Computer-Aided Design 43 (2011) 1126–1136

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Modeling smooth shape using subdivision on differential coordinates

Lifeng Zhu[a,b], Shengren Li[a,b,c], Guoping Wang[a,b,∗]

[a] The Key Lab on Machine Perception and Intelligence of MOE, Peking University, Beijing 100871, China
[b] Graphics and Interative Technology Lab, Peking University, Beijing 100871, China
[c] Department of Computer Science, University of California at Davis, United States

ARTICLE INFO

Article history:
Received 4 June 2010
Accepted 10 May 2011

Keywords:
Subdivision
Differential coordinates
Surface editing

ABSTRACT

Traditional subdivision schemes are applied on Euclidean coordinates (the spatial geometry of the control mesh). Although the subdivision limit surfaces are almost everywhere C^2 continuous, their mean-curvature normals are only C^0. In order to generate higher quality surfaces with better-distributed mean-curvature normals, we propose a novel framework to apply subdivision for shape modeling, which combines subdivision with differential shape processing. Our framework contains two parts: subdivision on differential coordinates (a kind of differential geometry of the control mesh), and mutual conversions between Euclidean coordinates and differential coordinates. Further discussions about various strategies in both parts include a special subdivision method for mean-curvature normals, additional surface editing options, and a version of our framework for curve design. Finally, we demonstrate the improvement on surface quality by comparing the results between our framework and traditional subdivision methods.

1. Introduction

Subdivision is well known for its modeling power in generating smooth surfaces from arbitrary polygonal meshes [1]. The uniformity of its representation allows users to model smooth shapes by designing control meshes, avoiding constraint management like NURBS patches management. Moreover, the subdivision surfaces naturally represent splines in the form of polygonal meshes, which leads to the potential for the seamless integration of CAD and CAE [2]. Successful subdivision schemes like the Catmull–Clark and Loop schemes iteratively refine control meshes and generate C^2 continuous surfaces, except at extraordinary vertices where the continuity is only C^1 [3]. How to improve the quality of subdivision surfaces, either globally or just near the extraordinary vertices, is more attractive in CAD applications and still intensively discussed in the field of shape modeling.

To describe the surface quality, besides parametric continuity, the curvature distribution is also an important criterion [4]. The ripple artifacts on subdivision surfaces near extraordinary vertices can be measured by the variation of curvatures. Minimizing the variation of Gaussian curvatures around extraordinary vertices can substantially improve the local surface quality. In this paper, instead of adjusting the subdivision stencil according to the curvature variation, we directly adjust the vertex geometry according to a refined mean-curvature distribution (see Fig. 1) at each iteration of subdivision. A smooth shape can be constructed after several iterations of such subdivision. We name this new modeling framework *differential space subdivision*.

Traditional subdivision is performed on the Euclidean coordinates, i.e. the spatial geometry indicating where the control mesh is embedded in $\mathbb{R}^3$. Our differential space subdivision instead subdivides the differential geometry of the control mesh. Guided by the refined differential geometry, a smooth shape is finally reconstructed. It turns out that the quality of the surfaces is improved under our modeling framework, which differs from previous efforts focusing on local stencil modifications near extraordinary vertices [4,5]. Another feature of this modeling framework is that the quality is improved all over the surface, not only near the extraordinary vertices.

One related work named "subdivision shading" [6] presents a subdivision on vertex normals for shading and reveals the idea that subdivision is a general interpolation technique. Our work is greatly inspired by this idea. However, we replace vertex normals with discrete mean-curvature normals, whose direction is the vertex normal and magnitude is the discrete vertex mean curvature. Higher-order differential geometry (mean curvature) is employed to improve the surface quality. A linear adjustment for vertex positions according to the refined differential geometry is also provided, which is not performed in [6].

Our work also follows a recent trend in shape modeling — differential shape processing. Differential quantities of shapes are used either directly or indirectly in shape editing [7–9] and image painting [10]. All these works show that manipulating or preserving the variation of geometry indirectly manipulates or preserves the details of the shape. It will be no surprise that smoothly interpolating the variation of geometry turns out a

∗ Corresponding author. Tel.: +86 010 62751781.
E-mail address: wgp@pku.edu.cn (G. Wang).

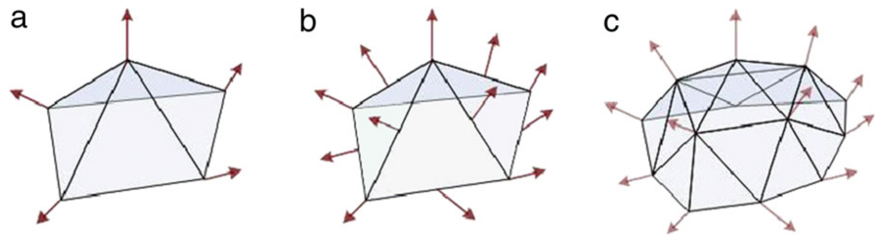


Fig. 1. (a) The control mesh, (b) the refined discrete mean-curvature normals, and (c) the refined geometry adjusted according to the refined discrete mean-curvature normals.

smoother shape. In the following text, we also call the discrete mean-curvature normal *differential coordinates* for consistency with the previous work on differential shape processing.

2. Related work

What has been subdivided. Subdivision schemes are commonly used to generate smooth surfaces from sparse discrete surfaces. Here we summarize them in the view of data interpolation, especially for geometric data. Traditional subdivision schemes can be classified into primal and dual schemes. Primal schemes like Catmull–Clark and Loop create dense vertex data from sparse control meshes. Dual schemes like Doo–Sabin and Midedge essentially split the vertices and create dense face data [3]. Wang et al. [11] present a novel class of subdivision schemes, which could generate smooth vector fields from discrete edge data on meshes. In contrast to subdividing spatial measurements on control meshes, Vanraes et al. [12] subdivide control triangles tangent to the surface at each vertex and then construct a surface, which is constrained to be tangent to the refined control triangles. Alexa et al. [6] subdivide the vertex normals of control meshes on $\mathbb{S}^2$ and this yields high quality normals for surface rendering.

Subdivision with differential geometry. Instead of vertex normals, we are subdividing the discrete mean-curvature normals, which additionally encode differential measurements of the control mesh. Existing subdivision schemes combined with differential measurements are mainly discussed in the curve design community. Ohtake et al. [13] compute refined edge normals by averaging normals of adjacent edges and then reconstructs the refined polygon guided by these normals using a nonlinear optimization. Yang [14] directly calculates the new vertices by averaging vertices and normals of the control polygon and provides a detailed convergence and continuity analysis. A variant of this scheme is shown in [15], in which normals are replaced by curvature normals. Stoddard et al. [16] use tangents to guide the interpolatory subdivision, whose new point corresponding to an edge is the incenter of a triangle, which is formed by the edge and the two tangent lines of the two end points. Compared to these subdivision schemes, the 2D version of our framework is linear, which is preferable to the above nonlinear schemes.

In the area of surface modeling, driving subdivision by differential measurements goes back to [17]. In that work, a control mesh is calculated to meet the interpolation constraints of its Catmull–Clark limit surface, and free vertices are set to minimize certain smooth energy concerning differential measurements. Later, Kobbelt's variational construction [18,19] places new vertices in the refined mesh to minimize a global smooth energy functional. In recent works, curvatures are usually used to guide the modifications near extraordinary vertices. For example, local subdivision stencils are modified in [4], and local control vertices are modified in [20].

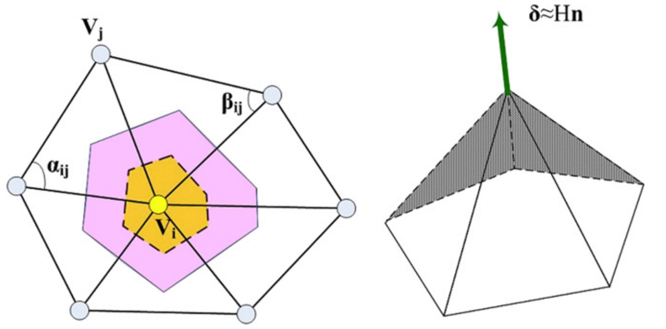
Differential Shape Processing. Discrete differential quantities are employed for mesh fairing [21] and thin shell simulation [22] at first. Both theoretical and experimental results show that minimizing an energy determined by certain differential measurements leads to fair and physically meaningful shapes. Recently, a collection of novel coordinates called differential coordinates have been widely used in computer graphics [23]. The conversions between Euclidean coordinates and differential coordinates are developed for shape processing on differential coordinates, e.g. quantization [24] and filtering [25]. The most significant application of differential coordinates is on mesh deformation, where shape details are preserved through preserving the differential coordinates in a least-square sense [8]. Direct manipulation on curvatures in [7] is based on a conversion from curvatures to spatial coordinates. However, it involves a nonlinear optimization stage.

Another area relevant to our work is mesh refinement based on discrete curvatures [26,27]. Discrete curvatures are used to define an energy or threshold to guide some local refinement operations such as edge swap, edge split or edge collapse. In this paper we propose another way to use discrete curvatures for mesh refining — to subdivide them instead of indirectly using them as energies or thresholds. Previous work concerning subdivision in differential shape editing is presented in [28]. The subdivision surfaces are deformed indirectly by deforming their control meshes using differential coordinates. Instead of subdivision surfaces deformation, our work concerns subdivision in the area of shape modeling. The major contribution of our work is directly combining subdivision with differential coordinates. It can be regarded as a subdivision performed in the differential space or an interpolating procedure on differential coordinates. A special variant of the subdivision method and several surface editing strategies exploiting differential coordinates are also designed for our modeling framework.

