DOI: 10.1111/cgf.14014

COMPUTER GRAPHICS *forum*
Volume 00 (2020), number 00 pp. 1–11

Non-Uniform Subdivision Surfaces with Sharp Features

Yufeng Tian, Xin Li and Falai Chen

School of Mathematical Sciences, University of Science and Technology of China, Kunming, China
tyf2015@mail.ustc.edu.cn, {lixustc, chenfl}@ustc.edu.cn

Abstract

Sharp features are important characteristics in surface modelling. However, it is still a significantly difficult task to create complex sharp features for Non-Uniform Rational B-Splines compatible subdivision surfaces. Current non-uniform subdivision methods produce sharp features generally by setting zero knot intervals, and these sharp features may have unpleasant visual effects. In this paper, we construct a non-uniform subdivision scheme to create complex sharp features by extending the eigen-polyhedron technique. The new scheme allows arbitrarily specifying sharp edges in the initial mesh and generates non-uniform cubic B-spline curves to represent the sharp features. Experimental results demonstrate that the present method can generate visually more pleasant sharp features than other existing approaches.

Keywords: non-uniform Catmull–Clark surface, subdivision, sharp feature, generalized eigen-polyhedron

ACM CCS: I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling—Curve; surface; solid; object representations

1. Introduction

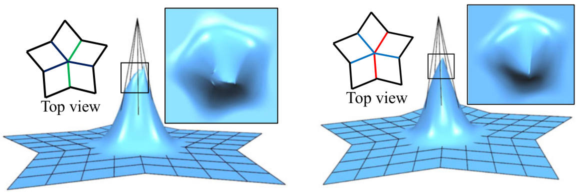
Sharp features on surfaces play an important role in geometric modelling, where Non-Uniform Rational B-Splines (NURBS) and subdivision surfaces are two principal surface representations. NURBS is the industry standard in computer-aided design (CAD) while subdivision surfaces are widely used in 3D computer animation. NURBS has the limitation of rectangular topology. In addition, sharp features on NURBS surfaces can only be iso-parameter lines. Therefore, it is an extremely difficult task to generate complex sharp features on NURBS surfaces.

The main advantage of subdivision representation is the efficient way to generate smooth surfaces of arbitrary topology. Subdivision surfaces are also attractive because they are conceptually simple and can be easily modified to create surface features without making major changes to the original subdivision rules. Lots of approaches [HDD*94, DKT98, BLZ00, BMZB02, KSD14a, KS99, NLG12] have been proposed to produce sharp features for subdivision surfaces. These schemes are directly modified from Catmull–Clark [CC78] or Loop [Loo87] subdivision rules. Many software systems, such as Maya, Pixar's proprietary Presto animation system and OpenSubdiv libraries 2019 support the generation of sharp features. [BLZ00] is a seminal work to introduce improved rules for Catmull–Clark and Loop surfaces that overcome several problems with the original schemes. However, all these approaches work only for uniform subdivision, which means all the sharp features can only be uniform cubic B-spline curves and are not totally NURBS-compatible.

In industry design, NURBS is the dominant standard. Thus, it is extremely important to construct NURBS compatible subdivision schemes in order to apply subdivision in CAD 2001, [Ma05]. For this purpose, [SZSS98] introduced the first NURBS-compatible non-uniform subdivision scheme. And later this scheme was improved by many researchers [MRF06, Urs09, MFR*10, KBZ15, LFS16]. In order to create non-uniform sharp features, a general approach is to set some knot intervals to be zeros [SZSS98, KSD14b, ZML15]. However, the sharp features created by these approaches may have unpleasant visual effects.

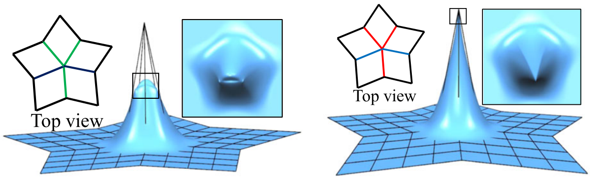
For the approach in [SZSS98], some unpredictable results occur when it is used to generate sharp features. Figures 1(a) and 2(a) show two examples of the blending functions at a valence five extraordinary point generated from the same initial control mesh with different tags. Similar behaviour arises for the sharp features generated by [ZML15].

The approach in [KSD14b] first inserts some zero knot intervals into the mesh that will not change the limit surface. Then one



(a) Result from [SZSS98] (b) Result by our method

Figure 1: *The knot intervals of dark blue edges are 0 while those of the blue edges are 0.2. The other knot intervals are all 1's. The red edges are marked as crease edges.*



(a) Result from [SZSS98]. (b) Result by our method.

Figure 2: *The knot intervals for the dark blue and blue edges are 0 and 0.2, respectively, and the other knot intervals are all 1's. The red edges are marked as crease edges.*

can move the associated vertices corresponding to crease edges to create sharp features. This method does not allow crease curves to span or extend to extraordinary points because the truncated multiple knot lines in their method actually do not work on the one-ring edges of the extraordinary points (see Figure 3, middle). Moreover, the generated surfaces still have the problem that the corresponding blending function at an extraordinary point may have two local maxima when the knot intervals are different [LFS16].

Therefore, unlike the uniform sharp feature scheme [BLZ00], the existing non-uniform sharp feature subdivision schemes cannot provide satisfactory results. In fact, it is extremely difficult to generate complex sharp features for NURBS and NURBS-compatible subdivision surfaces. This paper tries to solve the problem by generalizing the subdivision scheme in [BLZ00] to non-uniform subdivision, which can handle all desired features in a unified approach. Motivated by the idea of [LFS16], we find that the subdivision scheme in [BLZ00] can be reconstructed by extending the eigen-polyhedron technique. Then we generalize this technique to handle non-uniform parameterization and derive a non-uniform subdivision scheme supporting sharp features. This provides a NURBS-compatible subdivision the ability to support arbitrary non-uniform sharp features, and meanwhile retains the advantages of the uniform counterpart in [BLZ00]. The new scheme allows arbitrarily specified sharp edges in the initial mesh and generates non-uniform cubic B-spline curves to represent sharp features. Experimental results demonstrate that our method yields visually more pleasant sharp features than other existing approaches. Two examples generated by our method are shown in Figures 1(b) and 2(b).

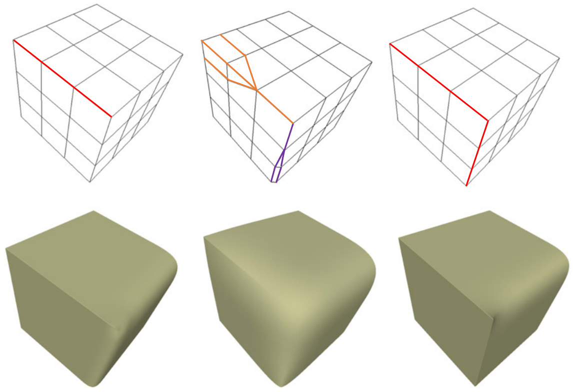


Figure 3: *The first and third columns are results generated using our method, and the second column is the result produced by [KSD14b]. Red edges are marked as crease edges and the edges corresponding to multiple knot lines in [KSD14b] are shown in orange(triple) and purple(double).*

The remainder of this paper is organized as follows. We first review the uniform sharp feature subdivision scheme and the basic idea of eigen-polyhedron technique in Section 2 and 3, respectively. In Section 4, we design a new non-uniform sharp feature subdivision scheme via the generalized eigen-polyhedron technique. The numerical examples are demonstrated in Section 5. Finally, we conclude the paper with a summary and future work in Section 6.

