorithms: we show how the iterative evaluation of surface and volume integrals can be efficiently implemented for real-time optimization, and provide an example of a segmentation algorithm for 3D medical images.

Our construction involves a class of smooth non-rational basis functions that have uniform shifts, which leads to a considerably simpler formulation than for traditional parametric methods. A control-point-based structure allows us to use fewer parameters than polygon or subdivision methods to achieve smoothness. Examples of the use of our method are shown in Fig. 1.

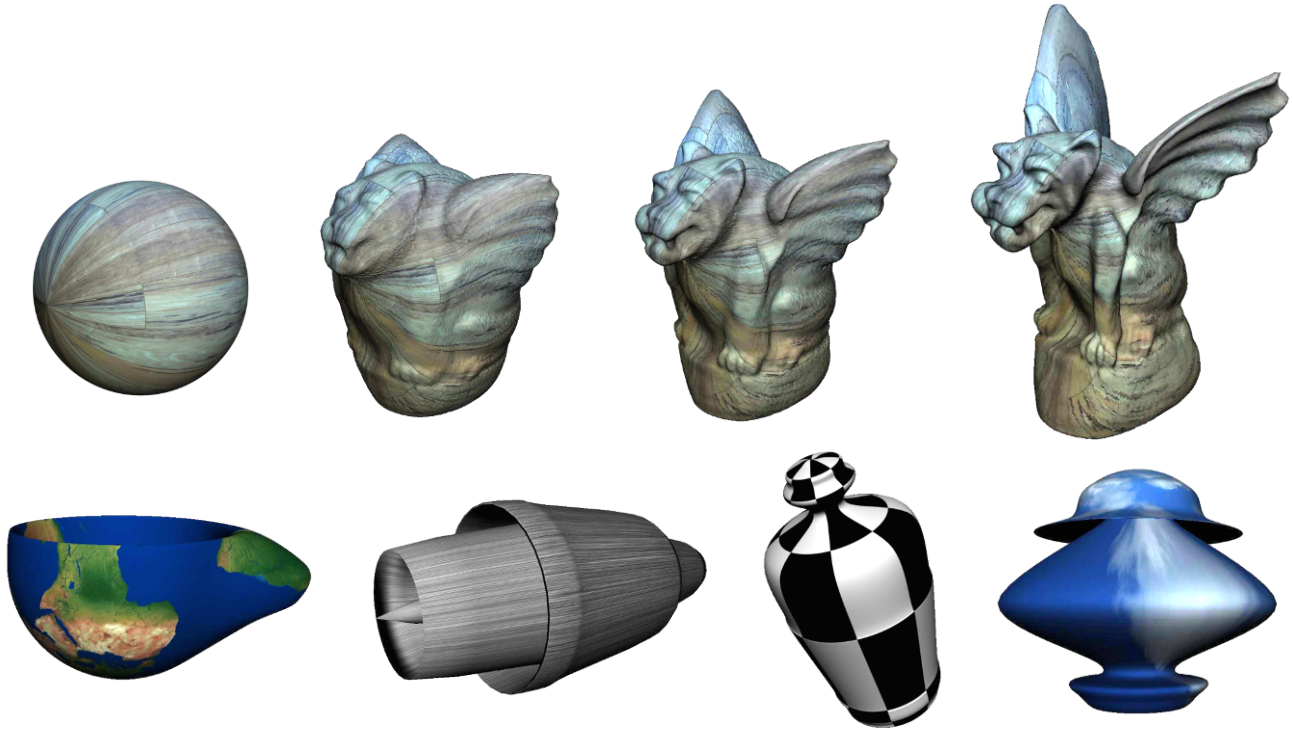

Fig. 1 Smooth modeling of shapes with spherical topology. Top: continuous deformation of a sphere into a gargoyle; a wood texture has been added to the surface. Bottom: shapes consisting of a single surface patch, constructed by interactive deformation of a sphere. The interpolating structure of the model allows us to intuitively design surfaces that can adopt shapes beyond classical spherical topology. Our framework is inherently smooth, facilitating natural texturing.

2 Related work

2.1 Continuous closed surfaces

The most widely used technique to construct deformable spheres in the continuous domain is NURBS [2, 3] which are a subfamily of T-splines [4]. Parametric NURBS surfaces are based on polynomial B-splines and are defined by a set of control points which allow local shape control [3, 5–8]. The main reason for using rational NURBS instead of (non-rational) polynomial B-splines is that NURBS are able to exactly reproduce conic sections [9]. Conceptually, this property is equivalent to reproducing trigonometric functions, which is a necessary requirement for constructing spheres. Several ways of constructing NURBS spheres exist, e.g., by constructing quarter or half circles and exploiting the properties of tensor-product splines or by constructing surfaces of revolution [10]. NURBS typically involve the explicit characterization of non-uniform knot vectors with double knots. A drawback of NURBS is their rational form, which leads to complicated expressions for related integrals and derivatives [7]. Furthermore, the NURBS formulation depends on additional weight parameters, which have no intuitive interpretation. Other constructions to approximate sphere-like surfaces based on B-splines have been studied in Refs. [11, 12], whereas in Ref. [13] an exact approach using exponential splines is proposed. Other models use (rational) Bézier surfaces [14], which are also related to splines [15].

2.2 Discrete closed surfaces

Popular discrete methods are based on polygon meshes [16, 17]. With these models it is possible to represent shapes of arbitrary topology and hence, closed surfaces with spherical topology can be easily generated. A vast literature exists on mesh optimization, processing and discretizing continuous-domain operators (e.g., see Refs. [18, 19]). Polygon models are usually interpolating the control points coinciding with the mesh vertices; this property implies that the shape is modified by points which directly lie on the boundary of the object. Related to polygon models are subdivision methods [20, 21] used to construct surfaces [22–26]. These methods are characterized by refinement operations iteratively applied to a set of points leading to a continuous limit surface with a certain regularity. Hence, subdivision can be seen as a hybrid method combining the discrete and the continuous-domain approach. Although in theory they are continuous, in practice, a finite number of iterations are applied, leading to a discrete mesh (thus, we categorize subdivision as a discrete method). As opposed to polygon mesh models, subdivision methods do not necessarily have interpolating control points. Different methods based on non-stationary refinement rules have been proposed to approximate spheres using subdivision [27–29]. One drawback of polygon and subdivision methods is that they require a large number of parameters which can be a challenge when computational speed is important (e.g., in finite element models [30]).

2.3 Spherical parameterization

The problem of finding a parameterization for an object with spherical topology is not trivial and has been tackled in Refs. [31, 32]. It is linked to the problem of ordering an unorganized set of points or a point cloud, which is significantly harder in 3D than in 2D. In Ref. [33], a method is presented to generate a spherical parameterization of a closed surface in the continuous domain by expressing it in a basis of spherical harmonics. A related problem is surface reconstruction from a point cloud [34, 35].

2.4 Interpolation

A widely used interpolating spline in computer graphics is the Catmull–Rom spline [36]. However, its nature is polynomial and hence, it cannot be used to exactly parameterize a sphere. A variant of the Catmull–Rom spline used in signal processing is the Keys cubic convolution interpolator [37] which has been generalized by Refs. [38, 39] to construct a trigonometric interpolation kernel that is able to reproduce conic sections. Other variants have been presented in Refs. [40–42]. An interpolating subdivision scheme was originally introduced by Deslaurier and Dubuc [43]. Variants of this scheme have been proposed in Ref. [44] which have also been used to construct conic sections [45].

3 Parametric shape representation

We use bold font for vectors and plain font for scalars, e.g., $\boldsymbol{c} = (c_x, c_y, c_z)$. To denote partial

derivatives, we use the notation $\partial\boldsymbol{\sigma}(u,v)/\partial u = \boldsymbol{\sigma}_u(u,v)$.

Note that throughout this article we will use the terms *spherical topology* and *closed surface* to describe the same kind of objects, namely connected surfaces without holes or boundaries. Using these terms to describe equivalent objects makes sense in computer graphics because in a digital environment, even continuous-domain objects can only be represented by a discretized approximation. However, in the field of mathematical topology a more rigorous definition of these terms would be required.

4 Tensor-product surfaces

4.1 Basics

We construct parametric shapes using integer shifts of a (non-rational) generator function φ. A 3D curve $\boldsymbol{r}(t)$ that is described by the coordinate functions $x(t)$, $y(t)$, and $z(t)$ with $t \in \mathbb{R}$ is then represented by a linear combination of integer shifts of φ as

$$\boldsymbol{r}(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix} = \sum_{k\in\mathbb{Z}} \boldsymbol{c}[k]\varphi(t-k) \tag{1}$$

where $\{\boldsymbol{c}[k] = (c_x[k], c_y[k], c_z[k])\}_{k\in\mathbb{Z}}$ are the 3D *control points*. The model (1) may be extended to construct a separable parametric *tensor product* surface $\boldsymbol{\sigma}(u,v)$ with $u,v \in \mathbb{R}$, represented as the component-wise product (denoted by the symbol $\times$) of two curves $\boldsymbol{r}_1 \times \boldsymbol{r}_2$, i.e.

$$\begin{aligned}\boldsymbol{\sigma}(u,v) &= \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix} = \begin{pmatrix} x_1(u)\cdot x_2(v) \\ y_1(u)\cdot y_2(v) \\ z_1(u)\cdot z_2(v) \end{pmatrix} \\ &= \underbrace{\sum_{k\in\mathbb{Z}} \boldsymbol{c}_1[k]\varphi_1(u-k)}_{\boldsymbol{r}_1} \times \underbrace{\sum_{l\in\mathbb{Z}} \boldsymbol{c}_2[l]\varphi_2(v-l)}_{\boldsymbol{r}_2} \\ &= \sum_{k\in\mathbb{Z}}\sum_{l\in\mathbb{Z}} \underbrace{\boldsymbol{c}_1[k]\times\boldsymbol{c}_2[l]}_{\boldsymbol{c}[k,l]} \varphi_1(u-k)\varphi_2(v-l) \end{aligned} \tag{2}$$

Based on this equation, an arbitrary non-separable surface with control points $\boldsymbol{c}[k,l]$ can be constructed whose expression corresponds to the last line of Eq. (2).