3. Differential space subdivision

3.1. Preliminaries

We use triangular meshes to demonstrate our modeling framework. Let $\mathcal{M}^0 = (V^0, E^0, F^0)$ be the given control mesh with n vertices, whose spatial geometry is given by its vertex positions $V^0 = \{\mathbf{v}_1^0, \mathbf{v}_2^0, \dots, \mathbf{v}_n^0\}$ in Euclidean space, and whose combinatorial structure is described by a set of edges E^0 and a set of faces F^0. Traditional primal subdivisions for triangular meshes refine the topology of the meshes by inserting new vertices on the edges and update the geometry through a series of modifications on both old and new vertices. The new positions of the vertices are determined by the affine weights assigned to the old vertices in the local combinatorial structure, which is also called the *stencil* of subdivision. Selected stencils like Loop and Butterfly schemes can generate a sequence of meshes $\{\mathcal{M}^1, \mathcal{M}^2, \dots\}$, which is convergent to an almost everywhere C^1 continuous surfaces. Assume the mesh $\mathcal{M}^k$ has n_k vertices, the operation subdividing $\mathcal{M}^{k-1}$ to $\mathcal{M}^k$ can be encoded in a $n_k \times n_{k-1}$ subdivision matrix S_k, such that $S_k V^{k-1} = V^k$, if the subdivision is linearly dependent on the control mesh. Note that the subdivision matrix S_k used in this paper is not a square matrix, which locally maps the vertices of control mesh



$\mathbf{V_j}$
$\boldsymbol{\beta}_{\mathbf{ij}}$
$\boldsymbol{\alpha}_{\mathbf{ij}}$
$\mathbf{V_i}$
$\boldsymbol{\delta}\approx \mathbf{Hn}$

Fig. 2. Parameters in discrete Laplace–Beltrami operators. The pink region is the Voronoi region A_i of vertex v_i. Yellow region bounded by dashed lines is the approximate Voronoi region after one iteration of subdivision, whose area is 1/4 of A_i. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

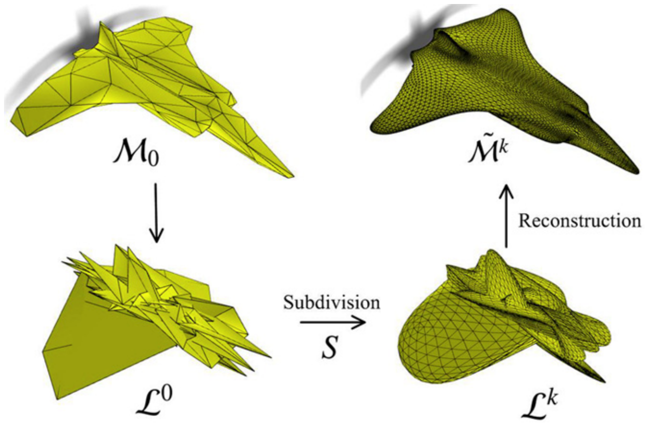


Fig. 3. A subdivision perspective of differential space subdivision.

to the subdivided mesh. It is a global matrix mapping all the vertices of the control mesh to the subdivided mesh. Our matrix is slightly different from the subdivision matrix for analyzing subdivision surfaces. In this paper, the subdivision matrix S_k is only used to introduce and formulate the differential space subdivision. It is never explicitly formed in our implementation. For better understanding our modeling framework, an example of such subdivision matrix is given in the Appendix B.

The mean-curvature normal $H\mathbf{n}$ of a vertex $\mathbf{v}$ on a surface $\mathcal{S}$ is a vector with the vertex mean curvature H as its magnitude and the vertex unit normal $\mathbf{n}$ as its direction. It simultaneously encodes two-order differential geometry (vertex mean curvature) and one-order differential geometry (vertex normal) of the surface. Computing mean-curvature normals in the discrete settings usually makes use of its connection with the Laplace–Beltrami operator [23],

$$\Delta_{\mathcal{S}}\mathbf{v} = H\mathbf{n}.$$

If we apply the Laplace–Beltrami operator on each vertex, we get its corresponding mean-curvature normal. Let $\boldsymbol{\delta}_i$ denote the discrete mean-curvature normal, then the discrete Laplace–Beltrami operator can be formulated as

$$\Delta_{\mathcal{M}}\mathbf{v}_i = \boldsymbol{\delta}_i = \sum_j \omega_{ij}(\mathbf{v}_i - \mathbf{v}_j). \quad (1)$$

Various discrete Laplace–Beltrami operators are developed for different applications. Here we list three of them, which are most relevant to our application:

1. *Combinatorial Laplacian* [29]. When the vertex j is in the 1-ring neighborhood $N(i)$ of the vertex i, the weight ω_{ij} in (1) is set as

   $$\omega_{ij} = 1/d_i, \quad (2)$$

   where d_i is the degree of vertex $\mathbf{v}_i$. Otherwise ω_{ij} is set to 0. Applying the combinatorial Laplacian is efficient because the weights are independent of the vertex geometry.
2. *Cotangent Laplacian* [30]. The weight ω_{ij} is set as 0 except when the vertex j is in the 1-ring neighborhood $N(i)$ of the vertex i,

   $$\omega_{ij} = \frac{\cot\alpha_{ij} + \cot\beta_{ij}}{\sum\limits_{j\in N(i)} \cot\alpha_{ij} + \cot\beta_{ij}}.$$

   As pointed out in [31], when cotangent Laplacian is scaled by the area of Voronoi region A_i (see Fig. 2), the differential coordinate is a good analogue for the vertex mean-curvature normal. Later, it is proved that under some assumptions, cotangent Laplacian is convergent to Laplace operator around regular vertices [32].
3. *Mesh Laplace operator* [33]. Mesh Laplace operator has a little complicated formulation, but it has a nice property that it is point-wise convergent to the Laplace–Beltrami operator on continuous surfaces. We refer its formulation and the convergence proof to [33].

If all the discrete mean-curvature normals and Euclidean coordinates of vertices are arranged in two vectors: $\boldsymbol{\Delta} = (\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \ldots, \boldsymbol{\delta}_n)^T$, $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n)^T$, we can get a linear operator represented in a $n \times n$ matrix form, called the Laplacian matrix L, such that $\boldsymbol{\Delta} = L\mathbf{V}$, where

$$L = \begin{cases} \sum_j \omega_{ij} & i = j \\ -\omega_{ij} & j \in N(i) \\ 0 & \text{otherwise}. \end{cases} \quad (3)$$

Stacking the inverse of Voronoi region into a $n \times n$ diagonal matrix $M = \{1/A_i\}$, called mass matrix, we get the Laplacian operator represented by ML.

Following the practice in the shape processing community [23], we use the term *differential coordinates* to denote the discrete mean-curvature normals. In the following, we show that subdividing the differential coordinates can create smooth shapes. Note that the shapes we design are intrinsic, i.e. they are equivalent under rigid transformations. Their smoothness is determined by the distribution of certain differential measurements, which is an important criterion besides the parametric continuity.

3.2. A framework for differential space subdivision

Instead of subdividing Euclidean coordinates, we subdivide differential coordinates in order to model smooth shapes, which is called *differential space subdivision* in the following. To be clear, we call the subdivision on Euclidean coordinates *spatial subdivisions*. In a perspective of subdivision, the discrete mean-curvature normals are subdivided. After a reconstruction from subdivided discrete mean-curvature normals, the limit surface is expected to have a nice quality (see Fig. 3). In a perspective of differential shape processing, discrete mean-curvature normals are smoothly interpolated on a manifold surface, with smoothed features on the reconstructed limit surface (see Fig. 1).

As stated in Section 3.1, when calculating the differential coordinates by $\boldsymbol{\Delta} = ML\mathbf{V}$, each $\boldsymbol{\delta}_\mathbf{i}$ is an approximation of the mean-curvature normal of the vertex $\mathbf{v_i}$. Assume vertex $\mathbf{v}_\mathbf{i}^{(0)}$ on $\mathcal{M}^0$ carries its discrete mean-curvature normal $\boldsymbol{\delta}_\mathbf{i}^{(0)}$ as its data. Then we employ a primal subdivision to generate denser smoothly distributed data over the control mesh, preferably C^2 continuous away from extraordinary vertices in the limit case. Therefore, if we can find a mesh whose differential coordinates are right the ones generated by subdivision, in the sense of "mean-curvature normal continuity", this mesh is hoped to be smoother than the one created by directly subdividing the spatial geometry of the control mesh.