2. Uniform subdivision scheme with sharp features

Before describing the uniform subdivision rules in [BLZ00], we first introduce tagged meshes which are used to specify the types for edges and vertices. Edges can be tagged as *crease edges* and a vertex with incident crease edges receives one of the following tags:

- *crease vertex*: joins exactly two incident crease edges smoothly.
- *corner vertex*: connects two or more creases in a corner (convex or concave).
- *dart vertex*: causes the crease to blend smoothly into the surface.

In addition, all the edges on the boundary of a mesh are tagged as crease edges and boundary vertices are tagged as corner or crease vertices. Crease edges divide the mesh into separate patches, several of which can meet in a corner vertex. At a corner vertex, the creases meeting at that vertex separate the ring of quadrilaterals around the vertex into sectors. Each sector of the mesh is labelled as *convex sector* or *concave sector* indicating how the surface should approach the corner, which are set by the user. The only restriction placed on sector tags is that concave sectors consist of at least two faces. An example of a tagged mesh is given in Figure 4. According to these markings, [BLZ00] proposes a modified Catmull–Clark rule to generate sharp feature curves on the subdivision surface.

Vertex points: The standard vertex rules are applied to reposition untagged vertices and dart vertices. The new control point at a vertex is the weighted average of the control points in its neighbourhood. If a vertex has n adjacent polygons, then the centre vertex has the weight $1-\beta_1-\beta_2$, while all the adjacent vertices have the weight β_1/n; the remaining vertices in the ring receive the weight β_2/n with $\beta_1 = \frac{3}{2n}$ and $\beta_2 = \frac{1}{4n}$. A crease vertex is refined as the average of its

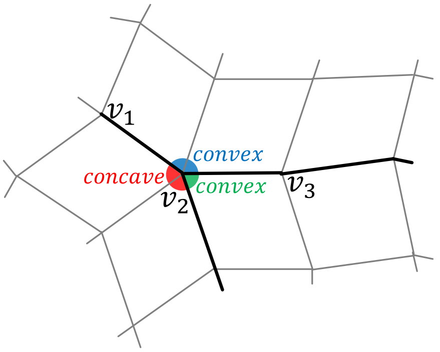


Figure 4: *A tagged mesh. Black edges are tagged as crease edges. v_1, v_2 and v_3 are dart , corner and crease vertices, respectively.*

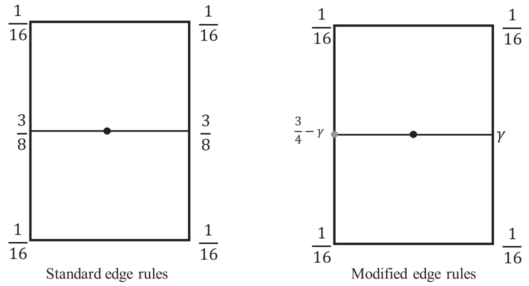


Figure 5: *The edge point rules for untagged edges. When both endpoints are untagged, standard rule is used (left) and the modified rule (right) is used in case of tagged endpoint.*

old position with the weight 3/4 and the two adjacent crease vertices with the weight 1/8. Corner vertices are interpolated.

Face points: The new vertex is inserted at the centroid of each face.

Edge points: This is the most complicated case. The rule for an edge point is chosen depending on the tag of the edge and the tags of adjacent vertices and sectors. In the absence of tags, the standard edge rules are applied, which are shown in Figure 5 (left). The case of an untagged edge adjacent to a tagged vertex is illustrated in Figure 5 (right). The standard edge rule is modified in the following way in this case: suppose that the rule is parameterized by an angle θ which depends on the adjacent vertex tag and sector tag and will be below. The new edge point rule is illustrated on the right of Figure 5, where γ is given in terms of the parameter θ:

$$\gamma = 3/8 - 1/4\cos\theta.$$

For a dart vertex, $\theta = 2\pi/m$, where m is the total number of polygons adjacent to the vertex. If the vertex is a crease vertex, $\theta = \pi/m$, where m is the number of polygons adjacent to vertex in the sector of the edge. At a convex corner vertex, $\theta = \alpha/m$, where α is the angle of the sector, that is the angle between the two crease edges spanning the sector (m is the same as above). If it is a concave corner, then $\theta = (2\pi - \alpha)/m$.

3. Eigen-polyhedra

This section reviews the basic idea of eigen-polyhedron technique, which is firstly used to define non-uniform Catmull–Clark subdivision surface in [LFS16]. For the notations, the capital letters with hats denote the control points in R^2 for an eigen-polyhedron in the following.

As we know, uniform subdivision schemes are well understood according to the eigen-properties of subdivision matrices, which can be exactly computed via discrete Fourier transformation. However, the approach fails for non-uniform subdivisions because the schemes are not shift invariant [PR08]. Moreover, the coefficients for non-uniform subdivision schemes are rational functions of knot intervals. Thus, it is almost impossible to directly compute the eigen-properties of a non-uniform subdivision matrix in terms of knot intervals.

One common approach to define a non-uniform subdivision scheme is to use a heuristic way to generalize non-uniform bi-cubic B-spline refinement rules to arbitrary valences, such as [SZSS98, KBZ15]. However, the subdivision matrices by these approaches cannot ensure the second and third eigenvalues to be identical, which is the necessary condition for the surface to be G^1 [PR08]. Another common approach is to convert the non-uniform knots to be uniform with or without additional pre-insertions [MRF06, Urs09, MFR*10]. Nevertheless, it still causes unnecessary awkward behaviours on resulting surfaces if the knot intervals are different [LFS16].

Instead of defining subdivision rules directly, eigen-polyhedron technique [LFS16] firstly defines an eigen-polyhedron in R^2 for each extraordinary point from which the subdivision matrix is derived. The main property is that, the eigen-polyhedron after mapping by the subdivision matrix is a scaling and translation of the original eigen-polyhedron, which ensures that the subdivision matrix has two identical eigenvalues. To be specific, eigen-polyhedron is a polyhedron mesh in 2D (as illustrated in Figure 6), denoted by $\hat{P}^0 = [\hat{F}_0^0, \cdots, \hat{F}_{n-1}^0, \hat{E}_0^0, \cdots, \hat{E}_{n-1}^0, \hat{V}^0]^T$ which is a $(2n+1) \times 2$ matrix. An eigen-polyhedron satisfies $\hat{P}^1 = M\hat{P}^0 = \lambda\hat{P}^0 + I\hat{T}^0$, where $\hat{P}^0$ has vertex $\hat{V}^0 = (0,0)$, M is a $(2n+1) \times (2n+1)$ matrix whose rows sum to one, $\lambda \in \mathbb{R}$, $\hat{T}^0 \in \mathbb{R}^2$ and I is a $2n+1$-dimensional

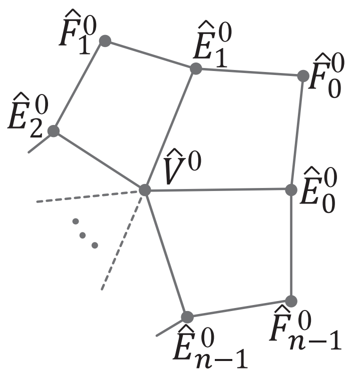


Figure 6: *The topology of the eigen-polyhedron $\hat{P}_0$.*

vector whose elements are all 1′s. In other words, $M\hat{P}^0$ produces a scale of $\hat{P}^0$ by a factor of λ, followed by a translation by $\hat{T}^0$.