4.2 Generator function φ

The shapes that model (2) can produce depend on the generator φ. For example, if φ is a B-spline, the resulting shapes are polynomial. In our case, we are interested in generating trigonometric shapes in order to be able to construct exact spheres. For this purpose, we use the piecewise exponential generator proposed by Ref. [39], which reproduces sines and cosines. It is defined as $\varphi = \beta * \psi$, where β is a third order exponential B-spline, ψ is an appropriate smoothing kernel and, $*$ denotes convolution. We provide the explicit expression for φ in Appendix A. The relevant characteristics of φ for our construction, besides its sphere-reproduction property, are that it is twice differentiable, with bounded second derivatives, and satisfies the interpolation property $\varphi(t=k) = \delta_k$, where δ_k denotes the Kronecker delta, $t \in \mathbb{R}$, and $k \in \mathbb{Z}$. Our generator constitutes a partition of unity, i.e., $\sum_k \varphi(t-k) = 1$, which is a necessary and sufficient condition for the represented shapes to be *affine invariant*. Because this generator depends on the number M of control points used to construct a curve $\boldsymbol{r}$, we use the notation φ_M instead of φ. The support of φ_M is equal to 4.

4.3 Definitions

As a simplification to indicate the M_1-periodized basis function, we write:

$$\phi_1(t) := \varphi_{M_1,\mathrm{per}}(t) = \sum_{n=-\infty}^{+\infty} \varphi_{M_1}(t - M_1 n) \tag{3}$$

and $\phi_2 := \varphi_{2M_2}$. To denote the integer shifts of the basis functions on the normalized parameter domain we use $\phi_{1,k}(t) := \phi_1(M_1 t - k)$ and $\phi_{2,k}(t) := \phi_2(M_2 t - k)$.

5 Spherical parameterization

5.1 The deformable sphere

In this section, we outline our proposed construction of the deformable sphere. Without loss of generality, we parameterize its surface as

$$\begin{aligned}\boldsymbol{\sigma}(u,v) &= \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix} = \begin{pmatrix} \cos(2\pi u)\sin(\pi v) \\ \sin(2\pi u)\sin(\pi v) \\ \cos(\pi v) \end{pmatrix} \\ &= \boldsymbol{r}_1(u) \times \boldsymbol{r}_2(v) \end{aligned} \tag{4}$$

with $u,v \in [0,1]$. In Ref. [39], it has been shown that:

$$\boldsymbol{r}_1\Big(\frac{u}{M_1}\Big) = \begin{pmatrix} \cos(2\pi u/M_1) \\ \sin(2\pi u/M_1) \\ 1 \end{pmatrix}$$

$$= \sum_{k=0}^{M_1-1} \begin{pmatrix} \cos\left(2\pi k/M_1\right) \\ \sin\left(2\pi k/M_1\right) \\ 1 \end{pmatrix} \phi_{1,k}(u/M_1) \qquad (5)$$

The periodization of ϕ_1 as defined in Eq. (3) allows us to express the u-dependent 1-periodic trigonometric functions in Eq. (4) using a finite sum and $M_1 \in \mathbb{Z}$ control points. The v-dependent trigonometric functions in Eq. (4) are not periodic and are expressed as

$$\boldsymbol{r}_2\Big(\frac{v}{M_2}\Big) = \begin{pmatrix} \sin(\pi v/M_2) \\ \sin(\pi v/M_2) \\ \cos(\pi v/M_2) \end{pmatrix} = \sum_{l=-1}^{M_2+1} \begin{pmatrix} \sin\left(\pi k/M_2\right) \\ \sin\left(\pi k/M_2\right) \\ \cos\left(\pi k/M_2\right) \end{pmatrix} \phi_{2,l}(v/M_2) \qquad (6)$$

Because the support of φ_M is equal to 4, for $v \in [0,1]$, we have $\varphi_{M_2}(v-l) = 0$ if $l \notin [-1,\ldots,M_2+1]$, which explains the limits of the sum in Eq. (6). Following the construction given in Eq. (2), we finally parameterize the sphere as

$$\boldsymbol{\sigma}(u,v) = \boldsymbol{r}_1(u) \times \boldsymbol{r}_2(v) = \sum_{k=0}^{M_1-1} \sum_{l=-1}^{M_2+1} \boldsymbol{c}[k,l]\phi_{1,k}(u)\phi_{2,l}(v) \qquad (7)$$

where the control points of the surface are given by its samples:

$$\boldsymbol{c}[k,l] = \begin{pmatrix} c_x[k,l] \\ c_y[k,l] \\ c_z[k,l] \end{pmatrix} = \begin{pmatrix} \cos\left(2\pi k/M_1\right)\sin\left(\pi l/M_2\right) \\ \sin\left(2\pi k/M_1\right)\sin\left(\pi l/M_2\right) \\ \cos\left(\pi l/M_2\right) \end{pmatrix} \qquad (8)$$

Note that M_1 and M_2 are the numbers of control points in the u- and v-directions. Hence, this representation allows us to construct a perfect sphere with any numbers M_1, M_2 of control points. The only condition for the integer shifts of φ_M to form a stable basis, i.e., to guarantee a stable implementation, is $M \geqslant 3$ [39]. A reconstructed sphere with interpolatory control points is shown in Fig. 2.

5.2 Smoothness conditions at the poles

Since $\varphi \in \mathcal{C}^1$, continuity is guaranteed nearly everywhere on the surface as long as the control points do not overlap. However, for the deformed sphere, smoothness is not guaranteed at the poles unless we take appropriate measures. In Ref. [11], it is shown that continuity at the poles is ensured if the deformable sphere is constructed with continuously varying tangent planes at these points. This condition is expressed mathematically as

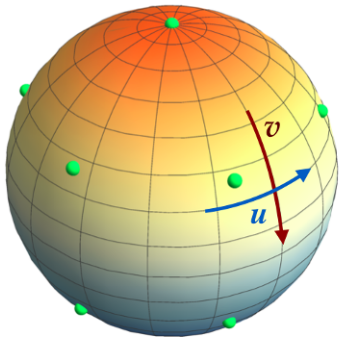

Fig. 2 Reconstructed sphere with interpolatory control points shown in green. The parametric directions are indicated by the blue and red arrows.

$$\boldsymbol{\sigma}_v(u,v)|_{v=0} = \boldsymbol{T}_{1,\mathrm{N}}\cos(2\pi u) + \boldsymbol{T}_{2,\mathrm{N}}\sin(2\pi u) \qquad (9)$$

for the North pole and

$$\boldsymbol{\sigma}_v(u,v)|_{v=1} = \boldsymbol{T}_{1,\mathrm{S}}\cos(2\pi u) + \boldsymbol{T}_{2,\mathrm{S}}\sin(2\pi u) \qquad (10)$$

for the South pole, where $\boldsymbol{T}_{1,\mathrm{N}}$, $\boldsymbol{T}_{2,\mathrm{N}}$, $\boldsymbol{T}_{1,\mathrm{S}}$, and $\boldsymbol{T}_{2,\mathrm{S}}$ are vector parameters that can be freely chosen. In Appendix B, we show that both sides of Eq. (9) can be simplified independently and we end up with the condition:

$$\boldsymbol{c}[k,-1] = \boldsymbol{c}[k,1] + \frac{\boldsymbol{T}_{1,\mathrm{N}}\cos\left(2\pi k/M_1\right) + \boldsymbol{T}_{2,\mathrm{N}}\sin\left(2\pi k/M_1\right)}{M_2\varphi'_{2M_2}(1)} \qquad (11)$$

Similarly, Eq. (10) simplifies to

$$\boldsymbol{c}[k,M_2+1] = \boldsymbol{c}[k,M_2-1] - \frac{\boldsymbol{T}_{1,\mathrm{S}}\cos\left(2\pi k/M_1\right) + \boldsymbol{T}_{2,\mathrm{S}}\sin\left(2\pi k/M_1\right)}{M_2\varphi'_{2M_2}(1)} \qquad (12)$$

The tangent plane at the poles is then spanned by the vectors $\boldsymbol{T}_{1,\mathrm{N}}$, $\boldsymbol{T}_{2,\mathrm{N}}$ and $\boldsymbol{T}_{1,\mathrm{S}}$, $\boldsymbol{T}_{2,\mathrm{S}}$. Figure 3 illustrates the effect of imposing the smoothness conditions at the poles.

5.3 Interpolation conditions at the poles

The sphere needs to remain closed when deforming in order to maintain spherical topology. Again, special attention needs to be paid to the poles: all the circles of longitude of the original sphere should originate and end at the poles of the surface. In accordance

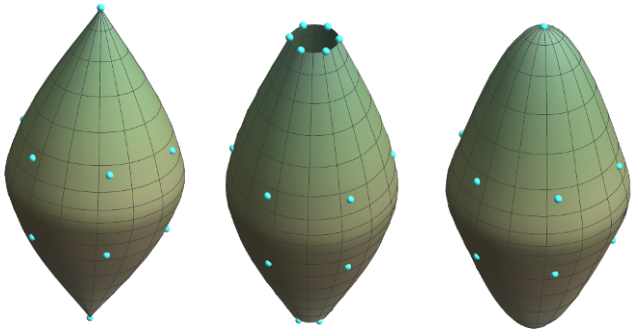

Fig. 3 Closed and smooth deformable sphere. Left: if no smoothness conditions are imposed, the surface is non-differentiable at the poles. Center: if no pole-interpolation conditions are imposed, the surface looses its spherical topology when deforming. Right: a closed and deformed sphere is shown with smoothly varying tangent planes at the poles.

with the parameterization in Eq. (4), this condition is expressed as

$$\begin{cases} \boldsymbol{\sigma}(u,0) = \boldsymbol{c}_{\mathrm{N}} & \text{(North pole)} \\ \boldsymbol{\sigma}(u,1) = \boldsymbol{c}_{\mathrm{S}} & \text{(South pole)} \end{cases} \tag{13}$$

In Appendix C, we show that condition (13) translates directly into

$$\begin{cases} \boldsymbol{c}[k,0] = \boldsymbol{c}_{\mathrm{N}} & \text{(North pole)} \\ \boldsymbol{c}[k,M_2] = \boldsymbol{c}_{\mathrm{S}} & \text{(South pole)} \end{cases} \tag{14}$$

$\forall k \in [0,\dots,M_1-1]$. In Fig. 3, we compare a deformed sphere with and without imposing the closeness conditions at the poles.