We illustrate our modeling framework in Fig. 3 and formulate it as follows. Define the *differential mesh* $\mathcal{L}^0 = (\Delta^0, E^0, F^0)$ of $\mathcal{M}_0$ with the same topology as $\mathcal{M}^0$ and the differential coordinates of $\mathcal{M}^0$ as its vertex positions. Subdividing $\mathcal{L}^0$, we get $\{\mathcal{L}^1, \mathcal{L}^2, \ldots\}$. Subdividing $\mathcal{M}^0$, we get $\{\mathcal{M}^1, \mathcal{M}^2, \ldots\}$. We aim to find a mesh $\tilde{\mathcal{M}}^k = (\tilde{V}^k, E^k, F^k)$, such that $\tilde{\mathcal{M}}^k$ has the same topology as $\mathcal{M}^k$, and

$$M_k L_k \tilde{\mathbf{V}}^k = \mathbf{\Delta}^k \tag{4}$$

where M_k, L_k are the mass matrix and the Laplacian matrix of $\tilde{\mathcal{M}}^k$, and the vectors $\mathbf{\Delta}^k$ are composed of the vertex positions of $\mathcal{L}^k$.

Let $\mathcal{L}$ denote the subdivision limit surface of control polygon $\mathcal{L}^0$. As shown in the Appendix A, when a convergent discrete Laplace–Beltrami operator is applied in our framework, such as mesh Laplace operator [33], the limit surface of $\tilde{\mathcal{M}}^k$ generated from our framework can be regarded as the solution of the Laplacian equation

$$\Delta_{\mathcal{S}} \tilde{\mathcal{M}} = \mathcal{L}, \tag{5}$$

which can be solved after adding proper boundary conditions. The analysis of the differential space subdivisions is briefly presented in the Appendix A.

Due to the fact that we have to solve differential equations to get the differential space subdivision surfaces, and what we are subdividing is no longer the simple geometry of the shape, we find that simply constructing the differential space subdivision surfaces from the above equation is costly, and it may introduce other artifacts to the shape. In order to reduce the computational cost while improving the quality of the shape, we construct our modeling framework from a practical point of view in the following subsections.

3.3. *Laplacian discretization and reconstruction*

In this section, we will explore how to choose the proper discrete Laplace–Beltrami operator for discretizing (5) in our modeling framework. Three candidates of discrete Laplace–Beltrami operators mentioned in Section 3.1 are discussed and compared here, and we recommend to use the combinatorial Laplacian in the end. Readers who are not interested in the details about how to implement the other two discrete Laplace–Beltrami operators can skip the following two paragraphs.

First, because we have to solve a Laplacian system (5), in order to reduce the computational time and memory usage, a sparse linear system is preferred. As stated in Section 3.2, if we use the mesh Laplace operator [33], the reconstruction result is supposed to be convergent theoretically. However, it is not efficient to apply mesh Laplace operator in our modeling framework, because the Laplacian matrix of mesh Laplace operator is a dense matrix.[1]

If we discretize (5) using the cotangent Laplacian with mass matrix, the differential coordinates can best approximate the vertex mean-curvature normals while the Laplacian matrix is sparse. The problem with this kind of discretization is that, α_{ij}, β_{ij} in (3) and A_i in mass matrix are all dependent on V, which is unknown before we reconstruct the surface. That is, Eq. (5) can be written as

$$M(V)L(V)V = \Delta. \tag{6}$$

It becomes a nonlinear equation of V. A possible solution for (6) is to convert it into a nonlinear optimization problem. Thus, we turn to solve the corresponding optimization problem of (6),

$$\min_V \|M(V)L(V)V - \Delta\|^2 \overset{\text{def}}{=} \min_V f(V). \tag{7}$$

Using the derivatives of $f(V)$ and the Levenberg–Marquardt algorithm, the approximate solution of (7) can be calculated iteratively. The process of the Levenberg–Marquardt algorithm and the derivative of the Laplacian weights and Voronoi region are discussed in [7].

While the number of vertices exponentially increases after subdivision, the computational cost and numerical instability prohibit the performance of the nonlinear optimization. Thus, we prefer to use the combinatorial Laplacian to linearize (5). In this way, the Laplacian matrix is independent of the geometry of the reconstructed mesh, and (5) can be reduced to a sparse linear system. Although the cotangent Laplacian better approximates the mean curvature normals, uniform weights do encode enough differential information of the control mesh. After subdividing $\mathcal{M}^0$ several times, in $\mathcal{M}^k$, most of the vertices are regular, and most of the faces are nearly isotropic. Therefore, the combinatorial Laplacian is a valid substitute for the cotangent Laplacian. In the uniform refinement scheme, $A_i^{(k)}$ is 1/4 of $A_i^{(k-1)}$, as illustrated in Fig. 2. Locally, the effect of subdivision matrix on mass matrix can be formulated as

$$\begin{aligned}\bar{V}^1 &= (\bar{M}_1\bar{L}_1)^{-1} S_1 \bar{M}_0 \bar{L}_0 \bar{V}^0 = \bar{L}_1^{-1}\bar{M}_1^{-1}\frac{1}{4}\bar{M}_1 S_1 \bar{L}_0 \bar{V}^0 \\ &= \frac{1}{4}\bar{L}_1^{-1} S_1 \bar{L}_0 \bar{V}^0,\end{aligned}$$

which means if we use the combinatorial Laplacian, we can simply scale it with a factor 1/4 instead of using the area of the Voronoi region after each step of subdivision. Thus, the Laplacian matrix becomes independent of the mesh geometry, and the geometry of $\tilde{\mathcal{M}}^k$ can be simply calculated by solving $(1/4)^k L_k^{-1} S_k S_{k-1} \ldots S_1 L_0 V^0$.[2]

Although no theoretical proof for the convergence of the combinatorial Laplacian exists, we find the surfaces modeled from our framework convergent through the statistics. Given the meshes $\{\tilde{\mathcal{M}}^1, \tilde{\mathcal{M}}^2, \ldots\}$ modeled from our framework using the combinatorial Laplacian, we check the Hausdorff distance between $\tilde{\mathcal{M}}^i$ and $\tilde{\mathcal{M}}^{i-1}$. Let $h_i = d_H(\tilde{\mathcal{M}}^i, \tilde{\mathcal{M}}^{i-1})$, where $d_H(M_1, M_2)$ is the Hausdorff distance of M_1 and M_2. If the sequence $\{h_1, h_2, \ldots\}$ converges to zero superlinearly, $\sum_{i=1}^n h_i$ converges, hence the sequence $\{\tilde{\mathcal{M}}^1, \tilde{\mathcal{M}}^2, \ldots\}$ is supposed to be a convergent sequence due to the fact that $d_H(\tilde{\mathcal{M}}^n, \tilde{\mathcal{M}}^0) < \sum_{i=1}^n h_i$. In Fig. 4, we plot the sequence $\{1^{1.5}h_1, 2^{1.5}h_2, \ldots\}$ of the local surface around the vertices whose degree ranges from 3 to 12. The statistics shows that $i^{1.5}h_i$ monotonically decreases to zero, thus h_i is convergent superlinearly and the surfaces around both the regular and extraordinary vertices are expected to be convergent.

We also need to note that differential coordinates are translation-invariant [23], indicating that the rank of the Laplacian matrix L is $n - 1$. To solve (5), additional constraints have to be added in order to make the system have an unique solution. In the shape editing community, users are asked to place several weighted handles to determine the reconstructed mesh. In our modeling framework, since the intrinsic shape is also translation-invariant, the placement of the shape does not matter. So we suggest to add just one handle to fix the placement of the reconstructed shape, e.g. make $\mathbf{v}_0$ at (0, 0, 0).

[1] In fact, a lot of elements in the Laplacian matrix of mesh Laplace operator are close to zero. However, in order to preserve the convergence property, the Laplacian matrix of mesh Laplace operator cannot be simply approximated by a sparse matrix without any strict proof for the convergence.

[2] Strictly, L_k^{-1} is not invertible ($rank(L^k) = n_k - 1$, where n_k is the number of vertices in M_k). We use $(\cdot)^{-1}$ to represent the pseudo-inverse of L_k after adding a handle.

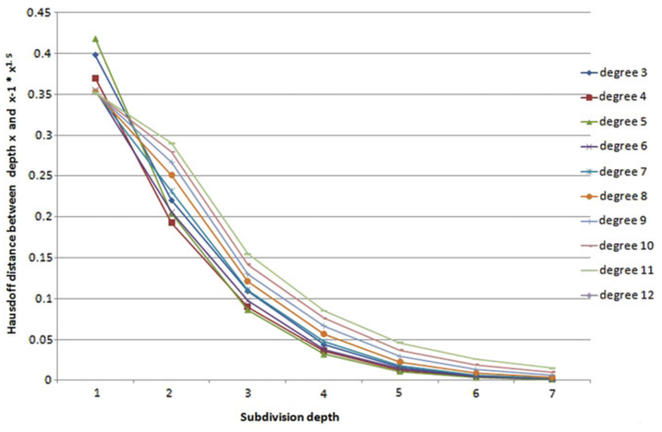


Fig. 4. The convergence of local surface around vertices whose degree range from 3 to 12.

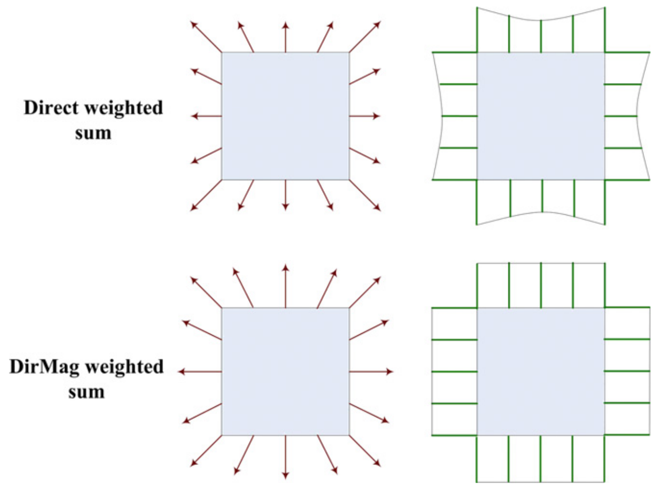


Fig. 5. Different ways for refining differential coordinates of a square by simple linear interpolation. The left column shows refined differential coordinates. The right column shows the mean-curvature distribution on the parametric domain, where the length of green lines represents the mean curvatures. A wavy distribution is found in the upper right figure.