After defining the eigen-polyhedra, one can derive the subdivision matrix M from the relationship $\hat{P}^1 = M\hat{P}^0$. Now the basic framework to construct a non-uniform subdivision scheme M via the eigen-polyhedron technique consists of following steps:

- Define the eigen-polyhedron $\hat{P}^0 = [\hat{F}_0, \dots, \hat{F}_{n-1}, \hat{E}_0, \dots, \hat{E}_{n-1}, \hat{V}]$. Note that for bi-cubic non-uniform B-splines and Catmull–Clark subdivision, the coordinates of the eigen-polyhedron can be defined as the two eigenvectors of the second and third eigenvalues. Thus, for non-uniform extraordinary points, the eigen-polyhedron is defined such that it can be reduced to bi-cubic non-uniform B-splines for valence four and to Catmull–Clark subdivision if the knot intervals are all the same.
- Define λ to be the second and third eigenvalues for the subdivision matrix and $\hat{T}$ to be the vertex point through the vertex point rule on $\hat{P}^0$. Let $\hat{P}^1 = \lambda \cdot \hat{P}^0 + I \cdot \hat{T}$.
- Use the relation $\hat{P}^1 = M\hat{P}^0$ to solve the non-zero elements of M.

The above construction has been verified in [LFS16] and a large number of examples show that the basis functions of the new subdivision scheme have much better shapes than all the existing approaches when knot intervals are different.

4. Design non-uniform subdivision scheme with sharp features

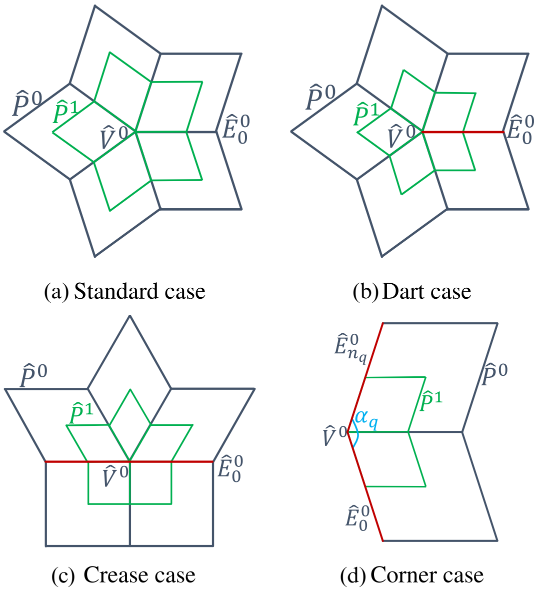
In order to design a non-uniform scheme that can produce high quality sharp features, we generalize the scheme in [BLZ00] to non-uniform case using a similar idea in [LFS16]. We find that the subdivision scheme in [BLZ00] has a similar eigen-polyhedron structure, which we call *generalized eigen polyhedron* (see Section 4.1 for details). We then define generalized eigen-polyhedra for non-uniform subdivision scheme in Section 4.2. Finally, the subdivision scheme is constructed by solving a system generated from the generalized eigen polyhedron in Section 4.3. As the scheme is generalized from [BLZ00], it automatically inherit all the advantages of the scheme in [BLZ00], such as the ability to arbitrarily specify sharp edges on the input mesh to generate different types of complex sharp features.

4.1. Reformulate the uniform scheme with sharp features

Corresponding to the tags in Section 2, the uniform scheme that produces sharp features in [BLZ00] can actually be divided into three cases: dart, crease and corner. For the sake of consistency, we also refer to the standard rule for untagged vertices as standard case. The subdivision schemes for these cases can be defined by constructing their corresponding subdivision matrices. Then all these subdivision matrices can be reformulated, respectively, through the following steps by referring to Figure 7.

1. Define the generalized eigen polyhedron $\hat{P}^0$:
Standard case: For $i = 0, 1, \dots, n-1$, the vertices of $\hat{P}^0$ are defined

$$\begin{cases} \hat{V}^0 &= (0,0), \\ \hat{E}_i^0 &= (\cos(i\cdot\theta^0), \sin(i\cdot\theta^0)), \\ \hat{F}_i^0 &= \tau(\hat{E}_i^0 + \hat{E}_{i+1}^0), \end{cases} \tag{1}$$



(a) Standard case (b) Dart case
(c) Crease case (d) Corner case

Figure 7: *The relation between $\hat{P}^1$ and $\hat{P}^0$ in various cases. Red lines correspond to crease edges.*

where $\tau = \frac{4}{c_n+1+\sqrt{(c_n+9)(c_n+1)}}$, $c_n = \cos(\theta^0)$, and $\theta^0 = \frac{2\pi}{n}$.
Dart case: For $i = 0, 1, \dots, n-1$, the vertices of $\hat{P}^0$ are computed through

$$\begin{cases} \hat{V}^0 &= (0,0); \\ \hat{E}_i^0 &= (\cos(i\cdot\theta^0), \sin(i\cdot\theta^0)), \\ \hat{F}_i^0 &= \hat{E}_i^0 + \hat{E}_{i+1}^0, \end{cases} \tag{2}$$

where $\hat{V}^0\hat{E}_0^0$ corresponds to the crease edge in dart case, and θ^0 is defined as above.
Crease case: Let n_1 be the number of polygons adjacent to v in the first sector. The number of polygons adjacent to v in the second sector is naturally $n_2 = n - n_1$, and $n_2 = 0$ if v is a boundary vertex. Then for $i = 0, \dots, n-1$, $j = 0, \dots, n_1 - 1$ and $k = 0, \dots, n_2 - 1$, the vertices of $\hat{P}^0$ are computed by

$$\begin{cases} \hat{V}^0 &= (0,0), \\ \hat{E}_j^0 &= (\cos(j\cdot\theta^1), \sin(j\cdot\theta^1)), \\ \hat{E}_k^0 &= (\cos(\pi + k\cdot\theta^2), \sin(\pi + k\cdot\theta^2)), \\ \hat{F}_i^0 &= \hat{E}_i^0 + \hat{E}_{i+1}^0, \ i = 0, 1, \dots, n-1, \end{cases} \tag{3}$$

where $\hat{V}^0\hat{E}_0^0$ and $\hat{V}^0\hat{E}_{n_1}^0$ denote the two crease edges in crease case, respectively, and $\theta^1 = \frac{\pi}{n_1}$, $\theta^2 = \frac{\pi}{n_2}$.
Corner case: Considering that the sum of the angles of all the sectors around v in this case may not be 2π, we deal with each sector separately. Let n_q be the number of polygons adjacent to v in the qth sector, and r is the total number of sectors around v, then $n = \sum_{q=1}^{r} n_q$. For $i = 0, \dots, n_q$ and $j = 0, \dots, n_q - 1$, the vertices of the qth sector of $\hat{P}^0$ are computed as

$$\begin{cases} \hat{V}^0 &= (0,0), \\ \hat{E}_i^0 &= (\cos(i\cdot\theta^q), \sin(i\cdot\theta^q)), \\ \hat{F}_j^0 &= \hat{E}_j^0 + \hat{E}_{j+1}^0, \end{cases} \tag{4}$$

where $\hat{V}^0\hat{E}^0_0$ and $\hat{V}^0\hat{E}^0_{n_q}$ denote the two crease edges spanning the qth sector. Denote the angle of the qth sector by α_q, then $\theta^q = \alpha_q/n_q$ if v is a convex corner, and $\theta^q = (2\pi - \alpha_q)/n_q$ if v is a concave corner.