5.4 Main result

We combine all of the above considerations together in order to state the main result of this article. A locally and smoothly deformable sphere is expressed by the parameterization in Eq. (7) subject to the smoothness conditions (11) and (12) and the closeness condition (14).

5.5 Useful properties of $\boldsymbol{\sigma}$ in practice

Our deformable sphere $\boldsymbol{\sigma}$ is affine invariant. Hence, its construction is independent of location and orientation, i.e.

$$\boldsymbol{A}\boldsymbol{\sigma}(u,v) + \boldsymbol{b} = \sum_{k\in\mathbb{Z}}\sum_{l\in Z}(\boldsymbol{A}\boldsymbol{c}[k,l] + \boldsymbol{b})\phi_{1,k}(u)\phi_{2,l}(v)$$

where $\boldsymbol{A}$ is a 3×3 matrix and $\boldsymbol{b}$ a constant vector in 3D.

Furthermore, since ϕ is twice differentiable, the surface has everywhere a well-defined tangent plane and Gaussian curvature. This property, for instance, allows us to compute the normal vector at any point on the surface, an important requirement to render a textured surface.

6 Results and applications

6.1 Interactive modeling

It is crucial in interactive shape modeling that the modeling process is intuitive. Standard modeling applications allow a user to modify a shape by dragging its control points with the mouse in order to displace them. If the control points lie directly on the surface of the shape, the modeling task is significantly simplified. This is the case for polygon models, but then the underlying shape is not smooth. On the other hand, NURBS allow for the construction of smooth shapes, but the control points do not interpolate the shape. This makes the modeling task less intuitive. Local shape control is difficult as the surface becomes more complex because it is no longer clear which part of the surface is affected by a specific control point. Our proposed construction solves this problem since φ_M satisfies the interpolation condition and is also smooth. Hence, even if the modeled surface is of great complexity, the modeling process remains intuitive and simple since the control points always lie on the boundary of the shape. Furthermore, thanks to the compact support of φ_M, local shape control is guaranteed. Figure 4 illustrates the interactive shape modeling process.

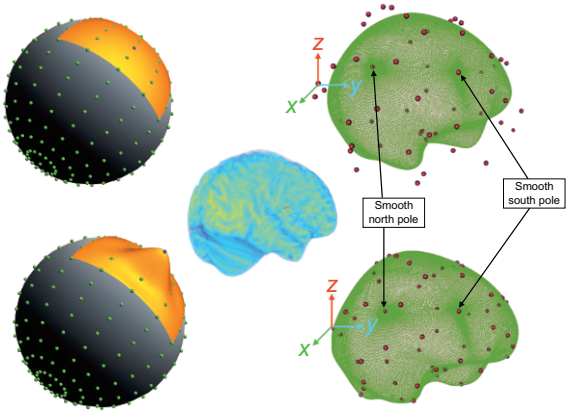

Fig. 4 Interactive modeling. Left: the region (yellow) affected by moving a single control point (blue) is shown; it corresponds to a patch of size 4×4 due to the support of the generator φ_M. Right: a brain (green) modeled using our interpolatory construction (bottom) and compared to the process where a non-interpolatory basis function is used (top) similar to NURBS. The coordinate system indicates a control point about to be interactively displaced in 3D space. Top right: it is unclear which region of the surface is controlled by a certain control point. The two poles are indicated in the figure to show the importance of the smoothness property in practice. Center: a smooth brain model rendered based on the modeling process illustrated on the right.

6.1.1 *Intuitive user interaction*

Our proposed framework can be exploited to make user-interactive shape modeling more intuitive and compelling. It is ideally suited for implementation in interactive shape modeling software, where the user modifies the shape by displacing the control points with the mouse. With relatively few control points, complex structures are easily constructed and modified. Figure 5 shows examples of the use of our framework in an interactive modeling environment. Final renderings, where texture is added to a shape, are achieved without discretization artifacts, independently of the number of control points chosen, since the underlying structure is smooth (see Fig. 1, bottom row). In Fig. 6, the effect of constraining the poles to be smooth is illustrated when performing interactive modeling.

6.2 Shape interpolation

When dealing with a parameterized point cloud whose points correspond to the samples of a surface with spherical topology, our formulation allows for an immediate reconstruction of the smooth shape. Several algorithms have been proposed to obtain such a parameterization (see Section 2). In this case, for each point $\boldsymbol{p} \in \mathbb{R}^3$ of the point cloud, a pair (u_k, v_l) of coordinates is assigned in the parameter domain and we can establish the relation $\boldsymbol{\sigma}(u_k, v_l) = \boldsymbol{p}_{k,l} = \boldsymbol{c}[k, l]$. For fixed numbers of points, M_1, M_2, in the u- and v-directions, the parametric coordinates for the normalized parametric domain, i.e., $u, v \in [0, 1]$, are given by $u_k = k/M_1$ and $v_l = l/M_2$. The resulting continuously defined surface $\boldsymbol{\sigma}(u, v)$ is immediately reconstructed since it is fully specified by its control points subject to the smoothness and pole-interpolation conditions described above. An example is shown in Fig. 7.

6.2.1 *Smooth modeling at arbitrary resolution*

Because our construction of $\boldsymbol{\sigma}$ is inherently smooth,

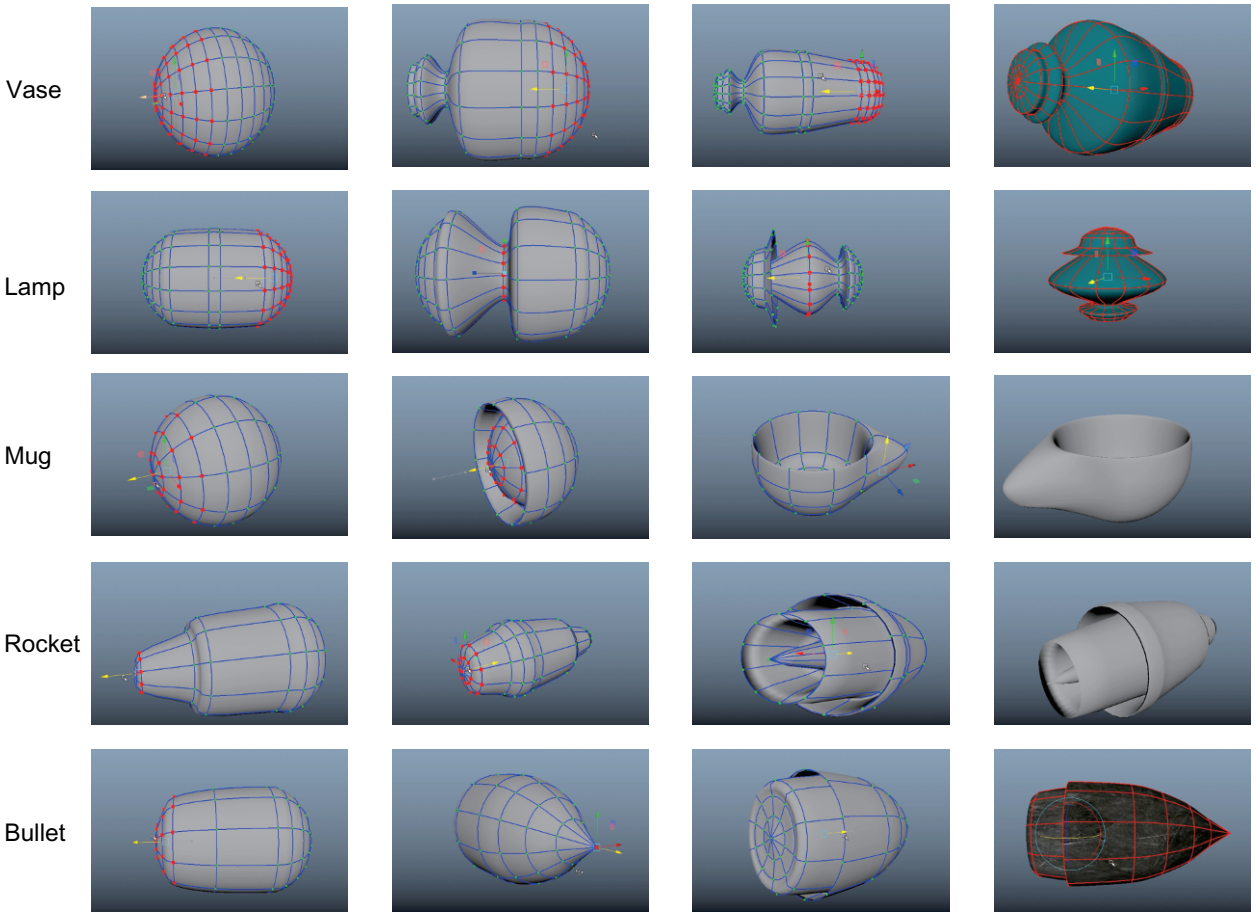

Fig. 5 Implementation of the framework in a shape modeling environment. Different shapes are interactively designed starting from a sphere (from left to right). The interpolatory control points allow us to easily model surfaces that can adopt shapes beyond traditional spherical topology, such as the mug, rocket, or bullet. The last two rows show shapes where only the closeness condition has been imposed in order to allow for the construction of sharp kinks.

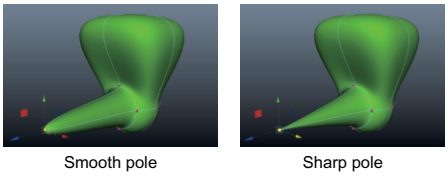

Fig. 6 Poles with continuously varying tangent plane. The effect of imposing the smoothness condition on the poles in interactive shape modeling is illustrated. Left: smooth pole. Right: sharp discontinuity at the pole resulting in a singularity.

Fig. 7 Interpolation of a parameterized point cloud. The dinosaurs (middle and right) are smooth reconstructions obtained by interpolating the point cloud on the left. Our surface construction is affine invariant and hence, rotating the shape is simply performed by rotating the point cloud.

