EG UK Computer Graphics & Visual Computing (2022)
M. Turner and P. Vangorp (Editors)

Semi-Sharp Subdivision Shading

Jun Zhou, Jan Boonstra, and Jiří Kosinka

Bernoulli Institute, University of Groningen, the Netherlands

Abstract

Subdivision is a method for generating a limit surface from a coarse mesh by recursively dividing its faces into several smaller faces. This process leads to smooth surfaces, but often suffers from shading artifacts near extraordinary points due to the lower quality of the normal field there. The idea of subdivision shading is to apply the same subdivision rules that are used to subdivide geometry to also subdivide the normals associated with mesh vertices. This leads to smoother normal fields, which in turn removes the shading artifacts. However, the original subdivision shading method does not support sharp and semi-sharp creases, which are important ingredients in subdivision surface modelling. We present two approaches to extending subdivision shading to work also on models with (semi-)sharp creases.

CCS Concepts

• ***Computing methodologies*** → ***Rendering; Shape modeling;***

1. Introduction

The *subdivision surface* is a popular 3D modelling primitive that relies on recursively subdividing the faces of an input mesh, called the *control mesh*, often of arbitrary manifold topology. This process, in the limit, leads to a smooth surface, called the *limit surface*. The most popular subdivision schemes include the Catmull-Clark scheme [CC78] based on quad(-dominant) meshes and bi-cubic B-splines, and the Loop subdivision scheme [Loo87] based on triangular meshes and the quartic box-spline. Both schemes produce C^2 continuous limit surfaces almost everywhere, except at *extraordinary points* (EPs), where their continuity drops to G^1. EPs correspond to extraordinary vertices (EVs) in the input meshes where other than the regular number of faces meet, i.e., other than four quads in the Catmull-Clark case, and other than six triangles in the Loop case. This reduced continuity of the limit surfaces at EPs manifests itself in shading artifacts, caused by the C^0 normal field of the limit surface there.

Subdivision shading [AB08] alleviates these artifacts by producing a *fake* normal field for the limit surface. The main idea is that if the vertex normals of the control mesh are subdivided using the same rules applied to the geometry, the so-produced limit normal field will be smoother than the true normal field of the limit surface. The original method has been improved in [BBK18] by enhancing the blending scheme of the two normal fields to further increase the quality of the (fake) normal field of the limit surfaces. However, neither of the subdivision shading methods works in combination with *(semi-)sharp creases* [DKT98; LWY08]. These creases greatly enhance the modelling capability of subdivision by allowing users to tag edges as sharp or semi-sharp and then by modifying the subdivision rules in the vicinity of these edges to produce the desired limit surface with (semi-)sharp creases.

Our contribution focuses on extending the subdivision shading method of [BBK18] to closed models with (semi-)sharp creases, thus obtaining the best from both worlds: limit surfaces with (semi-)sharp creases *and* with high-quality shading everywhere. Our method is built on Catmull-Clark subdivision, but the same ideas apply also to Loop subdivision and potentially other schemes.

2. Related work

We start by briefly reviewing related work on semi-sharp creases, subdivision shading, and normal field blending.

2.1. Semi-sharp creases

Subdivision is designed in such a way that it creates limit surfaces that are smooth everywhere. However, in some cases this might be undesirable as with these fixed subdivision rules it means that the only way to have sharp edges or corners in a limit surface is by adding a lot of extra vertices around the part of the mesh that needs to become sharp. This is inefficient both computationally as well as from the design perspective. Several methods have addressed this [KMDZ09; BFK*16; KSD14b]. Based on the ideas of [HDD*94], an efficient method to achieve (semi-)sharp creases was introduced in [DKT98] using modified subdivision rules, and a different method based on multiple knots was investigated in [KSD14a]. The solution of [DKT98] is to add a sharpness value s to each edge in the control mesh. An edge with sharpness s is then treated as a sharp edge for the first s iterations of the subdivision algorithm. If $s = 0$ for an edge, this edge is smooth, and if

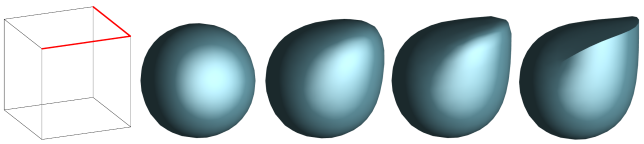


Figure 1: *A cube with two sharp edges (red) with different sharpness values s after 3 subdivision steps. From left to right: Control mesh, $s = 0$, $s = 1$, $s = 2$, and $s = 3$.*

$s = \infty$, the corresponding edge is (infinitely) sharp. Otherwise, the edge is semi-sharp.