2. Define the polyhedron $\hat{P}^1$: For the standard case, $\hat{P}^1 = \lambda\hat{P}^0$, where $\lambda = \frac{1+\tau}{4\tau} = \frac{5+c_n+\sqrt{(c_n+9)(c_n+1)}}{16}$ and for the other cases, $\hat{P}^1 = \frac{1}{2}\hat{P}^0$.
3. Determine the non-zero elements of M: Since each row of M corresponds to the face, edge or vertex point rule in subdivision, determining the non-zero elements in each row of M is equivalent to designing the stencils of these rules. These stencils are already evident in Section 2. First, the face point is the bilinear combination of the four vertices of its generating face. For the vertex point, both the untagged and dart vertex are linear combinations of all the vertices in its neighbourhood; A crease vertex is a linear combination of the corresponding vertex and the two adjacent crease vertices; Corner vertices are interpolated as itself. Finally, the edge point on the crease edge is a linear combination of the two end vertices and the edge point on the untagged edge is the linear combination of the six vertices of the two adjacent faces.
4. Use the relation $\hat{P}^1 = M\hat{P}^0$ to solve the non-zero elements of M.

We can verify that the subdivision scheme in Section 2 can be obtained through the above procedure. In the next two sections, we show that this idea can be generalized to non-uniform subdivision to support sharp features.

4.2. Define the generalized eigen-polyhedron $\hat{P}^0$ for non-uniform subdivision

Suppose the edges adjacent to the valence-n vertex v have knot interval values $d_0, \ldots, d_{n-1}$, as shown in Figure 8. For the generalized eigen polyhedron $\hat{P}^0$ around v, we set $\hat{V}^0 = (0,0)$, and express the points $\hat{E}^0_i$ and $\hat{F}^0_i$, $i = 0, \ldots, n-1$ as functions of n and $d_0, \ldots, d_{n-1}$. The only rigid requirements are that the scheme must reduce to Equations (1)–(4) when $d_0 = \cdots = d_{n-1}$.

The edge points $\hat{E}^0_i$ in $\hat{P}^0$ are determined by two quantities: (1) the angles between edges in $\hat{P}^0$, $\theta_i = \angle\hat{E}^0_i\hat{V}^0\hat{E}^0_{i+1}$, $i = 0, \ldots, n-1$, $i \equiv i \bmod n (i = 0, \ldots, n_{q-1}$ if v is a corner vertex), and (2) the length of the edges in $\hat{P}^0$, l_i, $i = 0, \ldots, n-1 (i = 0, \ldots, n_q$, if v is a corner vertex). Firstly, the angle θ_i of $\hat{P}^0$ is determined through the same way as the uniform case described in Section 4.1. Secondly, the length of the edges in $\hat{P}^0$ take the values as described in [LFS16], namely,

$$l_i = \frac{d_i + d_i^+ + d_i^-}{3}, i = 0, \ldots, n-1. \tag{5}$$

where

$$d_i^+ = \sum_{j=i, |i-j| \le \frac{n}{4}}^{i+n-1} d_j \cos\left(\frac{i-j}{n} \cdot 2\pi\right),$$

$$d_i^- = -\sum_{j=i, |i-j| > \frac{n}{4}}^{i+n-1} d_j \cos\left(\frac{i-j}{n} \cdot 2\pi\right).$$

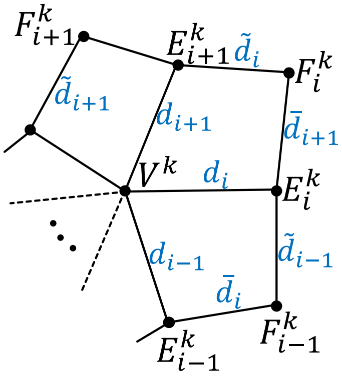


Figure 8: *Knot intervals near an extraordinary vertex, where $\bar{d}_i = \tilde{d}_i = d_i$.*

Note that the length in corner case should be assigned according to Equation (5) before dealing with each sector in order to ensure the consistency of the length of the common edge of adjacent sectors.

The face points $\hat{F}^0_i$ in $\hat{P}^0$ for standard case are obtained using $\hat{F}^0_i = \tau(\hat{E}^0_i + \hat{E}^0_{i+1})$ in Equation (1) and for the other cases, using $\hat{F}^0_i = \hat{E}^0_i + \hat{E}^0_{i+1}$ in Equations (2)–(4).

The polyhedron $\hat{P}^1$ can be defined in a similar way as that for uniform case except adding one more degree of freedom. That is, $\hat{P}^1 = \lambda\hat{P}^0 + I \cdot \hat{T}$, where $\lambda = \frac{1+\tau}{4\tau} = \frac{5+c_n+\sqrt{(c_n+9)(c_n+1)}}{16}$ for the standard case, and $\lambda = \frac{1}{2}$ for the other cases. Here $\hat{T} \in \mathbb{R}^2$ is determined by the vertex point rule as discussed in details in the next subsection.

Remark 1 There are some degrees of freedoms to define the generalized eigen-polyhedra, that is the angle θ_i and the length l_i. We tried different ways to define the generalized eigen-polyhedra and found that minor changes occur in resulting subdivision surfaces. We leave it as a future work to design some optimized generalized eigen-polyhedra.

4.3. Construct non-uniform subdivision scheme with sharp features

In this section, we directly utilize the relation $\hat{P}^1 = M\hat{P}^0$ to derive subdivision rules for the non-uniform sharp feature scheme. And the non-zero elements in the subdivision matrix M are the same as those for uniform case.

4.3.1. *Vertex point rules*

The only restriction for the vertex point rule is that it must reduce to uniform vertex point rule when all the knot intervals are the same. As $\hat{V}^0 = (0,0)$, $\hat{T} = \hat{V}^1$ is easily derived from the equations in Section 4.2 and the vertex point rule is also used to define $\hat{T}$.

Standard case: When there are no tagged information, referring to Figure 9(a), we use the vertex point rule in [LFS16], that is,

$$\hat{V}^{k+1} = \frac{n-3}{n}\hat{V}^k + \frac{3}{n} \cdot \frac{\sum_{i=0}^{n-1}(m_i H_i^k + f_i G_i^k)}{\sum_{i=0}^{n-1}(m_i + f_i)}, \tag{6}$$

where

$$\begin{cases} H_i^k & = g_i \hat{E}_i^k + (1-g_i)\hat{V}^k, \\ G_i^k & = g_i(1-g_{i+1})\hat{E}_i^k + g_{i+1}(1-g_i)\hat{E}_{i+1}^k \\ & \quad + g_i g_{i+1}\hat{F}_i^k + (1-g_i)(1-g_{i+1})\hat{V}^k, \\ g_i & = \frac{d_{i-2}+d_{i+2}+d_i}{d_{i-2}+d_{i+2}+4d_i}, \\ f_i & = \prod_{j=1, j\neq i, i+1}^{n} d_j^+, \\ m_i & = f_i + f_{i-1}. \end{cases} \tag{7}$$

and

$$\hat{T} = \hat{V}^1 = \frac{3}{n} \cdot \frac{\sum_{i=0}^{n-1}(m_i H_i^0 + f_i G_i^0)}{\sum_{i=0}^{n-1}(m_i + f_i)}. \tag{8}$$

Dart case: Referring to Figure 9(b), we introduce Equation (9) (which is a minor modification of Equation (6)) to decide the update of dart vertices.