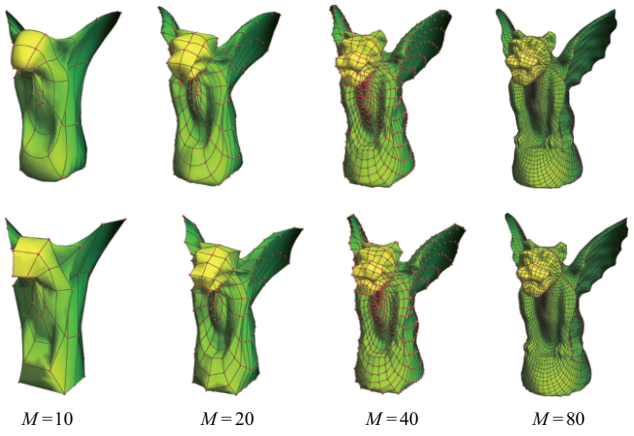

Fig. 8 Interpolation of shapes with spherical topology: smooth Gargoyle reconstructions at different resolutions. The same number of control points is used in both directions of the parameter domain, i.e., $M = M_1 = M_2$. Top: results obtained with our construction. Bottom: a (linear) polygon reconstruction method is applied. Note that with our approach the smoothness of the model does not depend on the number of parameters.

even with few control points, the tangent plane and Gaussian curvature are everywhere well-defined. This advantageous property allows construction of textured models with few parameters, which is useful for example in applications involving real-time rendering. As an example, we have parameterized the point cloud of the Gargoyle model using the algorithm described by Ref. [32], which allows us to reconstruct a smooth surface by interpolating the points. Additionally, we have subsampled the point cloud at different resolutions to obtain an approximation of the Gargoyle with varying levels of accuracy. Figure 8 illustrates the result and makes a comparison with a model based on polygons.

6.2.2 Compression

Related to the previous example is the problem of shape compression. Typically, the fewer coefficients are used to compress a smooth shape, the more discontinuous its representation becomes, which influences its texturing and rendering. The advantage of our model is that smoothness is always preserved, even with few coefficients, as shown in Fig. 8.

6.3 Efficient shape deformation

An advantage of using continuous-domain models based on control points is that the shapes are described by a finite number of control points, whereas the corresponding coordinate functions x, y, and z live in an infinite dimensional space; this allows us to describe a shape deformation process in the continuous domain just by displacing the control points. In the following, we provide two examples that illustrate how the minimization of distance criteria in the continuous domain can be efficiently formulated as conditions on the control points. Other deformation criteria which can be minimized in a similar way have been studied in Refs. [46, 47].

6.3.1 Minimum-energy deformation

We illustrate two deformation processes which correspond to minimum-energy deformation in $L_2([0,1]^2, \mathbb{R}^3)$. Both processes are formulated entirely with respect to the control points. Thereby, we can parameterize the path, which describes the deformation in the space that contains all parametric shapes. An immediate application is the construction of interpolated or extrapolated shapes, where the terms *interpolated* and *extrapolated* refer to a shape lying on the path in some *shape space*. Typically, such a shape space is described by a metric that provides a notion of distance between two points that lie in the space. Hence, in a given shape space, a shape is treated as a single point. Here we are interested in describing the deformation such that a minimum amount of *energy* is required in order to deform one shape into another. This translates into describing the deformation as the shortest path between two points in the shape space according to its underlying metric.

The Hilbert plane as a shape space. Given two surfaces $\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$ living in the Hilbert plane $L_2([0,1]^2, \mathbb{R}^3)$, the shortest path connecting them can be parameterized by the intermediate surface $\boldsymbol{\sigma}$ that minimizes:

$$\boldsymbol{F}(\tau, \boldsymbol{\sigma}) = \tau \|\boldsymbol{\sigma}_1 - \boldsymbol{\sigma}\|_{L_2}^2 + (1-\tau)\|\boldsymbol{\sigma}_2 - \boldsymbol{\sigma}\|_{L_2}^2 \quad (15)$$

for a given $\tau \in [0,1]$. We see immediately that, for $\tau = 0$, the minimizer is $\boldsymbol{\sigma} = \boldsymbol{\sigma}_2$, whereas for $\tau = 1$ it is $\boldsymbol{\sigma} = \boldsymbol{\sigma}_1$. For values of $\tau \in \mathbb{R} \setminus [0,1]$, the path $\boldsymbol{F}$ describes extrapolated shapes, i.e., shapes that do not lie between the two surfaces $\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$. The L_2-norm in Eq. (15) is induced by the L_2-inner product:

$$\langle \boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2 \rangle = \int_{\mathbb{R}} \int_{\mathbb{R}} \boldsymbol{\sigma}_1(u,v)^{\mathrm{T}} \boldsymbol{\sigma}_2(u,v)\, \mathrm{d}u\, \mathrm{d}v \quad (16)$$

Using the property that our parameterization is affine invariant, it is easy to show that the solution of $\min_{\boldsymbol{\sigma}} \boldsymbol{F}(\tau, \boldsymbol{\sigma})$ is given by

$$\boldsymbol{C}(\tau) = \tau \boldsymbol{C}_1 + (1-\tau)\boldsymbol{C}_2 \quad (17)$$

where $\boldsymbol{C}, \boldsymbol{C}_1, \boldsymbol{C}_2$ are the matrices which contain all the control points of the corresponding surfaces. As an example that illustrates the deformation process and also the effect of imposing the closeness-condition on the poles, we have deformed a disk into a sphere. Figure 9 illustrates this process and compares it to the case where no pole-interpolation conditions are imposed. Figure 1 shows the deformation of a sphere into a Gargoyle.

The Hilbert sphere as a shape space. Every parametric shape can be projected onto the unit Hilbert sphere by normalizing it such that $\|\boldsymbol{\sigma}\|_{L_2} = \sqrt{\langle \boldsymbol{\sigma}, \boldsymbol{\sigma} \rangle} = 1$. The shortest distance between two points $\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$ on the sphere lies on the great circle that passes through them. A possible parameterization of this great circle is

$$\boldsymbol{\Gamma}(\tau) = \frac{1}{\sin(\theta)}[\sin(\theta(1-\tau))\boldsymbol{\sigma}_1 + \sin(\theta\tau)\boldsymbol{\sigma}_2] \quad (18)$$

where $\theta = \cos^{-1}(\langle \boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2 \rangle)$, $\boldsymbol{\Gamma}(0) = \boldsymbol{\sigma}_1$, and $\boldsymbol{\Gamma}(1) = \boldsymbol{\sigma}_2$. Again, if $\tau \in [0,1]$, Eq. (18) describes interpolated shapes, whereas for $\tau \in \mathbb{R} \setminus [0,1]$, τ describes extrapolated shapes. As in the previous example, we exploit the affine invariance of our parameterization in order to describe the deformation as a function of the control points. The interpolating control points are given by

$$\boldsymbol{C}(\tau) = \frac{1}{\sin(\theta)}[\sin(\theta(1-\tau))\boldsymbol{C}_1 + \sin(\theta\tau)\boldsymbol{C}_2] \quad (19)$$

An example invoking this deformation is shown in Fig. 10.

Morphing. Computing morphs between two or several shapes is similar to computing the deformation between shapes. The difference is that the deformation is expressed as a parameterized weighted linear combination of two shapes, whereas a morph corresponds to a particular instance of the parameterized function. Concretely, if Eq. (17) or Eq. (19) is evaluated for a specific value of τ, we obtain a morph between $\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$. Examples of such smooth morph constructions are shown in Fig. 10, which correspond to morphed point clouds similar to the ones shown in Fig. 11.

Parameterization. An important aspect to consider when using our model is that the parameterization which describes the shape is not unique. This is natural in the case of surfaces with spherical topology and originates from the

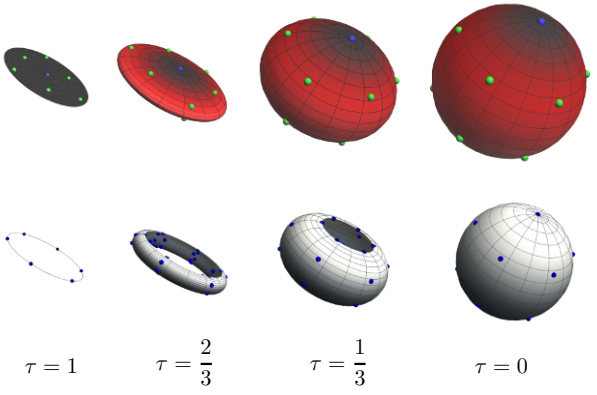

Fig. 9 Minimum-energy deformation in the Hilbert plane. Top: a disk is deformed into a sphere through Eq. (17). Bottom: the same process, but without imposing the pole-interpolation conditions in Eq. (14). In this case, the surface does not remain closed when deforming and Eq. (17) describes the deformation between a circle and a sphere.

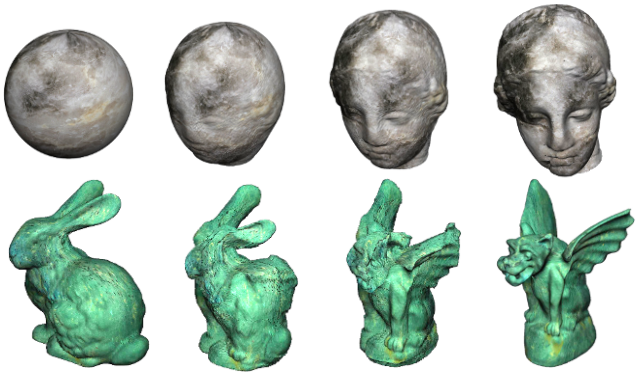

Fig. 10 Minimum-energy deformation on the Hilbert sphere. Top: sphere and Venus. Bottom: Stanford Bunny and Gargoyle. Both have been mapped through normalization onto the Hilbert sphere. Deformation is then described by Eq. (19), for values $\tau = 0, \frac{1}{3}, \frac{2}{3}, 1$ from left to right.