Determining the coordinates of a subdivided vertex is now done in one of three ways, based on the number of incident sharp edges:

1. *Smooth or dart vertex*: If a vertex has fewer than two incident sharp edges, its subdivided vertex is determined using the standard Catmull-Clark subdivision rules, called the *smooth rule*.
2. *Crease vertex*: If a vertex has exactly two incident sharp edges, it is subdivided as if it were a boundary vertex. The corresponding rule is the *crease rule*, coming from the uniform cubic B-spline curve subdivision scheme.
3. *Corner vertex:* If a vertex is incident with more than two sharp edges, its coordinates do not change throughout the subdivision process. The is called the *corner rule*.

When a new edge point is created on a sharp edge, its coordinates will be determined as if its edge were a boundary edge. The method for determining face points and faces remains the same as in standard smooth Catmull-Clark subdivision; see Figure 1.

2.2. Subdivision shading

Around regular (non-extraordinary) vertices, the limit surface can be shaded using its standard *limit normal field* N_L. However at EVs, the use of N_L can lead to visual artifacts. To address this, [AB08] proposed to subdivide not only the geometry/vertices, but also the associated normals. This leads to a *subdivided normal field* N_S which is C^2 everywhere except at EPs, where it is G^1 continuous. Because of this continuity, shading a mesh using N_S rather than N_L produces results with increased observed smoothness. When determining the subdivided normal of a vertex, the same subdivision rule as for vertices is used, but with a minor complication. As the unit normals are essentially objects on the unit sphere, they are subdivided as such, using *spherical averaging* [AB08].

2.3. Normal field blending

The subdivided normal field N_S is G^1 continuous at EPs and therefore using it in shading makes the limit surface look smooth, in contrast to using the limit normal field N_L which is only C^0 at EPs and thus leads to visual shading artifacts. However, using only N_S can cause the final result to look too smooth, and thus make some of the details in the mesh disappear [AB08]. Because of this, it is generally desirable to use the subdivided normals only in the vicinity of EPs, while retaining N_L elsewhere. The original method [AB08] comes with a blending scheme for the two normal fields, but the blended result is not guaranteed to be G^1 at EPs. This has been addressed in [BBK18] by improving the blending function to achieve a globally smooth blended normal field N_B. In detail,

$$N_B = (1 - b^p)N_L + b^p N_S, \tag{1}$$

where b is a suitably chosen *blending function* and p is a variable that can be used to control the transition between N_S and N_L in the blended normal field. Following [BBK18], the default value of p is set to 1 in our results. The blending function is defined via subdivision by providing a control blend weight for each vertex. The (limit) blending function should be 1 at EPs and smoothly transition to 0 away from EPs. This causes N_B at EPs to depend solely on N_S to prevent any visual artifacts that using N_L can create. At the same time, it also means the rest of the limit surface relies (mostly) on N_L, as desired.

The challenge in this approach is determining a suitable blending function b via its control weights. Two such approaches were designed in [AB08]. In both, the control blending weights are initialized with 1 at extraordinary vertices and 0 everywhere else. During subdivision, the first approach relies on bi-linear subdivision, whereas the second approach subdivides the weights using the same subdivision scheme as applied to the vertices and normals. Neither of these approaches leads to a globally smooth N_B. Later, [BBK18] improved on this using two variants. The first relies on inverting the subdivision limit stencils to initialise the control blending weights so that their limit values at EPs are equal to 1. The second and arguably better variant, which we use to generate our results, initialises the weights with 1 at EVs as well as their one-ring neighbourhoods. Both variants achieve the desired result and lead to globally smooth blended normal fields. All four approaches are compared in [BBK18].