$$\hat{V}^{k+1} = \frac{n-3}{n}\hat{V}^k + \frac{3}{n} \cdot (\mu \cdot \sum_{i=0}^{[\frac{n-1}{2}]}(m_i H_i^k + f_i G_i^k) + (1-\mu) \cdot \sum_{i=[\frac{n-1}{2}]+1}^{n-1}(m_i H_i^k + f_i G_i^k))/(\mu \cdot \sum_{i=0}^{[\frac{n-1}{2}]}(m_i + f_i) + (1-\mu) \cdot \sum_{i=[\frac{n-1}{2}]+1}^{n-1}(m_i + f_i)). \tag{9}$$

Denote

$$\begin{cases} (x_1, y_1) & = \sum_{i=0}^{[\frac{n-1}{2}]}(m_i H_i^0 + f_i G_i^0), \\ (x_2, y_2) & = \sum_{i=[\frac{n-1}{2}]+1}^{n-1}(m_i H_i^0 + f_i G_i^0), \end{cases}$$

then μ in Equation (9) satisfies $\mu y_1 + (1-\mu)y_2 = 0$, namely, $\mu = \frac{y_2}{y_2 - y_1}$. This choice of μ guarantees that $\hat{V}^1$ locates on the line of $\hat{V}^0\hat{E}_0^0$ in order to ensure $\hat{E}_0^1$ is a linear combination of $\hat{V}^0$ and $\hat{E}_0^0$. It can be verified that $\mu = \frac{1}{2}$ when $d_0 =, \cdots, = d_{n-1}$. So

$$\hat{T} = \hat{V}^1 = \frac{3}{n} \cdot (\mu \cdot \sum_{i=0}^{[\frac{n-1}{2}]}(m_i H_i^0 + f_i G_i^0) + (1-\mu) \cdot \sum_{i=[\frac{n-1}{2}]+1}^{n-1}(m_i H_i^0 + f_i G_i^0))/(\mu \cdot \sum_{i=0}^{[\frac{n-1}{2}]}(m_i + f_i) + (1-\mu) \cdot \sum_{i=[\frac{n-1}{2}]+1}^{n-1}(m_i + f_i)). \tag{10}$$

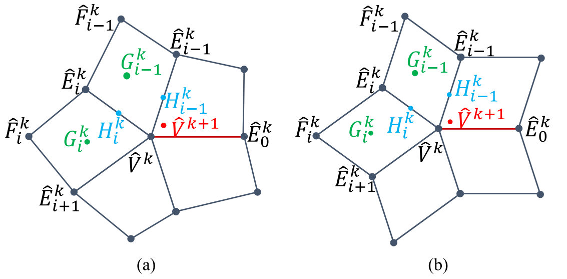


Figure 9: *Vertex point rules in standard case (a) and dart case (b). Red lines correspond to crease edges.*

Crease case: As the generalization of uniform subdivision scheme in [BLZ00], we naturally use the vertex point rule of non-uniform cubic B-spline curves as the vertex point equation here (see Figure 10(a)),

$$\hat{V}^{k+1} = \frac{d_0 \hat{E}_{n_1}^{k+1} + (d_{n_1} + d_0)\hat{V}^k + d_{n_1}\hat{E}_0^{k+1}}{2(d_{n_1} + d_0)}, \tag{11}$$

where

$$\hat{E}_{n_1}^{k+1} = \frac{(d_{n_1} + 2d_0)\hat{E}_{n_1}^k + 3d_{n_1}\hat{V}^k}{2(2d_{n_1} + d_0)}, \tag{12}$$

$$\hat{E}_0^{k+1} = \frac{3d_0\hat{V}^k + (d_0 + 2d_{n_1})\hat{E}_0^k}{2(d_{n_1} + 2d_0)}. \tag{13}$$

And

$$\hat{T} = \hat{V}^1 = \frac{2d_0(d_{n_1} + 2d_0)^2\hat{E}_{n_1}^0 + 2d_{n_1}(2d_{n_1} + d_0)^2\hat{E}_0^0}{8(d_{n_1} + d_0)(2d_{n_1} + d_0)(d_{n_1} + 2d_0)}. \tag{14}$$

Corner case: The corner vertices are interpolated (see Figure 10(b)),

$$\hat{V}^{k+1} = \hat{V}^k, \tag{15}$$

and

$$\hat{T} = \hat{V}^1 = \hat{V}^0 = (0, 0). \tag{16}$$

4.3.2. *Face point rules*

According to the restriction for the polyhedron $\hat{P}^1$, we have

$$\hat{F}_i^1 = \lambda \hat{F}_i^0 + \hat{T}. \tag{17}$$

At this stage of the process, we know $\hat{T}$, λ and $\hat{F}_i^0$. So the Cartesian coordinates of $\hat{F}_i^1$ can be computed.

To generalize the face point rule, we create an equation for $\hat{F}_i^1$ in terms of the four vertices of the face corresponding to $\hat{F}_i^1$. Referring

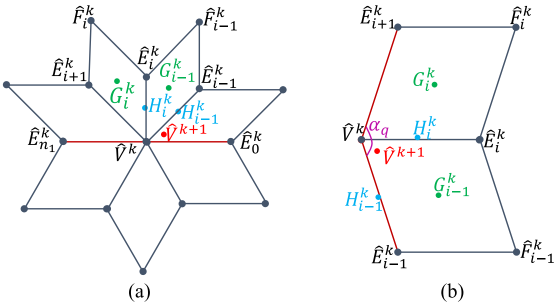


Figure 10: *Vertex point rules in crease case (a) and corner case (qth sector) (b). Red lines represent crease edges.*

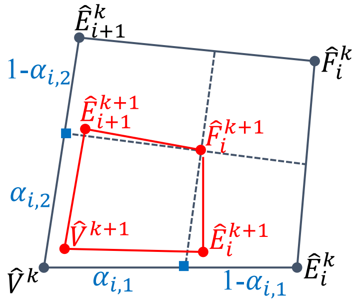


Figure 11: *Compute the face point rule.*

to Figure 11, a reasonable way to define the face point rule is using a bi-linear equation, where

$$\hat{F}^{k+1}_i = (1-\beta_{i,1})(1-\beta_{i,2})\hat{V}^k + \beta_{i,1}(1-\beta_{i,2})\hat{E}^k_i + (1-\beta_{i,1})\beta_{i,2}\hat{E}^k_{i+1} + \beta_{i,1}\beta_{i,2}\hat{F}^k_i, \tag{18}$$

with undetermined coefficients $\beta_{i,1}$ and $\beta_{i,2}$.

To determine $\beta_{i,1}$ and $\beta_{i,2}$, we let $k=0$ in the above equation to get

$$\hat{F}^1_i = (1-\beta_{i,1})(1-\beta_{i,2})\hat{V}^0 + \beta_{i,1}(1-\beta_{i,2})\hat{E}^0_i + (1-\beta_{i,1})\beta_{i,2}\hat{E}^0_{i+1} + \beta_{i,1}\beta_{i,2}\hat{F}^0_i. \tag{19}$$

Then $\beta_{i,1}$ and $\beta_{i,2}$ can be solved from the above equation using the formula in [Flo15a] and [Flo15b]). Denote

$$\begin{aligned} A_1 &= A(\hat{F}^1_i, \hat{V}^0, \hat{E}^0_i), \quad A_2 = A(\hat{F}^1_i, \hat{E}^0_i, \hat{F}^0_i), \\ A_3 &= A(\hat{F}^1_i, \hat{F}^0_i, \hat{E}^0_{i+1}), A_4 = A(\hat{F}^1_i, \hat{E}^0_{i+1}, \hat{V}^0), \\ B_1 &= A(\hat{F}^1_i, \hat{E}^0_{i+1}, \hat{E}^0_i), B_2 = A(\hat{F}^1_i, \hat{V}^0, \hat{F}^0_i). \end{aligned}$$

where

$$A(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) := \begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{vmatrix}.$$

Then

$$\beta_{i,1} = \frac{2A_4}{2A_4 - B_1 + B_2 + \sqrt{D}}, \tag{20}$$

$$\beta_{i,2} = \frac{2A_1}{2A_1 - B_1 - B_2 + \sqrt{D}}, \tag{21}$$

where $D = B_1^2 + B_2^2 + 2(A_1A_3 + A_2A_4)$.