3.4. Subdivide differential coordinates

In our modeling framework, all primal subdivision schemes for triangular meshes are valid when they are performed in the differential space. Since our aim is to produce a surface with higher quality of curvature distribution, we explore the curvature distribution of our reconstructed surface. When directly performing primal subdivision schemes in the differential space, wavy distributions of mean curvatures will emerge, as shown in Fig. 6(a)(c) (mean curvatures are mapped to colors). Looking closely into the subdivision stage, we found that the weighted sum refinement for the discrete mean-curvature normals is the reason for the wavy distribution of mean curvatures. According to the characteristics of the mean-curvature and normal distribution around vertices, we address the following two situations separately.

Case 1. When the discrete mean curvatures are almost uniformly distributed and the normals are in different directions, trivial weighted sum for vectors is prone to cause wavy distributions of mean curvatures on parametric domain, as illustrated in the upper row of Fig. 5. In this case, we introduce a new subdivision method for subdividing differential mesh called *DirMag subdivision* (short for Direction–magnitude subdivision). The implementation of DirMag subdivision contains two steps — separation and subdividing.

Suppose $\boldsymbol{\delta}_i$ is the differential coordinate of vertex $\mathbf{v}_i$ on $\mathcal{M}$. As stated in Section 3.1, $\boldsymbol{\delta}_i$ approximates the mean-curvature normal $H_i\mathbf{n}_i$ of $\mathbf{v}_i$. Thus $|H_i| = \|\boldsymbol{\delta}_i\|$. Noticing that mean curvature H_i can be positive or negative at convex or concave vertices, we need to determine the sign of H_i. We employ the angle weighted normal $\bar{\mathbf{n}}_i$ [34] to help determine the sign. If $\bar{\mathbf{n}}_i \cdot \boldsymbol{\delta}_i > 0, H_i = \|\boldsymbol{\delta}_i\|$, otherwise $H_i = -\|\boldsymbol{\delta}_i\|$, and then $\mathbf{n}_i = \boldsymbol{\delta}_i/H_i$. When $H_i = 0$, we simply let $\mathbf{n}_i = \bar{\mathbf{n}}_i$.

After the separation, we subdivide $\mathbf{n}_i$ and H_i respectively. Since normals are entities in $\mathbb{S}^2$, they are subdivided using weighted sums in $\mathbb{S}^2$. We use the method introduced in [6] to subdivide normals. On the other hand, mean curvature is a scalar associated with each vertex. We subdivide them by weighted sums in Euclidean space using the same subdivision scheme. Finally, the refined differential coordinates are calculated by multiplying the refined mean curvatures with the refined normals.

Case 2. When the discrete mean curvatures are sufficiently different or the normals are similar between neighboring vertices, the weighted sum in Euclidean space for mean curvatures contributes more than the weighted sum in $\mathbb{S}^2$ for normals. The wavy distribution caused by trivial weighted sum in $\mathbb{R}^3$ is insignificant. Therefore, in this case, we employ the trivial weighted sum in $\mathbb{R}^3$ to subdivide the differential coordinates.

In our experiment, we find that vertices on very coarse control meshes usually belong to the first case. So we need to apply DirMag subdivision. For well structured dense meshes, most of the vertices can be classified into the second case. Trivial weighted sum, due to its linearity and simplicity, is preferred. An interesting case is that when a mesh is subdivided one or two times, most of its vertices belong to the second case. Therefore, in our implementation, we only perform the DirMag subdivision in the first one or two iterations. A comparison between the modified Butterfly scheme and the DirMag subdivision using the modified Butterfly stencil is shown in Fig. 6.

3.5. Summary and extensions

We summarize the steps of our framework as follows:
Input: Control triangular mesh $\mathcal{M}^0 = (V^0, E^0, F^0)$
Output: Mesh subdivided k times from differential space subdivision, $\tilde{\mathcal{M}}^k$

1. Calculate the geometry of $\mathcal{L}^0$ by $\Delta = L_0 V^0$ using the combinatorial Laplacian, see (2) and (3).
2. Subdivide $\mathcal{L}^0$ k times using a primal subdivision scheme, such as the Loop scheme, the Modified Butterfly scheme, or our DirMag subdivision method, get a subdivided differential mesh $\mathcal{L}^k = (\Delta^k, E^k, F^k)$.
3. Let the combinatorial structure of $\tilde{\mathcal{M}}^k$ be the edge and face structure of $\mathcal{L}^k$. Calculate Laplacian matrix $\tilde{L}_k$ of $\tilde{\mathcal{M}}^k$ using the combinatorial Laplacian, see (2) and (3).
4. Solve for $\tilde{V}^k$ from $4^k \tilde{L}_k \tilde{V}^k = \Delta^k$ as the geometry of $\tilde{\mathcal{M}}^k$. For $\tilde{L}_k$ is not invertible, we add a constraint $\mathbf{v}_0 = (0, 0, 0)$ into the above linear system, form an over-determined linear equation $A\tilde{V}^k = b$, and get a least-square solution $\tilde{V}^k = (A^T A)^{-1} A^T b$.

Note that in the second stage, we do not need to index the subdivided mesh and explicitly construct the subdivision matrix. The subdivided mesh can be obtained by splitting the vertices and updating the vertex positions using the chosen subdivision stencils.

Curve case. In the 2D case, the differential coordinates of a polygon can be defined as

$$\boldsymbol{\delta}_i = \mathbf{v}_i - \frac{1}{2}(\mathbf{v}_{i-1} + \mathbf{v}_{i+1}). \tag{8}$$

Analogously, we define a differential polygon $\Delta = \{\boldsymbol{\delta}_i\}$ for the control polygon $P = \{\mathbf{v}_i\}$. Traditional subdivision schemes for curves,

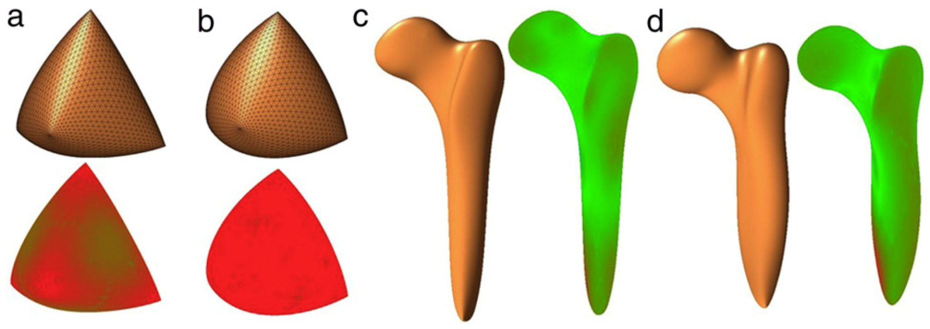


Fig. 6. Comparison between the Modified Butterfly Subdivision (a) (c) and DirMag Subdivision using the Modified Butterfly stencil (b) (d) in differential space subdivision. The mean curvatures are mapped to colors. A bad distribution of mean curvature is spotted in (a). However, it is less obvious in complex models (c). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

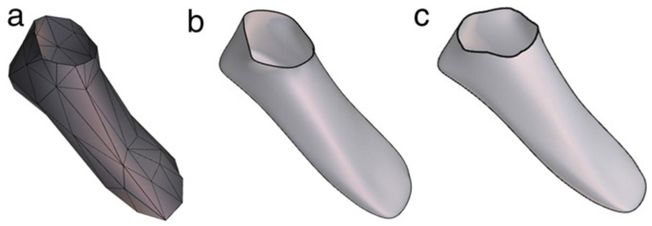


Fig. 7. From left to right are the control mesh, the subdivision surface with and without boundary treatment.

such as the cubic spline algorithm or the four-point interpolatory scheme, are performed on Δ. A smooth shape can be reconstructed from the subdivided differential polygon Δ^k, and the scaling factor is still $(1/4)^k$. The proofs for the convergence and smoothness of the differential space subdivision curves are included in the Appendix A.

Boundary treatment. When a control mesh $\mathcal{M}^0$ is not closed, special stencils are designed to handle the boundary in traditional subdivision schemes, which make the boundary of subdivision surfaces like the limit curve of the boundary polygon. In the differential space subdivision, directly using (1) will cause artifacts around surface boundary, see Fig. 7(right). Similar to the boundary cases of traditional subdivisions, we propose to use the previous curve version of differential coordinates (8) as a substitute for the discrete mean-curvature normals on the boundary. Therefore, the shape of the boundary is preserved, see Fig. 7(middle).