Although the original method [AB08] advocated spherical averaging for normals, [BBK18] showed that the difference between using spherical and standard linear averaging leads to visually nearly indistinguishable results. Our method and results thus rely solely on linear averaging when subdividing normal vectors. Moreover, by default we use subdivision shading from subdivision level 1 to balance smoothing and detail preservation. And unless stated otherwise, subdivision of blending weights also starts from level 1.

3. Subdivision shading with semi-sharp creases

We introduce two ways to combine the subdivision shading method with semi-sharp creases in Catmull-Clark subdivision. We described how subdivision affects geometry/vertices in the presence of (semi-)sharp creases and vertices in Section 2.1. We now extend that to also support the subdivision of normals at (semi-)sharp edges and vertices, as follows.

3.1. Blending at sharp vertices

We explore two ways of applying the normal field blending method (Section 2.3) to meshes containing (semi-)sharp creases. This means that the blending function definition needs to be extended to account for *sharp vertices*, i.e., vertices incident with at least two sharp edges. The two variants are as follows:

- **Variant A**: Treating sharp vertices as standard vertices. In other words, initialising extraordinary sharp vertices (and possibly their one ring neighborhood) with control blending weights that will lead to the value of 1 in the limit (see Section 2.3), and initialising regular sharp vertices with 0.
- **Variant B**: Initialising the control blending weights of sharp vertices to 0. The idea behind this is that sharp vertices are not meant to look smooth, and thus are not expected to benefit from subdivided normals.

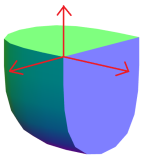
3.2. Blending at dart vertices

When subdividing the geometry of the mesh, vertices incident with just one sharp edge, *dart vertices*, are considered smooth. However, this does not automatically imply that normals should be treated the same way. We explore two options: treating dart vertices as sharp vertices, or treating them as standard vertices.

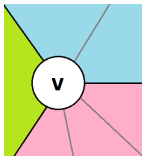
We believe that the options mentioned above cover all reasonable choices for handling infinite or semi-sharpness in subdivision meshes combined with subdivision shading.

3.3. Implementation

A standard Catmull-Clark implementation framework typically supports only one normal per vertex. However, in our method each vertex can have up to n distinct normals, where n is the number of sharp edges incident with that vertex. This is because the normals on both sides of a sharp crease can in general differ from each other, and thus to preserve the sharpness when shading the limit surface. Furthermore, we have extended this with extra subdivision rules depending on the sharpness of the edges in the control mesh and with the blending scheme as described above.



In the control mesh we assign each sharp vertex (incident with at least two sharp edges) a set of regions. Each region consists of all incident faces that are not separated from each other by a sharp edge (black in the inset). Then the vertex is assigned a normal for each of the regions (which is calculated as the average of the normals of the faces in that region if not specified by the user). When subdividing the normals, if a subdivision stencil covers a sharp vertex, it will not use the regular vertex normal of that vertex, but rather the sharp normal that is in the same region as the (subdivided) normal that is being determined. In our experimental tool, users can set sharpness values via an extended OBJ file or directly in the tool via its user interface.



4. Results

We now demonstrate our method on several results and provide a comparisons. We start with the Suzanne model in Figure 2. It shows the difference between using Catmull-Clark subdivision combined with semi-sharp creases without and with our method.

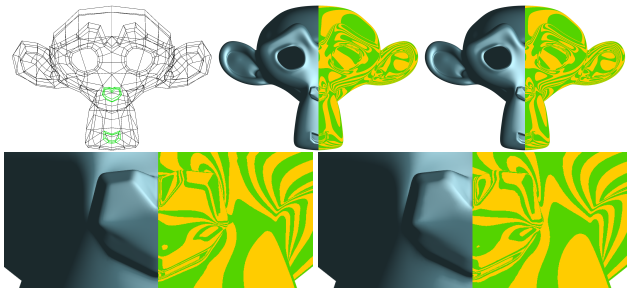


Figure 2: *The Suzanne model with semi-sharp creases coloured in green (sharpness $s = 2$). The result without subdivision shading (top middle) and with subdivision shading using Variant B (top right). Bottom row: insets in the same order. The result of Variant B is smoother; see also the supplementary video.*