4.3.3. *Edge point rules*

According to the restriction for the polyhedron $\hat{P}^1$, we have

$$\hat{E}^1_i = \lambda\hat{E}^0_i + \hat{T}. \tag{22}$$

As we know $\hat{T}$, λ and $\hat{E}^0_i$, so the Cartesian coordinates of $\hat{E}^1_i$ can be computed.

According to whether the associated edge e is tagged or not, the list of edge point rules in [BLZ00] can be divided into two types:

Normal edge: If the edge e is not tagged, the associated edge point is a linear combination of the six vertices of the two adjacent faces for e in the uniform case [BLZ00]. Inspired by the non-uniform B-spline refinement rules, we first denote

$$P^k_{i,1} = (1-\beta_{i-1,1})\hat{V}^k + \beta_{i-1,1}\hat{E}^k_{i-1} \tag{23}$$

$$P^k_{i,2} = (1-\beta_{i,2})\hat{V}^k + \beta_{i,2}\hat{E}^k_{i+1} \tag{24}$$

$$P^k_{i,3} = (1-\beta_{i-1,1})\hat{E}^k_i + \beta_{i-1,1}\hat{F}^k_{i-1} \tag{25}$$

$$P^k_{i,4} = (1-\beta_{i,2})\hat{E}^k_i + \beta_{i,2}\hat{F}^k_i, \tag{26}$$

then the edge point is computed via the following equation

$$\hat{E}^{k+1}_i = (1-\omega_{i,2})\left(\frac{1-\omega_{i,1}}{2}P^k_{i,1} + \frac{\omega_{i,1}}{2}P^k_{i,2} + \frac{1}{2}\hat{V}^k\right) + \omega_{i,2}\left(\frac{1-\omega_{i,1}}{2}P^k_{i,3} + \frac{\omega_{i,1}}{2}P^k_{i,4} + \frac{1}{2}\hat{E}^k_i\right). \tag{27}$$

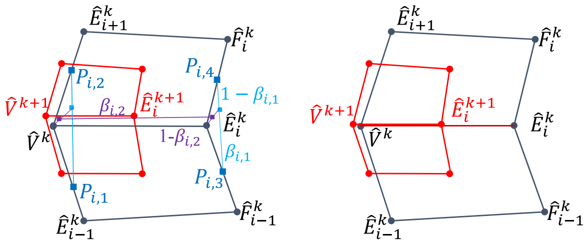
Let $k=0$, then

$$\hat{E}^1_i = (1-\omega_{i,2})\left(\frac{1-\omega_{i,1}}{2}P^0_{i,1} + \frac{\omega_{i,1}}{2}P^0_{i,2} + \frac{1}{2}\hat{V}^0\right) + \omega_{i,2}\left(\frac{1-\omega_{i,1}}{2}P^0_{i,3} + \frac{\omega_{i,1}}{2}P^0_{i,4} + \frac{1}{2}\hat{E}^0_i\right). \tag{28}$$

It is easy to see that $\hat{E}^1_i$ is also a bi-linear combination of four points $\frac{P^0_{i,1}+\hat{V}^0}{2}$, $\frac{P^0_{i,2}+\hat{V}^0}{2}$, $\frac{P^0_{i,3}+\hat{E}^0_i}{2}$ and $\frac{P^0_{i,4}+\hat{E}^0_i}{2}$ with coefficients $\omega_{i,1}$ and $\omega_{i,2}$. Thus, we can solve the coefficients using the same method as in Section 4.3.2. After solving $\omega_{i,1}$ and $\omega_{i,2}$, the normal edge point rule is defined via Equation (27).

Crease edge: When the edge e is a crease edge, we represent the associated edge point as a linear combination of the two end vertices of e like in [BLZ00] which is illustrated in Figure 12(b). Specifically, we set

$$\hat{E}^{k+1}_i = (1-\omega_i)\hat{V}^k + \omega_i\hat{E}^k_i. \tag{29}$$



(a) Normal edge. (b) Crease edge.

Figure 12: *Edge point rules.*

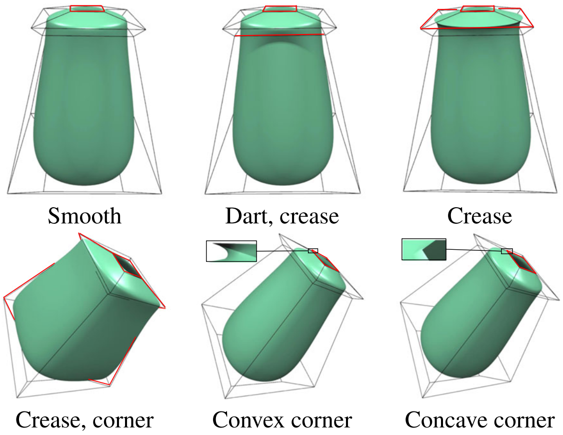


Figure 13: *Surface manipulation with various types of sharp features. Red edges in the control mesh are crease edges.*

Let $k = 0$, then

$$\hat{E}_i^1 = (1-\omega_i)\hat{V}^0 + \omega_i \hat{E}_i^0. \tag{30}$$

Denote $v_1 = \hat{E}_i^1 - \hat{E}_i^0$, $v_2 = \hat{E}_i^1 - \hat{V}^0$, it is easy to get

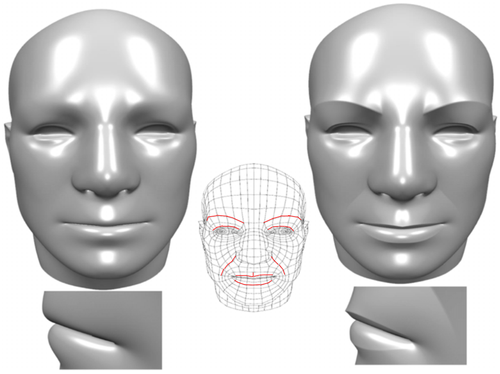
$$\omega_i = \frac{|v_1|}{|v_1| + |v_2|}. \tag{31}$$

Obviously, when applying the above subdivision scheme to a 3D mesh, the entire resulting surface is divided into several patches by the crease curves. It is easy to theoretically analyse that each surface patch is a non-uniform Catmull–Clark surface bounded by these crease curves which are non-uniform cubic B-splines. Each surface patch is reduced to a NURBS surface in the regular region. Moreover, since the knot intervals on opposing edges in a control mesh are identical, the subdivision matrix M is stationary without the influence of the updates of knot intervals. Of course, the new scheme is reduced to the uniform scheme described in Section 2 when all the knot intervals of the initial control mesh are equal.

5. Results and Discussion

In this section, we present experimental results of sharp features designed by our method and the comparisons with previous approaches are also provided. Experimental examples demonstrate that applying previous subdivision rules for arbitrary control meshes with non-uniform knots and arbitrarily specified tags yields good results.