4. Discussions and applications

4.1. Discussions

Computation in the reconstruction Compared to the spatial subdivisions for surfaces, which only need weighted sum operations to calculate the new vertex positions, our differential space subdivision is costly, because the final geometry is reconstructed from the subdivided discrete mean-curvature normals by solving sparse linear systems. However, the vertex positions of the mesh at depth k satisfy that

$$\begin{aligned}\tilde{V}^k &= (M_kL_k)^{-1}S(M_{k-1}L_{k-1}\tilde{V}^{k-1})\\ &= (M_kL_k)^{-1}S(M_{k-1}L_{k-1})\\ &\quad (M_{k-1}L_{k-1})^{-1}S(M_{k-2}L_{k-2}\tilde{V}^{k-2})\\ &= (M_kL_k)^{-1}S^2(M_{k-2}L_{k-2}\tilde{V}^{k-2})\\ &= \cdots = (M_kL_k)^{-1}S^k(M_0L_0\tilde{V}^0).\end{aligned}$$

It means that we do not need to solve the linear system k times. Only one linear system is required to access the subdivided mesh at a specific depth, which makes our scheme still practicable.

Linearity. Another nice feature of our framework is the linearity. Previous normal or curvature normal driven subdivision schemes for curves [13–15] are all nonlinear. Their generalizations to surfaces are supposed to be nonlinear. In our framework, since both the Laplacian operator M_0L_0 and the reconstruction operator $(M_kL_k)^{-1}$ are linear, if a linear subdivision scheme S is taken on the differential mesh $\mathcal{L}_0$, the final reconstructed subdivision mesh is supposed to be linear dependent on the control mesh, i.e. the entire framework is linear.

Globality. The vertex position on the limit surface of the differential space subdivision is affected by all the vertices on the control mesh. It is because the reconstructed vertex positions are solved from a linear system covering all the vertices on the control mesh. That will cause a global effect in the differential space subdivision, i.e. modifying one vertex of the control mesh can affect the geometry of the entire limit surface. However, it is also the globality that makes the entire limit surface possess a better surface quality. At the same subdivision depth, differential space subdivision can globally adjust vertices around the area, where bad surface quality issues may occur, to improve the surface quality. This is not supported in spatial subdivisions, because the affected region of each control vertex is local.

Differences from related works. Our modeling framework is closely related to two previous subdivision constructions — the divided difference schemes and the Kobbelt's variational subdivision. Divided difference schemes emerge in the analysis of subdivision schemes [3]. It points out that local refinement for divided differences improves the continuity of subdivision. Divided difference schemes locally subdivide the divided differences. For example, from the divided difference view of the four-point subdivision, only the inserted vertices are set to satisfy the subdivided differences. In contrast, performing subdivisions on differential meshes is a global version of divided difference scheme. All vertices are updated to satisfy the subdivided divided differences. In Kobbelt's variational construction [18,19], new vertices are set to minimize $\|\Delta V_{k+1}\|$ — a term measuring the total roughness of the shape. Comparatively, our modeling framework relaxes all the old and new vertices to minimize $\|\Delta V_{k+1} - S\Delta V_k\|$. Although they all implicitly define the subdivision results by solving linear systems, our method needs to solve only one linear system to access the mesh at a specific subdivision depth. Moreover, our differential space subdivision is a modeling framework rather than a subdivision scheme. Both the divided difference schemes and variational subdivisions can be employed in the subdivision stage.

4.2. Implementation

In our configuration, besides the DirMag subdivision mentioned in Section 3.4, all the primal subdivision schemes work on the differential meshes. The final distribution of discrete

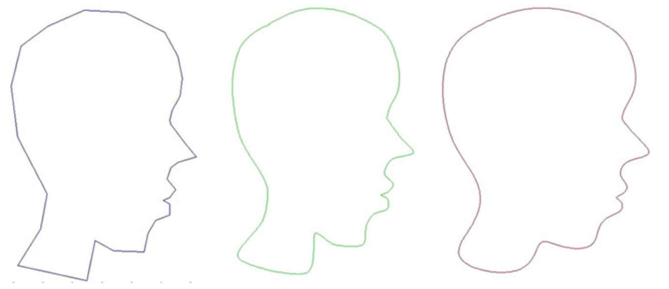


Fig. 8. An example of differential space subdivision for curves. From left to right are the control polygon, its four-point subdivision curve, and its differential space four-point subdivision curve.

mean-curvature normals possesses the characteristics of the chosen scheme. For example, the Loop scheme can generate an almost C^2 discrete mean-curvature normals; the Butterfly scheme can keep the discrete mean-curvature normal interpolatory at each vertex after subdivisions. Non-stationary modifications, such as blending the subdivision surfaces [35], are also applicable. Using such approaches, discrete mean-curvature normals are C^2 continuous at extraordinary vertices.

Among all the candidates for subdividing a shape in the differential space, we prefer to adopt interpolatory schemes. Interpolatory schemes keep the old mean-curvature normals after each iteration of the subdivision, and the shape of the control mesh is well preserved with its details properly interpolated. Various interpolatory schemes are available in [36].

4.3. Results and comparison

Examples of differential spaces subdivisions for surfaces can be seen in Figs. 3, 6, 7, 12(c). Fig. 8 is an illustration of differential space subdivision for curves. Table 1 lists the statistics for the models in Figs. 10 and 13. In our experiment, each model is subdivided three times and the Loop scheme is adopted in the subdivision stage of our modeling framework. Notice that the timing for the spatial subdivision is right the timing for the second stage in differential space subdivision, because the differential mesh and the control mesh have the same combinatorial structure. We use the CGAL library [37] to implement the subdivision, and the TAUCS library [38] to solve the sparse linear systems. All timings in Table 1 were measured on a PC with Intel Core i3 CPU 530/2.93 GHz.

One significant advantage of our modeling framework is the improvement on curvature distribution of the limit surface. Fig. 9 illustrates the comparison between our framework and traditional subdivisions. We map the Gaussian curvatures to the parametric

Table 1
Number of vertices and faces of models and the timing results in each stage (computing differential coordinates, subdividing and reconstruction) of differential space subdivision.

Model	#V	#F	Comp. Diff.	Sub-	Recon-
Star	50	96	0.00025s	0.04673s	0.15729s
Fertility	115	242	0.00052s	0.11356s	0.64840s
Vase	235	442	0.00097s	0.20417s	1.73913s
Kitten	265	530	0.00114s	0.24932s	1.98092s
Rock-arm	323	646	0.00142s	0.30475s	3.25046s

domain. Color maps of the Gaussian curvature are used to illustrate the surface quality. Since the Gaussian curvature of the differential space subdivision surface is relatively low, we magnify it to clearly illustrate the curvature distribution. In Fig. 9 we find that, around a convex vertex of valence 12, the Modified Butterfly subdivision and the Loop subdivision lead to ripples, while our framework does not (the Gaussian curvatures are all positive and small). Even though the continuity of the modified Butterfly scheme is lower than the Loop scheme, when it is adopted in the differential space, the quality of the result surface gets much more better than the Loop scheme in spatial subdivision.

The comparison on the smoothness can be characterized by highlight lines as well. Fig. 13 provides examples using complex models. We also provide a quantitative evaluation of the surface quality for the examples in Fig. 13. The normalized thin-plate energy

$$E = \frac{\int_{\mathcal{S}} (\kappa_1^2 + \kappa_2^2) ds}{\int_{\mathcal{S}} ds}$$

is employed to measure the surface quality here, where κ_1 and κ_2 are the two principle curvatures of the element ds on the surface $\mathcal{S}$. Since not all the subdivision surfaces have analytical expressions, we estimate the principle curvatures κ_1 and κ_2 on the subdivided mesh using the curvature tensor proposed in [39], as it approximates the curvature tensor on the smooth surface when the subdivided mesh is sufficiently dense. Note that smooth surfaces usually have low thin-plate energy, and the thin-plate energy is normalized to measure the smoothness per unit area.

As a substitute for subdivision surfaces, curved triangles also refine the control mesh guided by vertex normals [40]. However, the tangents are not continuous across the edges, because the interior geometry of a curved triangle is actually determined by only one triangle with its vertex normals. Differential space subdivision takes advantage of the combinatorial structure of the control mesh, and the mean-curvature normals are refined guided by the neighboring ones. It gives up the efficiency curved triangles may have and gets a better continuity across the edges. Fig. 10 shows a comparison between a recent progress in curved triangles [41] and our differential space subdivision.

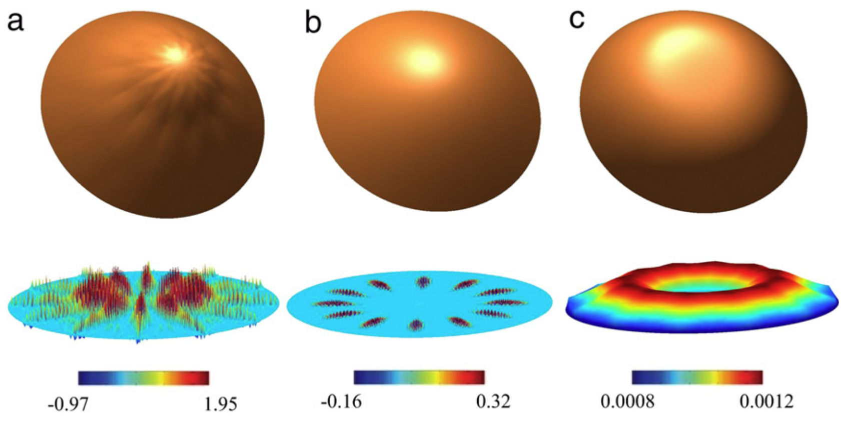


Fig. 9. Gaussian curvature distributions around a 12-degree extraordinary vertex of (a) the modified Butterfly subdivision, (b) the Loop subdivision and (c) the modified Butterfly subdivision in differential space.