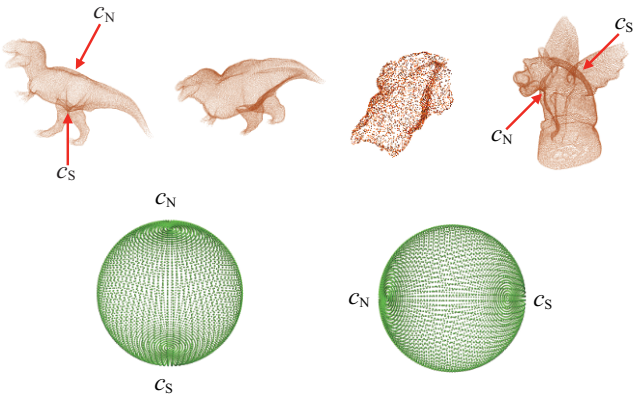

Fig. 11 Influence of the parameterization on the deformation. The point clouds (with $M = M_1 = M_2 = 270$) that define the control points of the dinosaur and the Gargoyle are parameterized and the locations of the poles are indicated with red arrows. The dinosaur (top left) is deformed into the Gargoyle (top right). The two intermediate shapes in the top row illustrate the deformation. Bottom: the poles on the sphere can be placed at different locations. For instance, if a different parameterization of the dinosaur is chosen such that the the North pole $\boldsymbol{c}_N$ and South pole $\boldsymbol{c}_S$ are exchanged, then the deformation process is different.

fact that there exist an infinite number of ways to place the two poles (with the constraint that they must be opposite to each other). However, considering Eqs. (17) and (19), we see that there is a unique correspondence between two given spherical parameterizations, which implies that given two surfaces, $\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$, each control point $\boldsymbol{c}_1[k_0, l_0]$ is transformed into $\boldsymbol{c}_2[k_0, l_0]$. If a different parameterization is chosen for at least one of the two surfaces (i.e., if the poles are placed differently), then the resulting deformation will inevitably be different. This is illustrated in Fig. 11. Insights into finding an optimal correspondence between shapes can be found in Refs. [48] and [49].

6.4 Fast computation of surface and volume integrals

In certain applications that require iterative optimization, it is necessary to rapidly compute surface or volume integrals efficiently. An example is the deformation of a surface guided by optimizing an energy functional in real time.

6.4.1 Flux across surface

We illustrate how a flux E across a surface S, parameterized by $\boldsymbol{\sigma}(u, v)$, may be computed rapidly and efficiently. Given a vector field $\boldsymbol{f}$, one way of expressing the flux E is by

$$E(\boldsymbol{\sigma}) = \iint_S \cdot \mathrm{d}\boldsymbol{S} = \int_0^1 \int_0^1 g^x(\boldsymbol{\sigma})\, \mathrm{d}y \wedge \mathrm{d}z \qquad (20)$$

where $\mathrm{d}\boldsymbol{S}$ represents the vector differential of the surface area, $\wedge$ denotes the wedge product, and $g^x(x, y, z) = \int_{-\infty}^{x} \mathrm{div}\boldsymbol{f}(\tau, y, z)\, \mathrm{d}\tau$ is the pre-integrated divergence of the vector field $\boldsymbol{f}$ in the x-direction. Typically, $\boldsymbol{f}$ does not depend on the surface and hence, g^x can be precomputed and stored in a look-up table to significantly speed up the computation. We derive Eq. (20) in Appendix D. The use of pre-integrated functions is only possible because we define the surface $\boldsymbol{\sigma}$ in the continuous domain. Next, the flux E can be efficiently optimized by computing the gradient of E with respect to the control points using a gradient-based iterative method. An explicit expression of the gradient can be obtained easily, and hence implemented in an exact way.

Example. We illustrate the above computation by segmenting the surface of a human brain in a 3D MRI image. We first compute an edge map of the 3D image using a standard surface extraction algorithm [50] and construct an energy functional E that depends on the gradient of the edgemap. Hence, in Eq. (20), the gradient becomes $\boldsymbol{f}$. By minimizing Eq. (20), $\boldsymbol{\sigma}$ deforms iteratively in order to approximate the edge map, as shown in Fig. 12. The result can easily be manually adjusted by a clinician (see Fig. 4), which is an additional advantage of our algorithm compared to existing methods.

6.4.2 Exact volume computation

For $k_i \in [0, \ldots, M_1 - 1]$ and $l_i \in [-1, \ldots, M_2 + 1]$ the volume enclosed by the surface $\boldsymbol{\sigma}$ is computed by

$$\mathrm{Vol}(\boldsymbol{\sigma}) = \sum_{\substack{k_1,k_2,k_3 \\ l_1,l_2,l_3}} c_x[k_1, l_1] c_y[k_2, l_2] c_z[k_3, l_3] \times \alpha(k_1, k_2, k_3, l_1, l_2, l_3) \qquad (21)$$

where c_x, c_y, and c_z are the x, y, and z coordinates of the control points of $\boldsymbol{\sigma}$ and

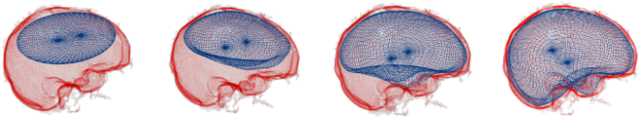

Fig. 12 Brain segmentation in a 3D medical MRI image. The red surface is a rendered edge map extracted from medical data. An ellipsoidal surface is initialized inside the brain surface (left) and evolves by iteratively minimizing (20) (from left to right). The final result (right) corresponds to a smooth and continuous closed surface shape.

$$\begin{aligned}&\alpha(k_1,k_2,k_3,l_1,l_2,l_3)\\&= M_1M_2\Big(\int_0^1 \phi_{1,k_1}\phi'_{1,k_2}\phi_{1,k_3}\mathrm{d}u\int_0^1 \phi_{2,l_1}\phi_{2,l_2}\phi'_{2,l_3}\mathrm{d}v\\&-\int_0^1 \phi_{1,k_1}\phi_{1,k_2}\phi'_{1,k_3}\mathrm{d}u\int_0^1 \phi_{2,l_1}\phi'_{2,l_2}\phi_{2,l_3}\mathrm{d}v\Big)\end{aligned}\tag{22}$$

Since α does not depend on the control points $\boldsymbol{c}$, it can be precomputed and stored in a look-up table in order to quickly evaluate the volume in interactive optimization schemes. Furthermore, because ϕ and its derivative ϕ' have compact support, the number of non-zero elements in Sum (21) is small, which additionally simplifies the computation. We derive the formula for the volume in Appendix E. The integrals in Eq. (22) can be further simplified and exactly evaluated using techniques from spline theory similar to the approaches in Refs. [51, 52].

7 Implementation

In this section, we describe some important details regarding the implementation.

7.1 Choice of free parameters

7.1.1 Exact sphere

The orientation of the sphere is given by Eq. (4) and therefore, the coordinates of the North pole are $\boldsymbol{c}_{\mathrm{N}} = (0,0,1)$, and for the South pole, $\boldsymbol{c}_{\mathrm{S}} = (0,0,-1)$. Since by construction, the vectors $\boldsymbol{T}_{1,\mathrm{N}}$ and $\boldsymbol{T}_{2,\mathrm{N}}$ span the tangent plane at the North pole of the sphere, a natural choice is to set $\boldsymbol{T}_{1,\mathrm{N}} = (1,0,1) - \boldsymbol{c}_{\mathrm{N}} = (1,0,0)$ and $\boldsymbol{T}_{2,\mathrm{N}} = (0,1,1) - \boldsymbol{c}_{\mathrm{N}} = (0,1,0)$. With the same approach we also obtain $\boldsymbol{T}_{1,\mathrm{S}} = (1,0,0)$ and $\boldsymbol{T}_{2,\mathrm{S}} = (0,1,0)$ for the South tangent plane. Note that because our construction is affine invariant, for a sphere with a different size or orientation, the new coordinates are found by applying the corresponding affine transformation to the existing control points.

7.1.2 Arbitrary shape with spherical topology

A simple method to estimate the tangent plane is to compute the plane that best approximates the points lying on the first circle of latitude next to the North or South pole. Any two vectors spanning this plane can be chosen as $\boldsymbol{T}_{1,\mathrm{N}}$, $\boldsymbol{T}_{2,\mathrm{N}}$ and $\boldsymbol{T}_{1,\mathrm{S}}$, $\boldsymbol{T}_{2,\mathrm{S}}$.

7.2 Discretization of basis functions

Because our construction is formulated in the continuous domain, the shape representation can be discretized with arbitrary precision in order to implement it. An efficient way is to discretize the interpolator φ rather than the surface, which becomes highly beneficial, for example, in interactive applications where the shapes to be constructed are not known beforehand. By discretizing φ prior to surface construction, the samples of the interpolator can be stored in a look-up table to speed up the surface reconstruction. Thus, the sampling rate is freely chosen. In Figs. 13 and 14, we show the effects of different sampling rates. A sampling rate equal to one corresponds to a polygon model (i.e., linear interpolation between points), which means only the blue sample in Fig. 13 is non-zero. The higher the sampling rate, the closer the approximation of the continuous domain model. Its effect on surface reconstruction is shown in Fig. 14. In practice, if a large number of control points is used, one can already obtain satisfactory smoothness of the surface with a low sampling rate, whereas with a small number of control points, the sampling rate must be higher to obtain a smooth surface.

8 Discussion and future work

8.1 Comparison with NURBS

Our formulation has several advantages compared to an approach using NURBS. With the NURBS formulation, a sphere can only be represented using multiple surface patches. The NURBS formulation requires not only more control points to represent a sphere, but also more parameters in total, due to the weights used in that formulation. Further,

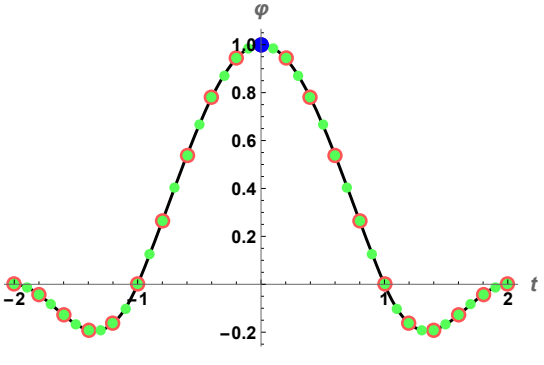

Fig. 13 Sampling of the interpolator φ. Because φ is formulated in the continuous domain, it can be discretized with arbitrary precision. If only one sample is considered (blue sample), then the result corresponds to linear interpolation, which is equivalent to a polygon model. The samples denoted by orange circles correspond to a lower sampling rate than the green samples.