As demonstrated in Figures 13 and 14, our method can generate surface models with various types of sharp features including darts, creases and corners. All the surfaces in Figure 13 are generated from the same control mesh by applying diverse tags. Figure 14 shows a character model, where we use different sharp features on the mouth, cheeks, eyelids and brow ridges. All these models are parameterized by the average length of the edges of the control grid [LC18].



(a) Smooth surface. (b) Crease curves.

Figure 14: *Character animation model. Red edges in the control mesh are marked as crease edges. Note that the two edges closest to nose are adjacent to an extraordinary vertex and the middle point of the upper lip is portrayed as a corner point.*

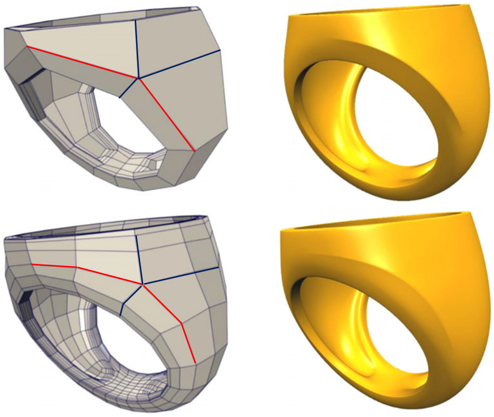


Figure 15: *Sharp features are added to two models with different tessellations. The bottom model is created by applying the non-uniform subdivision on the up one and set the corresponding edges to be crease edges. The knot intervals of dark blue edges are 1, those of red edges are 3 and the red edges are crease edges.*

Figures 15 and 16 show two real-world models. Both models are created through the commercial NURBS software–T-spline plugin of Rhinoceros. The knot information is designed from the original control meshes which are also shown in the figures. The sharp features are created using our new subdivision scheme. In Figure 15, we create sharp features for two different tessellations. The bottom model is created by applying the non-uniform subdivision on the up one and set the corresponding edges to be crease edges. We can observe that both tessellations generate satisfactory sharp features but the features of the up model are sharper.

Even for the regular region, our approach can produce sharp features where NURBS cannot do. Two regular B-spline control meshes with different feature edges are shown in Figure 17. Our new subdivision scheme can create a very smooth desirable crease curve. We should mention that these kinds of sharp features are very common in NURBS-based modelling systems.

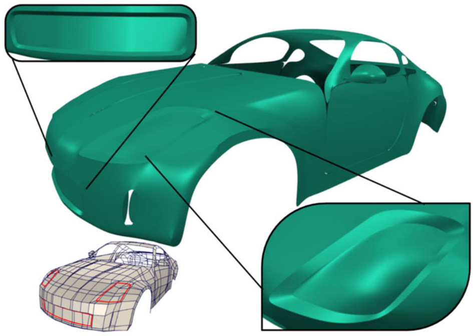
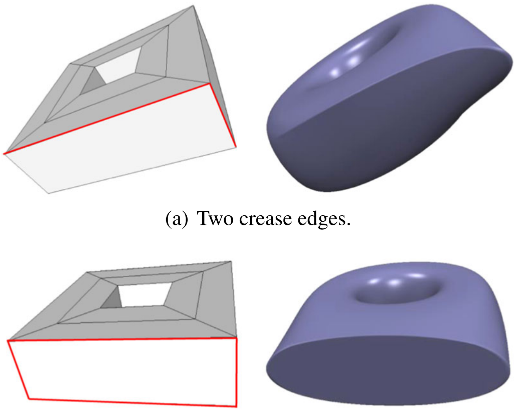


Figure 16: *Car model. The knot intervals of red edges are 3, those of other edges are 1 and the red edges are also crease edges.*



(a) Two crease edges.

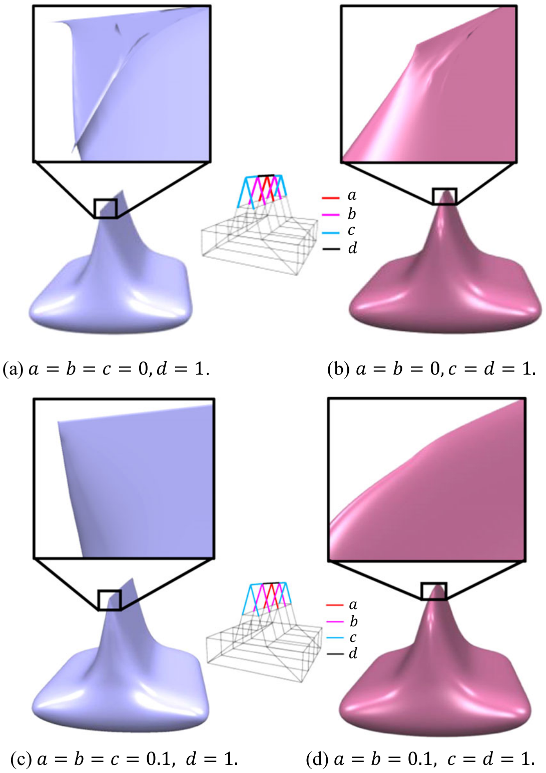
(b) Closed crease edges.

Figure 17: *The control mesh is topologically isomorphic to torus and the red edges are crease edges. However, this type of sharp features cannot be produced by NURBS.*

The approach in [SZSS98] can generate unpredictable sharp features for non-uniform parameterization. Figure 18(a) and (b) shows the results of a wedge model with different knot intervals, which produce unacceptable wrinkles. On contrast, our method produces natural and wrinkle-free sharp features for the same model, as shown in Figure 18(c) and (d).

Similar to [KSD14a] and [DKT98], our method can also produce semi-sharp features by interactively mixing the sharp rule and smooth rule throughout the subdivision process. Specifically, after subdividing a tagged initial mesh several times, we clear the tagged information in the control mesh and continue to subdivide until the limit surface is obtained. We associate each original edge with a sharpness factor s, which indicates the subdivision times before clearing the tagged information and the value of s on different edges can be different. An example of a blending function with different sharp factors is shown in Figure 19.

Our method can also be used to implement shape editing on surfaces, where crease curves generated by our method are non-uniform B-spline curves. An example is illustrated in Figure 20.



(a) $a = b = c = 0, d = 1.$ (b) $a = b = 0, c = d = 1.$

(c) $a = b = c = 0.1,\ d = 1.$ (d) $a = b = 0.1,\ c = d = 1.$

Figure 18: *A wedge model with different kinds of sharp features using the method in [SZSS98] by assigning zero knot interval to different edges (Figure (a) and (b)). The corresponding results of our method are shown in Figure (c) and (d).*

In addition, the crease curves on the subdivision surface are end-point interpolating non-uniform cubic B-splines curves and the resulting surfaces can be evaluated exactly at arbitrary parameter values based on the similar method as that in [Sta98]. This permits many algorithms and analysis techniques specifically developed for