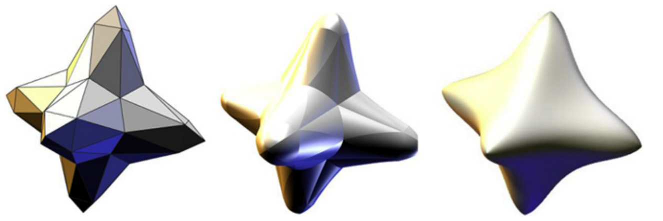


Fig. 10. From left to right are the control mesh, the geometry of Phong tessellation (image courtesy of [41]), and the modified Butterfly subdivision in differential space.

4.4. Applications

To manipulate the limit shape, besides manipulating the vertex positions of the control mesh, it is also possible to adjust the vertex positions of its differential mesh. As mentioned above, the vertex position of the differential mesh is an approximation to the vertex mean-curvature normal. So we can modify the subdivision surfaces with the support of additional normal and curvature controls. Fig. 11(b) demonstrates the editing result by modifying the normal directions of the control mesh. Four vertex normals at corners and four vertex normals on edges are adjusted upwards (red) and downwards (yellow dotted) respectively.

Figs. 11(c) and 12 illustrates the editing result by adjusting the mean curvatures. In Fig. 11(c) the mean curvatures of the four vertices at corners are enlarged. In Fig. 12, the mean-curvature distribution on the control mesh is normalized to [0, 1]. Then a filter

$$f(x) = \begin{cases} 1/2 + (x/8 - 1/16)^{1/4}, & 0.5 < x \leq 1 \\ 1/2 - (1/16 - x/8)^{1/4}, & 0 \leq x \leq 0.5 \end{cases}$$

is employed to polarize the mean-curvature distribution. The differential space subdivision generates a smooth shape with exaggerated geometric features of the control mesh.

4.5. Limitations

Differential space subdivision is inappropriate for applications like real-time interactive modeling and local adjustment for very complex meshes. The high quality of the shape is at the cost of high computational cost. The locality of traditional subdivision schemes enables them to be efficiently mapped to GPU [42], while the global minimization in the differential space subdivision has high computational costs. Some local operations enforce that most of the surface is fixed during the editing. They are not supported in the differential space subdivision due to its globality.

5. Conclusion and future work

We propose a shape modeling framework, which combines subdivision with differential coordinates. Based on the investigation about discrete approximation for mean-curvature normals, we design a framework for subdivision in differential space, together with a special subdivision method for differential coordinates and a curve version of the framework. The improvements on the curvature distribution and editing power are then presented. The experimental results show that subdividing the geometry in differential space can improve the quality of the subdivision surfaces.

In the future, we will look for other possible quantities in differential space, which have efficient mutual conversions with vertex positions in $\mathbb{R}^3$, and discuss the possibility they can be smoothly refined. Following the progress on discrete Laplacian operators of quadrilateral meshes [43], the generalization of our algorithm to quadrilateral meshes using the Catmull–Clark subdivision and its interpolatory counterpart is also a realistic goal. In differential space subdivision, subdivisions are performed on the differential quantities of a surface. We think ripples that may occur on the surface are moved to the higher-order derivatives of the surface. The remaining roughness in the magnified map (see Fig. 9) implies that there is still room for further improvement on the surface quality.

Acknowledgments

This work is partially supported by the National Basic Research Program of China (Project No. 2010CB328002) and the National Natural Science Foundation of China (Project No. 60925007, 60833007, 90915010). Models are courtesy of the AIM@SHAPE Repository. We are also grateful to the anonymous reviewers for their helpful suggestions and comments.

Appendix A

Here we briefly list the proof sketch for the properties of the differential space subdivision curves and surfaces.

First, let us recall the notations for the differential space subdivision curves. Denote by $P_0 = \{\mathbf{v}_i^{(0)}\}$ a closed control polygon, and denote by $\Delta_0 = \{\boldsymbol{\delta}_i^{(0)}\}$ its differential polygon, where $\boldsymbol{\delta}_i^{(0)}$ is computed from (8). Assume the curve subdivided k times has

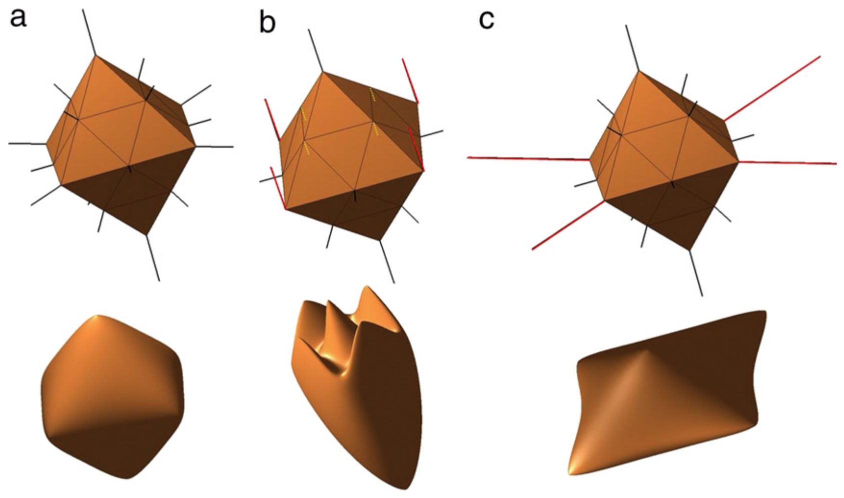


Fig. 11. (a) Differential space modified Butterfly subdivision for octahedron, (b) editing result by adjusting normal directions, (c) editing result by enlarging mean curvatures.

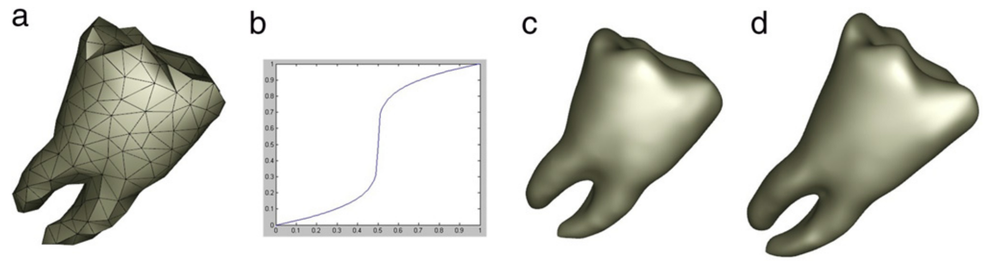


Fig. 12. The differential space subdivision surface of tooth model (a) is modified by filtering the mean curvatures on (a) with a filter (b). The editing result is (d), comparing to the result without modification (c).

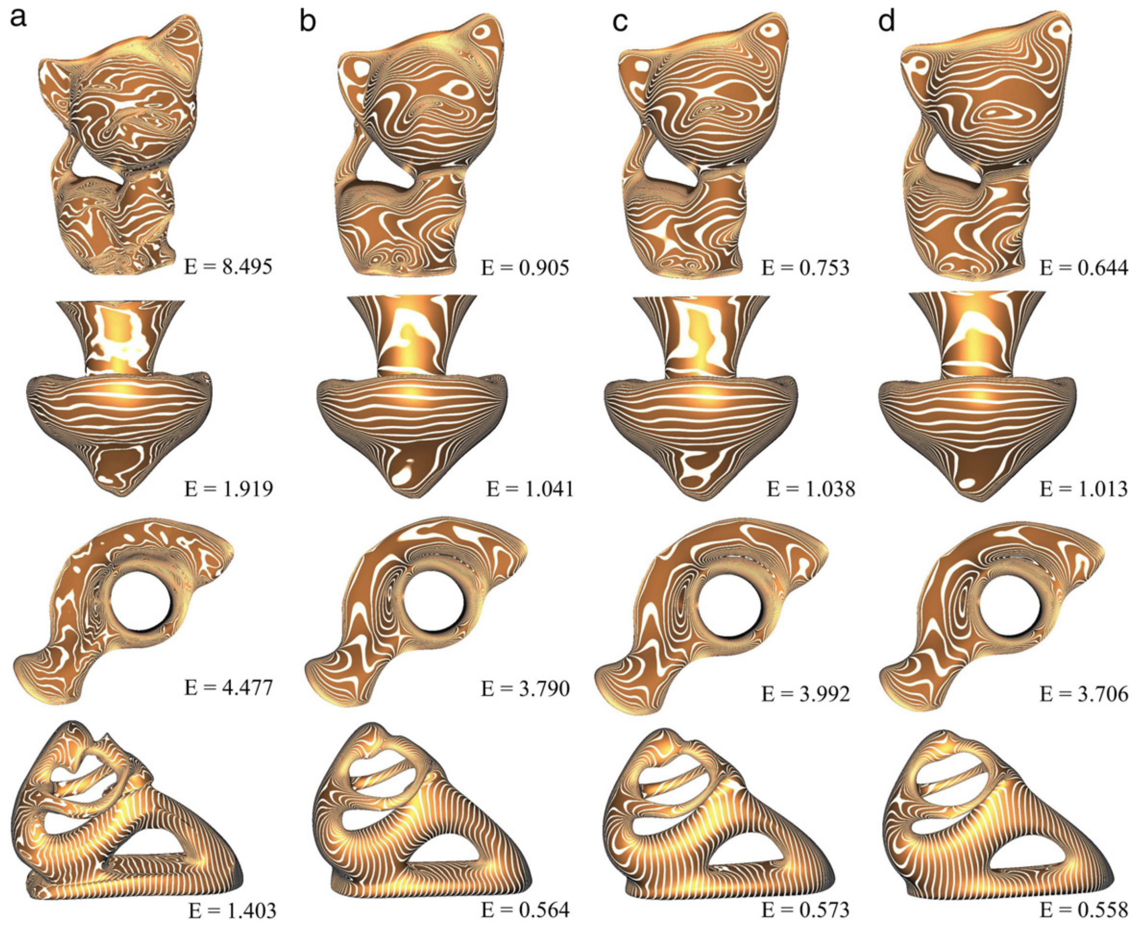


Fig. 13. Highlight lines for (a) the modified Butterfly subdivision, (b) the modified Butterfly subdivision in differential space, (c) the Loop subdivision and (d) the Loop subdivision in differential space. Each model's normalized thin-plate energy E is indicated below.