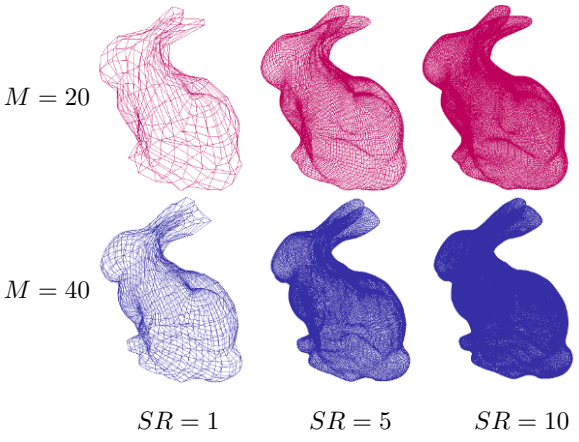

Fig. 14 Effects of different sampling rates, increased from left to right. The surfaces are constructed with $M = M_1 = M_2$. The red wireframe corresponds to a lower number of control points ($M = 20$) used to reconstruct the bunny than the blue wireframe ($M = 40$).

because the basis functions of NURBS are rational, the computation of derivatives and integrals results in complicated expressions. This can become a problem when integral-dependent quantities need to be computed, such as in the evaluation of surface or volume integrals in optimization schemes. Also, the optimization itself must be carried out simultaneously with respect to the control points and the weight parameters, which introduces additional complexity. For interactive shape design, the interpolation property of our framework makes the modeling task more intuitive. Complex shapes that require more detail, and hence, more control points, are especially modeled more easily with our solution (see Fig. 4). Furthermore, interpolating parameterized point clouds with spherical topology is difficult with NURBS due to their non-interpolatory nature; it involves complex NURBS approximation techniques or inverse filtering, which is not straightforward because of the smoothness conditions at the poles. The only NURBS that are interpolatory are zero and first degree NURBS, which are non-smooth.

Moreover, our formulation allows for a shape representation using only integer shifts. NURBS usually have non-uniform shifts. The advantage of considering integer shifts is that it allows convolution and filtering techniques as well as frequency domain calculus. This can be useful when performing surface resampling, projections onto other spline spaces, and evaluation of inner products, for example to compute L_2-distances between surfaces. It also allows for a simpler formulation of the surface by specifying control points instead of non-uniform knot vectors including double knots.

8.2 Comparison with polygon models

Polygon models are inherently interpolatory schemes because the control points coincide with the vertices of the mesh. Similar to subdivision schemes, these models require more parameters than our model in order to achieve a higher degree of smoothness (see Fig. 8). Geometric operators and quantities, such as tangent planes, normals, curvatures, or the Laplacian have to be approximated by polygon mesh processing techniques. The same holds true for integral and derivative-based quantities. However, polygon meshes do not require an underlying parameterization of the model.

8.3 Comparison with the Catmull–Rom interpolator

The Catmull–Rom [36] or Keys interpolator [37] are interpolating and smooth. Because they are polynomial it is not possible to construct an exact sphere with these functions. However, construction of a model with spherical topology (which excludes exact spheres and ellipsoids) is possible by replacing φ in our framework with the Catmull–Rom spline. Because its support is the same as for φ, our formulation for the smoothness and interpolation conditions at the poles can easily be adapted to the purely polynomial case.

8.4 Extending the framework to other topologies

Our concept can be extended to surfaces with other topologies (e.g., cylindrical or rectangular) in order to create a unifying framework for smooth shape modeling with interpolatory control points. These topologies do not require special attention to poles and are easier to parameterize using tensor products and a suitable interpolator. One way to parameterize the rectangle is with the polynomial Keys interpolator [37], whereas the cylinder may be parameterized using φ for the trigonometric part (i.e., the circles in one direction) and the Keys interpolator for the linear part (i.e., along the axis). Another example is the torus which is easily parameterized using φ since it is periodic in u and v. In Fig. 15, examples of these topologies are shown as well as how they can be smoothly deformed by

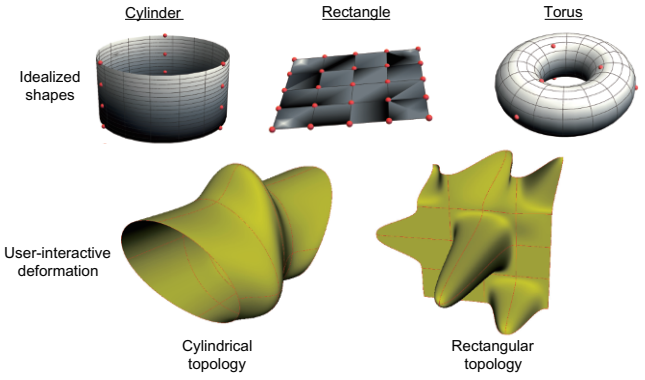

Fig. 15 Smooth modeling of different topologies with interpolatory control points. Top: idealized shapes that define the topologies. The red points indicate the interpolatory control points. Bottom: a smooth deformation of these shapes.

exploiting the interpolation property in interactive settings.

9 Conclusions

The standard method for smooth, parametric shape modeling in industry is NURBS. In this paper, we presented an alternative method to model smooth shapes with spherical topology. The fundamental difference with the existing standard is that our basis functions are interpolatory and non-rational and we only use uniform shifts. Our formulation is simpler than NURBS and thus has several advantages in practical applications, including immediate reconstruction of smooth surfaces by interpolating parameterized point clouds, more intuitive shape modeling, and simplified formulation of optimization algorithms that involve integral- and derivative-dependent quantities. Our framework is extensible to a richer family of topologies. A video illustrating the use of our framework in practice is available at http://bigwww.epfl.ch/demo/siggraph2016/.

Appendix A Explicit expression for φ

The generator is given by $\varphi = \beta * \psi$, where $*$ denotes continuous convolution. The function β is a third-order exponential B-spline defined by

$$\beta(t) = \begin{cases} \frac{M^2 \sin^2\left(\frac{\pi t}{M}\right)}{2\pi^2}, & 0 < t \leqslant 1 \\ \frac{M^2\left(\cos\left(\frac{2\pi(t-2)}{M}\right)+\cos\left(\frac{2\pi(t-1)}{M}\right)-2\cos\left(\frac{2\pi}{M}\right)\right)}{4\pi^2}, & 1 < t \leqslant 2 \\ \frac{M^2 \sin^2\left(\frac{\pi(t-3)}{M}\right)}{2\pi^2}, & 2 < t < 3 \\ 0, & \text{else} \end{cases}$$

and

$$\psi(t) = \gamma_1(M)\beta^0(t+2) + \gamma_2(M)(\delta(t+2)+\delta(t+1))$$

is a smoothing kernel with

$$\beta^0(t) = \begin{cases} 1, & 0 < t < 1 \\ 0, & \text{else} \end{cases}$$

being the zero degree polynomial B-spline, δ the Dirac function. Further

$$\gamma_1(M) = \frac{\pi^3 \sec^2\left(\frac{\pi}{M}\right)}{M^2\left(M\tan\left(\frac{\pi}{M}\right)-\pi\right)}$$

and

$$\gamma_2(M) = \frac{\pi^2 \csc\left(\frac{\pi}{M}\right)\csc\left(\frac{2\pi}{M}\right)\left(M-2\pi\csc\left(\frac{2\pi}{M}\right)\right)}{M^2\left(M\sec\left(\frac{\pi}{M}\right)-\pi\csc\left(\frac{\pi}{M}\right)\right)}$$

are constants that depend only on M. The explicit expression for $\varphi = \varphi_M$ is given by Eq. (23). Note that φ is non-rational with respect to its parameter.

Appendix B Smoothness conditions at poles

The left-hand-side of Eq. (9) is developed as

$$\left.\frac{\partial \boldsymbol{\sigma}(u,v)}{v}\right|_{v=0} =$$

$$M_2 \sum_{k=0}^{M_1-1}\sum_{l=-1}^{M_2+1} \boldsymbol{c}[k,l]\varphi_{M_1,\mathrm{per}}(M_1u-k)\varphi'_{2M_2}(M_2v-l)\Big|_{v=0}$$

$$= M_2 \sum_{k=0}^{M_1-1} \varphi_{M_1,\mathrm{per}}(M_1u-k) \sum_{l=-1}^{M_2+1} \boldsymbol{c}[k,l]\varphi'_{2M_2}(-l)$$

$$= M_2 \sum_{k=0}^{M_1-1} \varphi_{M_1,\mathrm{per}}(M_1u-k)$$

$$\varphi_M(t) = \begin{cases} \frac{\sin^2\left(\frac{\pi}{M}\right)\left(M\sin\left(\frac{2\pi(t-2)}{M}\right)-2\pi(t-2)\right)+\left(M\sin\left(\frac{2\pi}{M}\right)-2\pi\right)\sin^2\left(\frac{\pi(t-2)}{M}\right)}{2\left(\cos\left(\frac{2\pi}{M}\right)-1\right)\left(-M\sin\left(\frac{2\pi}{M}\right)+\pi\cos\left(\frac{2\pi}{M}\right)+\pi\right)}, & 1 < t < 2 \\ \frac{M\left(\sin\left(\frac{2\pi(t-2)}{M}\right)-2\sin\left(\frac{2\pi(t-1)}{M}\right)+\sin\left(\frac{2\pi(t+1)}{M}\right)+\sin\left(\frac{2\pi}{M}\right)-\sin\left(\frac{4\pi}{M}\right)\right)+2\pi t\cos\left(\frac{2\pi}{M}\right)-2\pi(t-1)\cos\left(\frac{4\pi}{M}\right)-2\pi\cos\left(\frac{2\pi t}{M}\right)}{4\left(\cos\left(\frac{2\pi}{M}\right)-1\right)\left(-M\sin\left(\frac{2\pi}{M}\right)+\pi\cos\left(\frac{2\pi}{M}\right)+\pi\right)}, & 0 < t \leqslant 1 \\ -\frac{M\left(\sin\left(\frac{2\pi(t-1)}{M}\right)-2\sin\left(\frac{2\pi(t+1)}{M}\right)+\sin\left(\frac{2\pi(t+2)}{M}\right)-\sin\left(\frac{2\pi}{M}\right)+\sin\left(\frac{4\pi}{M}\right)\right)+2\pi\left(t\cos\left(\frac{2\pi}{M}\right)-(t+1)\cos\left(\frac{4\pi}{M}\right)+\cos\left(\frac{2\pi t}{M}\right)\right)}{4\left(\cos\left(\frac{2\pi}{M}\right)-1\right)\left(-M\sin\left(\frac{2\pi}{M}\right)+\pi\cos\left(\frac{2\pi}{M}\right)+\pi\right)}, & -1 < t \leqslant 0 \\ \frac{\sin^2\left(\frac{\pi}{M}\right)\left(2\pi(t+2)-M\sin\left(\frac{2\pi(t+2)}{M}\right)\right)+\left(M\sin\left(\frac{2\pi}{M}\right)-2\pi\right)\sin^2\left(\frac{\pi(t+2)}{M}\right)}{2\left(\cos\left(\frac{2\pi}{M}\right)-1\right)\left(-M\sin\left(\frac{2\pi}{M}\right)+\pi\cos\left(\frac{2\pi}{M}\right)+\pi\right)}, & -2 < t \leqslant -1 \\ 0, & |t| \geqslant 2 \end{cases} \quad (23)$$