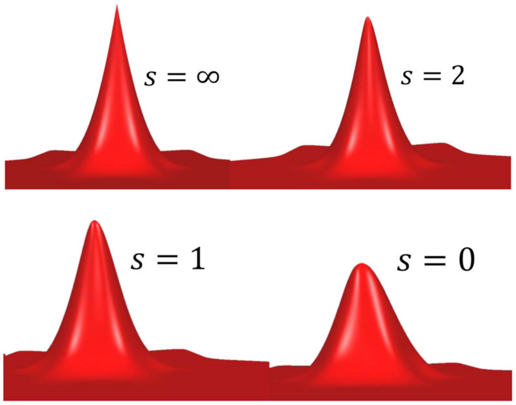
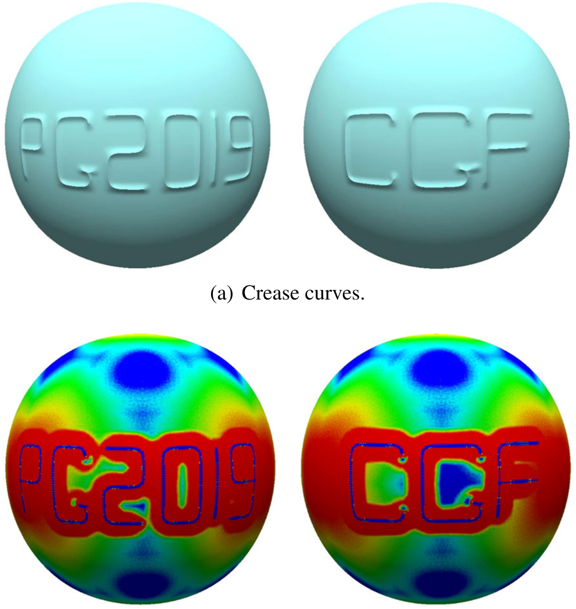


Figure 19: *Semi-sharp features generated by our method. For simplicity, the sharpness factors on the six edges adjacent to the central extraordinary vertex vary uniformly, while those on the other edges in the control mesh are always zero.*



(a) Crease curves.

(b) Gaussian curvature plots.

Figure 20: *After subdividing a cube four times, some edges in the control mesh are marked as crease edges to produce the words 'PG2019' and 'CGF'. To highlight these glyphs, we made a minor overall translation of the crease edges along the vertical direction before continuing the subdivision process.*

parametric surfaces to work for the new subdivision representation. Furthermore, we can seamlessly merge two limit surfaces because the boundary curves of the subdivision surface depend only on the boundary control points.

6. Conclusions and Future Work

Although uniform subdivision is very popular in computer graphics, but it is not compatible with NURBS representations. Thus NURBS-compatible subdivision is indispensable in order to use subdivision in CAD. On the other hand, complex sharp features are required in many CAD product design. So it is imperative to have a NURBS-compatible subdivision with this capability. In this paper, we generalize the uniform sharp feature subdivision scheme in [BLZ00] to non-uniform case using the generalized eigen-polyhedron technique. The method also supports semi-sharp features and glyph editing on surfaces. Experimental examples demonstrate that our method can generate various types of sharp features with pleasant visual effects. The subdivision surfaces are wrinkle free at the extraordinary vertices for non-uniform parameterization while the other existing approaches are not.

According to our observation, the subdivision surfaces by our method are G^1 continuous at the interior extraordinary points. However, theoretical analysis about the convergence and continuity are very challenging due to arbitrary parameterizations. And no non-uniform subdivision schemes have made any progress on this topic so far. The subdivision matrix M designed according to our approach must have two identical eigenvalues λ, which always turned out to be the second and third eigenvalues of M by many tests, but its mathematical proof can only be left as a future research.

Acknowledgements

The authors are grateful for the critical comments and suggestions to improve the manuscript by the anonymous reviewers. This work is supported by the National Natural Science Foundation of China (No. 61972368, No.61872328), SRF for ROCS SE and the Youth Innovation Promotion Association CAS.

References

[BLZ00] Biermann H., Levin A., Zorin D.: Piecewise smooth subdivision surfaces with normal control. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (2000), pp. 113–120.

[BMZB02] Biermann H., Martin I. M., Zorin D., Bernardini F.: Sharp features on multiresolution subdivision surfaces. *Graphical Models 64*, 2 (2002), 61–77.

[CC78] Catmull E., Clark J.: Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design 10*, 6 (1978), 350–355.

[DKT98] DeRose T., Kass M., Truong T.: Subdivision surfaces in character animation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (1998), pp. 85–94.

[DM01] Dan G., Marian N.: Subdivision surfaces–can they be useful for geometric modeling applications. *Technical Report 01-011, The Boeing Company* (2001).

[Flo15a] Floater M. S.: Generalized barycentric coordinates and applications. *Acta Numerica 24* (2015), 161–214.

[Flo15b] Floater M. S.: The inverse of a rational bilinear mapping. *Computer Aided Geometric Design 33* (2015), 46–50.

[HDD*94] Hoppe H., DeRose T., Duchamp T., Halstead M., Jin H., McDonald J., Schweitzer J., Stuetzle W.: Piecewise smooth surface reconstruction. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques* (1994), pp. 295–302.

[KBZ15] Kovacs D., Bisceglio J., Zorin D.: Dyadic T-mesh subdivision. *ACM Transactions on Graphics 34*, 4 (2015), 143.

[KS99] Khodakovsky A., Schroder P.: Fine level feature editing for subdivision surfaces. In *Symposium on Solid Modeling and Applications* (1999), pp. 203–211.

[KSD14a] Kosinka J., Sabin M. A., Dodgson N. A.: Semi-sharp creases on subdivision curves and surfaces. *Computer Graphics Forum 33*, 5 (2014), 217–226.

[KSD14b] Kosinka J., Sabin M. A., Dodgson N. A.: Subdivision surfaces with creases and truncated multiple knot lines. *Computer Graphics Forum 33*, 1 (2014), 118–128.

[LC18] Li X., Chang Y.: Non-uniform interpolatory subdivision surface. *Applied Mathematics and Computation 324* (2018), 239–253.

[LFS16] Li X., Finnigan T., Sederberg T. W.: G^1 non-uniform Catmull-Clark surfaces. *ACM Transactions on Graphics 35*, 4 (2016), 135.

[Loo87] Loop C.: *Smooth Subdivision Surfaces based on Triangles*. Master's thesis, University of Utah, 1987.

[Ma05] Ma W.: Subdivision surfaces for cadąłan overview. *Computer-Aided Design 37*, 7 (2005), 693–709.

[MFR*10] Müller K., Fünfzig C., Reusche L., Hansford D., Farin G., Hagen H.: Dinus: Double insertion, nonuniform, stationary subdivision surfaces. *ACM Transactions on Graphics 29*, 3 (2010), 25.

[MRF06] Müller K., Reusche L., Fellner D.: Extended subdivision surfaces: Building a bridge between NURBS and Catmull-Clark surfaces. *ACM Transactions on Graphics 25*, 2 (2006), 268–292.

[NLG12] Niessner M., Loop C., Greiner G.: Efficient evaluation of semi-smooth creases in Catmull-Clark subdivision surfaces. In *Eurographics – Short Papers* (2012), The Eurographics Association, pp. 41–44.

[NLM*19] Niessner M., Loop C., Meyer M., DeRose T., Greiner G.: http://graphics.pixar.com/opensubdiv/docs/intro.html, Accessed July 2019.

[PR08] Peters J., Reif U.: *Subdivision Methods for Geometric Design*. Morgan Kaufmann, Burlington, MA, 2008.

[Sta98] Stam J.: Exact evaluation of Catmull-Clark subdivision surfaces at arbitrary parameter values. *Siggraph 98* (1998), 395–404.

[SZSS98] Sederberg T. W., Zheng J., Sewell D., Sabin M.: Non-uniform recursive subdivision surfaces. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (1998), pp. 387–394.

[Urs09] Ursula A.: NURBS with extraordinary points: High-degree non-uniform subdivision surfaces. *ACM Transactions on Graphics 28* (2009), 1–9.

[ZML15] Zhang X., Ma X., Li M.: Sharp feature construction of non-uniform Doo-Sabin subdivision surface. *Computer Integrated Manufacturing Systems 21*, 11 (2015), 2827–2836.