n_k vertices. Then the polygon P_k subdivided k times from the differential space subdivision satisfies

$$4D_kP_k = S_kD_{k-1}P_{k-1}, \tag{A.1}$$

where the $n_k \times n_k$ matrix D_k is the second-order difference operator and $n_k \times n_{k-1}$ matrix S_k is the subdivision operator.

Lemma 1. *Suppose polygon $P = \{\mathbf{p}_i\}$ is uniformly sampled from a sufficiently smooth curve $c(u)$, i.e. $c(u_i) = \mathbf{p}_i$ and $u_{i+1} - u_i = h$. Then the second-order difference of $\mathbf{p}_i$ satisfies $(\mathbf{p}_{i-1} + \mathbf{p}_{i+1} - 2\mathbf{p}_i)/h^2 = c''(u_i) + O(h^2)$.*

Proof. By Taylor's formula,

$$c(u_i + h) = c(u_i) + c'(u_i)h + \frac{c''(u_i)h^2}{2} + \frac{c'''(u)h^3}{6} + O(h^4)$$

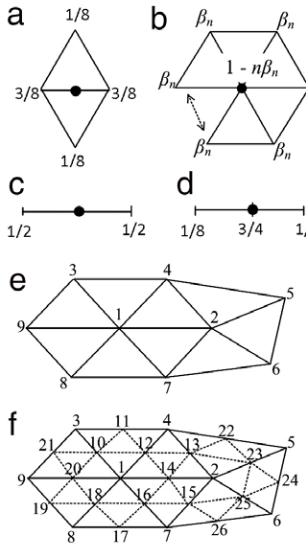
$$c(u_i - h) = c(u_i) - c'(u_i)h + \frac{c''(u_i)h^2}{2} - \frac{c'''(u)h^3}{6} + O(h^4).$$

Summing up the above two equations, we get

$$\frac{\mathbf{p}_{i-1} + \mathbf{p}_{i+1} - 2\mathbf{p}_i}{h^2} = \frac{c(u_i - h) + c(u_i + h) - 2c(u_i)}{h^2}$$
$$= c''(u_i) + O(h^2). \quad \square$$

Lemma 2. *Under uniform parameterization, the second-order divided differences of vertices on P_k (P_k is computed from (A.1)) can be computed by subdividing the second-order divided differences of vertices on P_0 k times.*

Proof. Suppose the control polygon $P_0 = \{\mathbf{v}_i^{(0)}\}$ has n_0 vertices and its parameter domain is $[0, n_0h]$. Under uniform parameterization,



$$S_1 = \begin{pmatrix}
5/8 & 1/16 & 1/16 & 1/16 & 0 & 0 & 1/16 & 1/16 & 1/16 \\
\beta_5 & 1-5\beta_5 & 0 & \beta_5 & \beta_5 & \beta_5 & \beta_5 & 0 & 0 \\
0 & 0 & 3/4 & 1/8 & 0 & 0 & 0 & 0 & 1/8 \\
0 & 0 & 1/8 & 3/4 & 1/8 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/8 & 3/4 & 1/8 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1/8 & 3/4 & 1/8 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1/8 & 3/4 & 1/8 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1/8 & 3/4 & 1/8 \\
0 & 0 & 1/8 & 0 & 0 & 0 & 0 & 1/8 & 3/4 \\
3/8 & 0 & 3/8 & 1/8 & 0 & 0 & 0 & 0 & 1/8 \\
0 & 0 & 1/2 & 1/2 & 0 & 0 & 0 & 0 & 0 \\
3/8 & 1/8 & 1/8 & 3/8 & 0 & 0 & 0 & 0 & 0 \\
1/8 & 3/8 & 0 & 3/8 & 1/8 & 0 & 0 & 0 & 0 \\
3/8 & 3/8 & 0 & 1/8 & 0 & 0 & 1/8 & 0 & 0 \\
1/8 & 3/8 & 0 & 0 & 0 & 1/8 & 3/8 & 0 & 0 \\
3/8 & 1/8 & 0 & 0 & 0 & 0 & 3/8 & 1/8 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 \\
3/8 & 0 & 0 & 0 & 0 & 0 & 1/8 & 3/8 & 1/8 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 \\
3/8 & 0 & 1/8 & 0 & 0 & 0 & 0 & 1/8 & 3/8 \\
0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 & 1/2 \\
0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 & 0 & 0 \\
0 & 3/8 & 0 & 1/8 & 3/8 & 1/8 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 & 0 \\
0 & 3/8 & 0 & 0 & 1/8 & 3/8 & 1/8 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0
\end{pmatrix}_{26\times 9}$$

Fig. B.14. (a) The stencil of the Loop scheme for interior edge-points. (b) The stencil of the Loop scheme for interior vertex-points of valence n, here $\beta_n = \frac{1}{n}(\frac{5}{8} - \frac{1}{64}\sqrt{3 + 2 * \cos(\frac{2\pi}{n})})$ when $n \neq 6$ and $\beta_n = \frac{1}{16}$ when $n = 6$. (c) The stencil of the Loop scheme for boundary edge-points. (d) The stencil of the Loop scheme for boundary vertex-points. (e) The indexes of vertices on $\mathcal{M}_0$. (f) The indexes of vertices on $\mathcal{M}_1$.

the parameter values of $\mathbf{v}_i^{(0)}$ is ih. When subdivided k times, the vertices $\mathbf{v}_i^{(k)}$ on P_k have parameter values $ih/2^k$ under uniform parameterization. Let $h_k = h/2^k$, by Eq. (A.1), we get

$$\frac{1}{(0.5h_{k-1})^2} D_k P_k = \frac{1}{(h_{k-1})^2} S_k D_{k-1} P_{k-1}.$$

Iteratively substitute $\frac{1}{(0.5h_{k-2})^2} D_{k-1} P_{k-1} = \frac{1}{(h_{k-2})^2} S_{k-1} D_{k-2} P_{k-2}$ into the above equation, finally we get

$$\frac{1}{h_k^2} D_k P_k = S_k S_{k-1} \dots S_1 \frac{1}{h_0^2} D_0 P_0. \quad \square \tag{A.2}$$

Proposition 1. *Adding proper boundary conditions, the differential space subdivision curves defined by (A.1) converge.*

Proof. If P_k is uniformly sampled from a curve $d_k(u)$, by Lemma 1, the left side of (A.2) can be written as $d_k''(u) + O(h_k^2)$. Using standard subdivision rules, when $k \to \infty$, the right side of (A.2) uniformly converges to the subdivision limit curve $c(u)$ of the control polygon $D_0 P_0$, which means $d_k''(u)$ uniformly converges to $c(u)$. Thus, $d_k(u)$ converges to the solution of the differential equation

$$d^2 x(u)/du^2 = c(u). \tag{A.3}$$

Adding proper boundary conditions, such as fixing one vertex of the closed curve (Dirichlet boundary condition), the differential space subdivision curve converges to the solution of the differential equation (A.3), which can be obtained by integrating it twice. $\square$

Proposition 2. *If the subdivision schemes used in the subdivision stage of differential space subdivision can generate C^k continuous curves, the differential space subdivision curves defined by (A.1) are at least C^{k+2} continuous.*

Proof. If the subdivision scheme $S_k S_{k-1} \dots S_1$ generates a C^k continuous curve, then subdividing the differential polygon Δ_0, we get a C^k continuous limit curve $c(u)$. By Proposition 1, the differential space subdivision curve $d(u)$ can be regarded as the solution of (A.3). So $d(u)$ is at least C^{k+2} continuous. $\square$

The proof for the convergence of the differential space subdivision curves can also be extended to the surface case. For differential space subdivision surfaces defined by (4), the differential mesh Δ_k uniformly converges under standard subdivision rules, which means when $k \to \infty$, the right hand side of (4) converges to the limit surface of the differential mesh, denoted by $\mathcal{L}$. Just like the second-order difference operator converges to the second-order differential operator in the curve case, if the discrete Laplace–Beltrami operator $M_k L_k$ in (4) converges to the Laplace–Beltrami operator $\Delta_{\mathcal{S}}$, the differential space subdivision surface is expected to be convergent to the solution of the Laplacian equation $\Delta_{\mathcal{S}} \mathcal{X} = \mathcal{L}$. Convergent discrete Laplace–Beltrami operators do exist [33], but the formulation and the proof for the convergence is more complicated than the curve case, and we are not planning to repeat it here. Interested readers are referred to [33] for more details.