$$\times\Big(\boldsymbol{c}[k,-1]\varphi'_{2M_2}(1)+\boldsymbol{c}[k,0]\varphi'_{2M_2}(0)+\boldsymbol{c}[k,1]\varphi'_{2M_2}(-1)\Big)$$
$$= M_2 \sum_{k=0}^{M_1-1} \varphi_{M_1,\mathrm{per}}(M_1u-k) \times \left(\varphi'_{2M_2}(-1)(\boldsymbol{c}[k,-1]-\boldsymbol{c}[k,1])\right) \tag{24}$$

where we have used the fact that φ' is odd (φ is even). The right-hand-side of Eq. (9) may be expressed as

$$\begin{aligned}&\boldsymbol{T}_{1,\mathrm{N}}\cos(2\pi u)+\boldsymbol{T}_{2,\mathrm{N}}\sin(2\pi u)\\ &=\boldsymbol{T}_{1,\mathrm{N}}\sum_{k=0}^{M_1-1}\cos\left(\frac{2\pi k}{M_1}\right)\varphi_{M_1,\mathrm{per}}(M_1u-k)\\ &\quad+\boldsymbol{T}_{2,\mathrm{N}}\sum_{k=0}^{M_1-1}\sin\left(\frac{2\pi k}{M_1}\right)\varphi_{M_1,\mathrm{per}}(M_1u-k)\\ &=\sum_{k=0}^{M_1-1}\left(\boldsymbol{T}_{1,\mathrm{N}}\cos\left(\frac{2\pi k}{M_1}\right)+\boldsymbol{T}_{2,\mathrm{N}}\sin\left(\frac{2\pi k}{M_1}\right)\right)\\ &\quad\times\varphi_{M_1,\mathrm{per}}(M_1u-k)\end{aligned} \tag{25}$$

By equating Eqs. (24) and (25) and identifying the coefficients, we obtain Eqs. (11) and (12).

Appendix C Interpolation conditions at poles

At the North pole, we have:

$$\boldsymbol{\sigma}(u,0)=\sum_{k=0}^{M_1-1}\sum_{l=-1}^{M_2+1}\boldsymbol{c}[k,l]\varphi_{M_1,\mathrm{per}}(M_1u-k)\varphi_{2M_2}(-l) \tag{26}$$

Since φ_M satisfies the interpolation condition, the term that depends on l is always zero unless $l=0 \Leftrightarrow \varphi_{2M_2}(l=0)=1$. Hence, Eq. (26) simplifies to

$$\boldsymbol{\sigma}(u,0)=\sum_{k=0}^{M_1-1}\boldsymbol{c}[k,0]\varphi_{M_1,\mathrm{per}}(M_1u-k):=\boldsymbol{c}_{\mathrm{N}} \tag{27}$$

Because the integer shifts of φ_{M_1} build a basis [39] and φ_{M_1} satisfies the partition of unity property, Eq. (27) only holds if $\boldsymbol{c}[k,0]=\boldsymbol{C}$, with $\boldsymbol{C}$ a constant vector for all k. Thus,

$$\begin{aligned}\boldsymbol{\sigma}(u,0)&=\sum_{k=0}^{M_1-1}\boldsymbol{C}\varphi_{M_1,\mathrm{per}}(M_1u-k)\\ &=\boldsymbol{C}\sum_{k=0}^{M_1-1}\underbrace{\varphi_{M_1,\mathrm{per}}(M_1u-k)}_{=1}\\ &=\boldsymbol{C}\\ &=\boldsymbol{c}[k,0]=\boldsymbol{c}_{\mathrm{N}}\quad\forall k\in[0,\dots,M_1-1]\end{aligned}$$

A similar derivation leads to the interpolation condition at the South pole.

Appendix D Flux across surface

We denote by $\boldsymbol{n}$ the normal vector to the surface and make use of the divergence theorem to compute:

$$\begin{aligned}E(\boldsymbol{\sigma})&=\iint_S\boldsymbol{f}\cdot\mathrm{d}\boldsymbol{S}=\iint_S\left(\boldsymbol{f}\cdot\frac{\boldsymbol{n}}{\|n\|}\right)\mathrm{d}S\\ &=\iiint_V\underbrace{\mathrm{div}\boldsymbol{f}}_{g}\,\mathrm{d}V=\iint_{\partial V=S}g^x\,\mathrm{d}y\wedge\mathrm{d}z\\ &=\iint_{\partial V=S}g^y\,\mathrm{d}x\wedge\mathrm{d}z=\iint_{\partial V=S}g^z\,\mathrm{d}x\wedge\mathrm{d}y\end{aligned}$$

where g^x, g^y, g^z are the pre-integrated functions along directions x, y, or z. The wedge operator is defined as

$$\mathrm{d}y\wedge dz=\frac{\partial y}{\partial u}\frac{\partial z}{\partial v}-\frac{\partial y}{\partial v}\frac{\partial z}{\partial u} \tag{28}$$

and is explicitly computed using $\partial\boldsymbol{\sigma}/\partial u=(\partial x/\partial u, \partial y/\partial u, \partial z/\partial u)$ and $\partial\boldsymbol{\sigma}/\partial v=(\partial x/\partial v, \partial y/\partial v, \partial z/\partial v)$ with

$$\frac{\partial\boldsymbol{\sigma}}{\partial u}(u,v)= M_1\sum_{k=0}^{M_1-1}\sum_{l=-1}^{M_2+1}\boldsymbol{c}[k,l]\varphi'_{M_1,\mathrm{per}}(M_1u-k)\varphi_{2M_2}(M_2v-l) \tag{29}$$

and

$$\frac{\partial\boldsymbol{\sigma}}{\partial v}(u,v)= M_2\sum_{k=0}^{M_1-1}\sum_{l=-1}^{M_2+1}\boldsymbol{c}[k,l]\varphi_{M_1,\mathrm{per}}(M_1u-k)\varphi'_{2M_2}(M_2v-l) \tag{30}$$

Appendix E Volume computation

By the divergence theorem, the volume of a parametric surface is given by

$$\mathrm{Vol}(\boldsymbol{\sigma})=\iint_S(x,0,0)\cdot\boldsymbol{n}\,\mathrm{d}S=\iint_S x\,\mathrm{d}y\wedge\mathrm{d}z \tag{31}$$

By applying the same simplifications to compute the wedge operator (28) as in Appendix D, and using

$$x(u,v)=\sum_{k=0}^{M_1-1}\sum_{l=-1}^{M_2+1}\boldsymbol{c}[k,l]\varphi_{M_1,\mathrm{per}}(M_1u-k)\varphi_{2M_2}(M_2v-l)$$

the volume computation simplifies to

$$\mathrm{Vol}(\boldsymbol{\sigma})=\int_0^1\int_0^1 x(u,v)\cdot\left(\frac{\partial y(u,v)}{\partial u}\frac{\partial z(u,v)}{\partial v}-\frac{\partial y(u,v)}{\partial v}\frac{\partial z(u,v)}{\partial u}\right)\mathrm{d}u\mathrm{d}v$$

Because only the basis functions depend on u and v, the integral-dependent terms can be isolated and precomputed to obtain Eq. (21).

Acknowledgements

This work was funded by the Swiss National Science Foundation under Grant 200020-162343. We are grateful to Zsuzsanna Püspöki for help with the figures and to Irina Radu for help with the video. We also appreciate the interesting discussions on the subject that we had with Masih Nilchian and Emrah Bostan. We thank Mike McCann for proof-reading the manuscript.

References

[1] Schmitter, D.; García-Amorena, P.; Unser, M. Smoothly deformable spheres: Modeling, deformation, and interaction. In: Proceedings of the SIGGRAPH ASIA 2016 Technical Briefs, Article No. 2, 2016.

[2] Piegl, L. On NURBS: A survey. *IEEE Computer Graphics and Applications* Vol. 11, No. 1, 55–71, 1991.

[3] Piegl, L.; Tiller, W. *The NURBS Book*, 2nd edn. Springer Berlin Heidelberg, 2010.

[4] Sederberg, T. W.; Zheng, J.; Bakenov, A.; Nasri, A. T splines and T NURCCS. *ACM Transactions on Graphics* Vol. 22, No. 3, 477–484, 2003.

[5] Lyche, T. Discrete cubic spline interpolation. *BIT Numerical Mathematics* Vol. 16, No. 3, 281–290, 1976.

[6] Lyche, T. Local spline approximation methods. *Journal of Approximation Theory* Vol. 15, No. 4, 294–325, 1975.

[7] Manni, C.; Pelosi, F.; Sampoli, L. Generalized B splines as a tool in isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering* Vol. 200, Nos. 5–8, 867–881, 2011.

[8] Schumaker, L. L. *Spline Functions: Basic Theory*. J. Wiley & Sons, 1981.

[9] Farin, G. E. *NURBS: From Projective Geometry to Practical Use*, 2nd edn. Natick, MA, USA: A. K. Peters, Ltd., 1999.

[10] Rogers, D. F. Preface. In: *An Introduction to fNURBSg, The Morgan Kaufmann Series in Computer Graphics*. Rogers, D. F. Ed. San Francisco: Morgan Kaufmann, xv–xvii, 2001.

[11] Dierckx, P. Algorithms for smoothing data on the sphere with tensor product splines. *Computing* Vol. 32, No. 4, 319–342, 1984.

[12] Gmelig Meyling, R. H. J.; Pfluger, P. R. B spline approximation of a closed surface. *IMA Journal of Numerical Analysis* Vol. 7, No. 1, 73–96, 1987.

[13] Delgado Gonzalo, R.; Chenouard, N.; Unser, M. Spline based deforming ellipsoids for interactive 3D bioimage segmentation. *IEEE Transactions on Image Processing* Vol. 22, No. 10, 3926–3940, 2013.