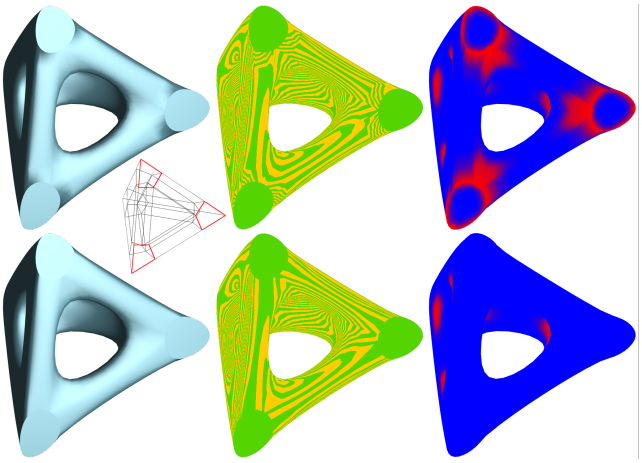


Figure 3: *The three planar rounded faces in the front of the object are completely made up of sharp edges (shown in red in the control mesh). The results in the top row are the shading, isophotes and blend weights using Variant A, and the bottom row using Variant B. In the right column, blue corresponds to the value of* 0 *and red to* 1 *of the limit blending function b; see (1). In this example, subdivision of blending weights starts from level* 3.

4.1. Blend weights at creases

We tested the two variants on models containing sharp edges. Figure 3 shows a comparison between the two methods on a model with three 'legs' composed of sharp edges. A clear difference is observable around the sharp faces. When sharp vertices are treated as standard ones (Variant A; top row), the normals misbehave around the sharp edges. This problem does not exist when sharp vertices are initialized with a blend weight of zero (Variant B; bottom row). This is even clearer in the isophote plots: in the top image, the lines form wiggles around the sharp edges, while in the bottom image the lines are smoother all the way up to the sharp edges. Figure 4 shows a comparison on an icosahedron model with three sharp edges. Towards the top-right of the shown sharp loop, the highlight fades away near the edge when shading 'regularly' (top row), while the same highlight ends neatly on the sharp edge when setting the blend weights there to 0 (bottom row).

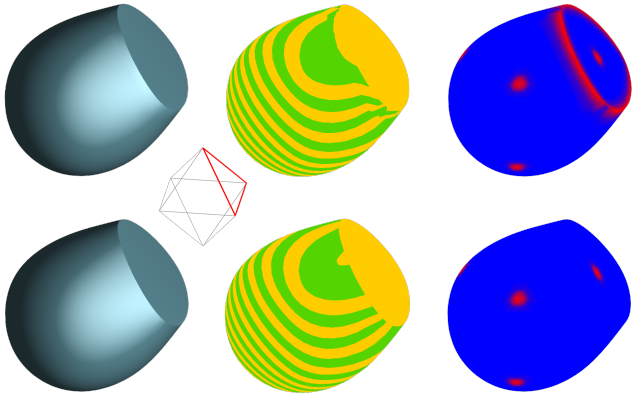


Figure 4: *A subdivided icosahedron with* 3 *sharp edges (shown in red). The images in the top row are the shading, isophotes and blend weights using Variant A, and the bottom row using Variant B. In this example, subdivision of blending weights starts from level* 3.