Proposition 3. *Using convergent discrete Laplace–Beltrami operators, the differential space subdivision surfaces defined by (4) converge when proper boundary conditions are added.*

By Proposition 2, we know that the differential space subdivision curve is smoother than its spatial subdivision counterpart in the sense of geometric continuity. For the surface case, when convergent discrete Laplace–Beltrami operators are adopted, the vertices on the differential mesh are convergent to the mean-curvature normals of the differential space subdivision surface. So its distribution enjoys the same property of the chosen subdivision schemes performed in the differential spaces.

Appendix B

The construction of the subdivision matrix S_k used in this paper depends on the chosen subdivision scheme and how the vertices are indexed. Here we give an example of S_k, see Fig. B.14. In the example, the mesh $\mathcal{M}_0$ is subdivided one time using the Loop scheme. The original mesh $\mathcal{M}_0$ and the subdivided mesh $\mathcal{M}_1$ are indexed as Fig. B.14(e) and (f). The element s_{ij} in the matrix is the affine weight of the jth vertex on $\mathcal{M}_0$ when we are calculating the ith vertex on $\mathcal{M}_1$. For example, in Fig. B.14, according to the stencil, the vertex $v_1^{(1)}$ on $\mathcal{M}_1$ is computed as a weighted sum of vertices on $\mathcal{M}_0$,

$$v_1^{(1)} = \frac{5}{8} v_1^{(0)} + \frac{1}{16} v_2^{(0)} + \frac{1}{16} v_3^{(0)} + \frac{1}{16} v_4^{(0)} + \frac{1}{16} v_7^{(0)} + \frac{1}{16} v_8^{(0)} + \frac{1}{16} v_9^{(0)}.$$

Therefore, the elements $s_{11}, s_{12}, s_{13}, s_{14}, s_{17}, s_{18}, s_{19}$ of the subdivision matrix are $5/8$, $1/16$, $1/16$, $1/16$, $1/16$, $1/16$, $1/16$ respectively, and the other elements on the first row are all zero.

References

[1] Ma Weiyin. Subdivision surfaces for CAD — an overview. Comput Aided Design 2005;37(7):693–709.

[2] Wang L, Wang GH. Subdivision-based integration of geometric design and finite element mesh generation for sheet metal structures. Int J Comput Appl Technol 2008;32(1).

[3] Zorin D, Schroder P. Subdivision for modeling and animation. Computer graphics proceedings, annual conference series: SIGGRAPH conference proceedings, 2000.

[4] Augsdorfer UH, Dodgson NA, Sabin MA. Tuning subdivision by minimising gaussian curvature variation near extraordinary vertices. Comput Graph Forum 2006;25(3):263–72.

[5] Myles A, Peters J. Bi-3 C^2 polar subdivision. In: Proc. of SIGGRAPH. ACM Trans Graph 2009;28(3).

[6] Alexa M, Boubekeur T. Subdivision shading. ACM Trans Graph 2008;27(5).

[7] Eigensatz M, Sumner RW, Pauly M. Curvature-domain shape processing. In: Proceeding of Eurographics 2008. Comput Graph Forum 27(2): 241–50.

[8] Sorkine O, Cohen-or D, Lipman Y, Alexa M, Rossel C, Seidel H-P. Laplacian surface editing. In: Proceedings of SGP. 2004.

[9] Yu Y, Zhou K, Xu D, Shi X, Bao H, Guo B, et al. Mesh editing with poisson-based gradient field manipulation. ACM Trans Graph 2004;23(3):644–51.

[10] McCann J, Pollard NS. Real-time gradient-domain painting. In: Proc. of SIGGRAPH. ACM Trans Graph 2008;27(3).

[11] Wang K Weiwei, Tong Y, Desbrun M, Schroder P. Edge subdivision schemes and the construction of smooth vector fields. ACM Trans Graph 2006;25(3): 1041–8.

[12] Vanraes E, Bultheel A. A tangent subdivision scheme. ACM Trans Graph 2006; 25(2):340–55.

[13] Ohtake Y, Belyaev A, Seidel H-P. Interpolatory subdivision curves via diffusion of normals. Computer Graphics International 2003.

[14] Yang X. Normal based subdivision scheme for curve design. Comput Aided Geom Design 2006;23(3):243–60.

[15] Zhao H, Qiu X, Liang L, Sun C, Zou B. Curvature normal vector driven interpolatory subdivision. In: Proceeding of IEEE international conference on shape modeling and applications. 2009.

[16] Stoddard Jacob, Daniel Bergeron R, House Donald. Tangent driven interpolative subdivision. Comput Graph 2007;31:737–49.

[17] Halstead M, Kass M, DeRose T. Efficient, fair interpolation using Catmull–Clark surfaces. Computer graphics (Proceedings of SIGGRAPH), 1993. p. 35–44.

[18] Kobbelt L. A variational approach to subdivision. Comput Aided Geom Design 1996;13:743–61.

[19] Kobbelt L. Discrete fairing and variational subdivision for freeform surface design. Vis Comput 2000;16(3/4):142–58.

[20] Ginkel I, Umlauf G. Loop subdivision with curvature control. In: Eurographics symposium on geom. proc., Eurographics. 2006. p. 163–71.

[21] Desbrun M, Meyer M, Schroder P, Barr A. Implicit fairing of irregular meshes using diffusion and curvature flow. In: Computer graphics (Proceedings of SIGGRAPH). 1999. p. 317–324.

[22] Grinspun E, Hirani A, Desbrun M, Schroder P. Discrete shells. In: ACM SIGGRAPH symposium on computer animation. 2003. p. 62–7.

[23] Sorkine O. Laplacian mesh processing. In: Eurographics 2005—State of the art reports. Eurographics. Dublin (Ireland): The Eurographics Association; p. 53–70.

[24] Sorkine O, Cohen-or D, Toledo S. High-pass quantization for mesh encoding. In: Proceedings of the Eurographics/ACM SIGGRAPH symposium on geometry processing. 2003. p. 42–51.

[25] Nealen A, Igarashi T, Sorkine O, Alexa M. Laplacian mesh optimization. In: Proceedings of GRAPHITE. 2006. p. 381–9.

[26] Dyn N, Hormann K, KIM S-J, Levin D. Optimizating 3D triangulations using discrete curvature. In: Mathematical methods for curves and surfaces. Oslo 2000. 2001. p. 135–46.

[27] Surazhsky V, Gotsman C. Explicit surface remeshing. In: Proceedings of eurographics symposium on geometry processing. 2003. p. 17–28.

[28] Zhou K, Huang X, Xu W, Guo B, Shum H-Y. Direct manipulation of subdivision surfaces on GPUs. ACM Trans Graph 2007;26(3).

[29] Zhang H. Discrete combinatorial Laplacian operators for digital geometry processing. In: Proc. SIAM conf. geom. design and comp. 2004. p. 575–92.

[30] Pinkall U, Polthier K. Computing discrete minimal surfaces and their conjugates. Exp Math 1993;2(1):15–36.

[31] Meyer M, Desbrun M, Schroder P, Barr A. Discrete differential-geometry operator for triangulated 2-manifolds. In: Proc. VisMath2002.

[32] Xu Guoliang. Discrete Laplace–Beltrami operators and their convergence. Comput Aided Geom Design 2004;21(8):767–84.

[33] Belkin M, Sun J, Wang YS. Discrete laplace operator on meshed surfaces. In: Proceedings of the 24th symp. on comp. geom. 2008. p. 278–87.

[34] Thurmer G, Wuthrich CA. Computing vertex normals from polygonal facets. J Graphics Tools 1998;3(1):43–6.

[35] Levin A. Modified subdivision surfaces with continuous curvature. ACM Trans Graph 2006;25(3):1035–40.

[36] Lin S, Luo X, You F, Li Z. Deducing interpolating subdivision schemes from approximating subdivision schemes. ACM Trans Graph 2008;27(3) Article 146.

[37] Cgal. version 3.5.1. Computational geometry algorithms library. http://www.cgal.org.

[38] Taucs: a library of sparse linear solvers, version 2.2. Available online at http://www.tau.ac.il/~stoledo/taucs/. Sept. 2003.

[39] Cohen-steiner D, Morvan J-M. Restricted delaunay triangulations and normal cycle. In: Symposium on computational geometry. 2003. p. 312–21.

[40] Vlachos A, Peters J, Boyd C, Mitchell J. Curved PN triangles. In: Proceedings of ACM symposium on interactive 3D. 2001. p. 159–66.

[41] Boubekeur T, Alexa M. Phong tessellation. ACM Trans Graph 2008;27(5).

[42] Shiue L, Jones I, Peters J. A realtime GPU subdivision kernel. In: Proc. of SIGGRAPH. ACM Trans Graph 2005;24(3).

[43] Liu D, Xu GL, Zhang Q. A discrete scheme of Laplace–Beltrami operator and its convergence over quadrilateral meshes. Comput Math Appl 2008;55(6): 1081–93.