[14] Prautzsch, H.; Boehm, W.; Paluszny, M. *Bézier and B Spline Techniques*. Springer Verlag New York, 2002.

[15] Romani, L.; Sabin, M. A. The conversion matrix between uniform B spline and Bézier representations. *Computer Aided Geometric Design* Vol. 21, No. 6, 549–560, 2004.

[16] Botsch, M.; Kobbelt, L.; Pauly, M.; Alliez, P.; Lévy, B. *Polygon Mesh Processing*. CRC Press, 2010.

[17] Botsch, M.; Steinberg, S.; Bischoff, S.; Kobbelt, L. OpenMesh–A generic and efficient polygon mesh data structure. In: Proceedings of the OpenSG Symposium, 2002.

[18] Sorkine, O. Differential representations for mesh processing. *Computer Graphics Forum* Vol. 25, No. 4, 789–807, 2006.

[19] Sorkine, O.; Cohen-Or, D.; Lipman, Y.; Alexa, M.; Rössl, C.; Seidel, H. P. Laplacian surface editing. In: Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, 175–184, 2004.

[20] Dyn, N.; Farkhi, E. Spline subdivision schemes for compact sets. A survey. *Serdica Mathematical Journal* Vol. 28, No. 4, 349–360, 2002.

[21] Lounsbery, M.; DeRose, T. D.; Warren, J. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Transactions on Graphics* Vol. 16, No. 1, 34–73, 1997.

[22] Catmull, E.; Clark, J. Recursively generated B spline surfaces on arbitrary topological meshes. In: *Seminal Graphics: Poineering Efforts that Shaped the Field*. New York: ACM, 183–188, 1998.

[23] DeRose, T.; Kass, M.; Truong, T. Subdivision surfaces in character animation. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, 85–94, 1998.

[24] Doo, D.; Sabin, M. Behaviour of recursive division surfaces near extraordinary points. *Computer-Aided Design* Vol. 10, No. 6, 356–360, 1978.

[25] Loop, C.; Schaefer, S. Approximating Catmull–Clark subdivision surfaces with bicubic patches. *ACM Transactions on Graphics* Vol. 27, No. 1, Article No. 8, 2008.

[26] Stam, J.; Loop, C. T. Quad/triangle subdivision. *Computer Graphics Forum* Vol. 22, No. 1, 79–85, 2003.

[27] Charina, M.; Conti, C.; Romani, L. Reproduction of exponential polynomials by multivariate non stationary subdivision schemes with a general dilation matrix. *Numerische Mathematik* Vol. 127, No. 2, 223–254, 2014.

[28] Novara, P.; Romani, L. Building blocks for designing arbitrarily smooth subdivision schemes with conic precision. *Journal of Computational and Applied Mathematics* Vol. 279, 67–79, 2015.

[29] Warren, J.; Weime, H. *Subdivision Methods for Geometric Design: A Constructive Approach*. San Francisco: Morgan Kaufmann, 2001.

[30] Hughes, T. J. R. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Courier Corporation, 2012.

[31] Asirvatham, A.; Praun, E.; Hoppe, H. Consistent spherical parameterization. In: *Computational Science–ICCS 2005*. Sunderam, V. S.; van Albada, G. D.; Sloot, P. M.; Dongarra, J. Eds. Springer Berlin Heidelberg, 265–272, 2005.

[32] Praun, E.; Hoppe, H. Spherical parametrization and remeshing. *ACM Transactions on Graphics* Vol. 22, No. 3, 340–349, 2003.

[33] Brechbühler, Ch.; Gerig, G.; Kübler, O. Parametrization of closed surfaces for 3D shape description. *Computer Vision and Image Understanding* Vol. 61, No. 2, 154–170, 1995.

[34] Berger, M.; Tagliasacchi, A.; Seversky, L.; Alliez, P.; Levine, J.; Sharf, A.; Silva, C. State of the art in surface reconstruction from point clouds. In: *Eurographics 2014–State of the Art Reports*. Lefebvre, S.; Spagnuolo, M. Eds. The Eurographics Association, 161–185, 2014.

[35] Kazhdan, M.; Bolitho, M.; Hoppe, H. Poisson surface reconstruction. In: Proceedings of the 4th Eurographics Symposium on Geometry Processing, 61–70, 2006.

[36] Catmull, E.; Rom, R. A class of local interpolating splines. *Computer Aided Geometric Design* Vol. 74, 317–326, 1974.

[37] Keys, R. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing* Vol. 29, No. 6, 1153–1160, 1981.

[38] Schmitter, D.; Delgado-Gonzalo, R.; Unser, M. A family of smooth and interpolatory basis functions for parametric curve and surface representation. *Applied Mathematics and Computation* Vol. 272, No. 1, 53–63, 2016.

[39] Schmitter, D.; Delgado-Gonzalo, R.; Unser, M. Trigonometric interpolation kernel to construct deformable shapes for user interactive applications. *IEEE Signal Processing Letters* Vol. 22, No. 11, 2097–2101, 2015.

[40] Beccari, C. V.; Casciola, G.; Roman, L. Construction and characterization of non uniform local interpolating polynomial splines. *Journal of Computational Applied Mathematics* Vol. 240, 5–19, 2013.

[41] Schmitter, D.; Gaudet-Blavignac, C.; Piccini, D.; Unser, M. New parametric 3D snake for medical segmentation of structures with cylindrical topology. In: Proceedings of the IEEE International Conference on Image Processing, 276–280, 2015.

[42] Zhang, R. J. Uniform interpolation curves and surfaces based on a family of symmetric splines. *Computer Aided Geometric Design* Vol. 30, No. 9, 844–860, 2013.

[43] Deslauriers, G.; Dubuc, S. Symmetric iterative interpolation processes. In: *Constructive Approximation*. DeVore, R. A.; Saff, E. B. Eds. Springer US, 49–68, 1989.

[44] Beccari, C. V.; Casciola, G.; Romani, L. Non-uniform interpolatory curve subdivision with edge parameters built upon compactly supported fundamental splines. *BIT Numerical Mathematics* Vol. 51, No. 4, 781–808, 2011.

[45] Conti, C.; Gemignani, L.; Romani, L. Exponential pseudo splines: Looking beyond exponential B splines. *Journal of Mathematical Analysis and Applications* Vol. 439, No. 1, 32–56, 2016.

[46] Botsch, M; Sorkine, O. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics* Vol. 14, No. 1, 213–230, 2008.

[47] Terzopoulos, D.; Platt, J.; Barr, A.; Fleischer, K. Elastically deformable models. *ACM SIGGRAPH Computer Graphics* Vol. 21, No. 4, 205–214, 1987.

[48] Davies, R. H.; Cootes, T. F.; Taylor, C. J. A minimum description length approach to statistical shape modeling. In: *Information Processing in Medical Imaging*. Insana, M. F.; Leahy, R. M. Eds. Springer Berlin Heidelberg, 50–63, 2001.

[49] Styner, M. A.; Rajamani, K. T.; Nolte, L. P.; Zsemlye, G.; Székely, G.; Taylor, C. J.; Davies, R. H. Evaluation of 3D correspondence methods for model building. In: *Information Processing in Medical Imaging*. Taylor, C.; Noble, J. A. Eds. Springer Berlin Heidelberg, 63–75, 2003.

[50] Aguet, F.; Jacob, M.; Unser, M. Three dimensional feature detection using optimal steerable filters. In: Proceedings of the IEEE International Conference on Image Processing, II-1158–II-1161, 2005.

[51] Badoual, A.; Schmitter, D.; Unser, M. An inner product calculus for periodic functions and curves. *IEEE Signal Processing Letters* Vol. 23, No. 6, 878–882, 2016.

[52] Jacob, M.; Blu, T.; Unser, M. An exact method for computing the area moments of wavelet and spline curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 23, No. 6, 633–642, 2001.

D. Schmitter received his master degree in bioengineering and biomedical technologies from the École Polytechnique Fédérale de Lausanne (EPFL), Switzerland, in 2013. He was with the Advanced Clinical Imaging Technology Group, Siemens, at the Center for Biomedical Imaging, Switzerland, where he was one of the main contributors working on brain-imaging software and related image-processing algorithms. Currently, he is a Ph.D. student at the Biomedical Imaging Group, EPFL, where he is working on spline-based shape representation and segmentation problems. He has developed several segmentation and tracking methods in the field of biomedical imaging.

P. García-Amorena was ranked in the top ten of the Spanish regional university entrance exam of Catalunya in 2009 and obtained double-bachelor degrees in mathematics and industrial engineering at Barcelona Tech (UPC), Spain. He received an Introduction to Research Grant from the Spanish National Research Council (CSIC) to work on 3D image reconstruction from deformation scans. In 2014 he won a La Caixa award for top undergraduate Spanish students. In 2016 he obtained his master degree in computational science and engineering at EPFL, Switzerland. His research interests include numerical methods and mathematical foundations for shape modeling, computer vision, and image processing.

M. Unser is the professor and director of EPFL's Biomedical Imaging Group, Lausanne, Switzerland. His primary area of investigation is biomedical image processing. He is internationally recognized for his research contributions to sampling theory, wavelets, the use of splines for image processing, stochastic processes, and computational bioimaging. He has published over 250 journal papers on those topics. He is the author with P. Tafti of the book *An Introduction to Sparse Stochastic Processes*, Cambridge University Press, 2014. From 1985 to 1997, he was with the Biomedical Engineering and Instrumentation Program, National Institutes of Health, Bethesda, USA, conducting research on bioimaging. Dr. Unser has held the position of associate Editor-in-Chief (2003–2005) for the *IEEE Transactions on Medical Imaging*. He is currently a member of the editorial boards of *SIAM J. Imaging Sciences*, *IEEE J. Selected Topics in Signal Processing*, and *Foundations and Trends in Signal Processing*. He is the founding chair of the technical committee on Bio Imaging and Signal Processing (BISP) of the IEEE Signal Processing Society. Prof. Unser is a Fellow of the IEEE (1999), an EURASIP Fellow (2009), and a member of the Swiss Academy of Engineering Sciences. He is the recipient of several international prizes including three IEEE-SPS Best Paper Awards and two Technical Achievement Awards from the IEEE (2008 SPS and EMBS 2010).