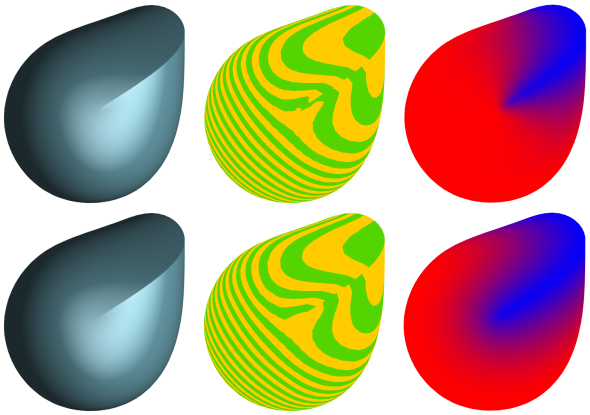


Figure 5: *The cube model with two sharp edges (see Figure 1) when darts are treated as regular vertices (top) or as sharp vertices (bottom) when setting the blend weights.*

4.2. Blend weights at darts

We subdivided models containing darts and compared the result when either treating them as sharp vertices or as non-sharp vertices when initializing the blend weights; see Figure 5. The blending algorithm produces small differences in the resulting shaded mesh around the dart. When treating the dart as a sharp vertex when subdividing the normals (bottom row), the isophotes clearly show that more sharpness is preserved around the sharp edge. However, this is typically not desirable since the sharp edge ends at the dart and therefore the normal field should transition from being sharp along the creases to smooth towards the dart. Based on this, we argue that it is better to treat darts as smooth vertices. However, if a dart is also an EV, we treat it as an EV.

4.3. Discussion

Although in general our method provides smooth shading, small visual artifacts can still appear around sharp edges, as can be seen in Figure 4. It would therefore be interesting to investigate whether this is caused by shading or rather the underlying geometry (and thus subdivision scheme).

If a model contains no infinitely sharp creases, the efficiency of our method is the same as that of the original subdivision shading relying on linear averaging of normals. However, in the case of models with sharp creases, it can no longer rely on indexed rendering due to the multiple normals being associated with some of the vertices, which is of course the case for all methods dealing with sharp creases, which in turns makes the method marginally slower.

Initialising the blend weights of all sharp (and smooth) vertices to zero means that, in general, after subdivision fewer and fewer vertices have a positive blend weight. In theory this means that the subdivided normal has to be calculated for fewer vertices since it affects the results only when the (limit) blend weight is non-zero. So we could skip the subdivided normal computation for the sharp vertices in order to improve efficiency.

It should be noted that in practice, infinitely sharp creases are rarely used as they lead to (other) issues, such as when using displacement maps; they are often replaced with semi-sharp creases with a finite sharpness value. However, this does not suggest that rules for sharp vertices/edges are no useful. To the contrary, sharp rules are still needed to be able to handle semi-sharp creases in the subdivision process.

We have presented results based on the Catmull-Clark subdivision scheme. Nevertheless, the ideas in our method directly apply to other schemes, such as Loop subdivision with (semi-)sharp creases.

5. Conclusion

We have extended subdivision shading to make it compatible with models including (semi-)sharp creases. This is a simple, yet powerful combination of subdivision shading [AB08; BBK18] and semi-sharp creases [DKT98] in Catmull-Clark subdivision.

We have proposed two variants for initialising the control blending weights of sharp vertices. Our results and observations have shown that in most cases initialising the weights to zero produces better shading in the regions near sharp creases. This makes sense since the normal blending formula tries to make a model look C^1 smooth across the entire model, while it actually by definition should not be so near sharp edges.

Furthermore, according to our experiments, darts should not be treated as sharp vertices when blending the normals. This is consistent with the way darts are treated when subdividing the geometry/vertices of the mesh, and it produces superior results.

As mentioned above, it still remains an open question whether initialising the control blending weights at sharp vertices can be utilised to increase the efficiency of the method. This, along with implementing the entire subdivision process using subdivision tables [DV21], is a promising future direction for speeding up the method. Another interesting direction is to investigate how to further reduce the appearance of minor but still existing shading artifacts in some situations near sharp creases.

References

[AB08] ALEXA, MARC and BOUBEKEUR, TAMY. "Subdivision shading". *ACM SIGGRAPH Asia 2008*. 2008, 1–4 1, 2, 4.

[BBK18] BAKKER, JELLE, BARENDRECHT, PIETER, and KOSINKA, JIŘÍ. "Smooth Blended Subdivision Shading." *Eurographics (Short Papers)*. 2018, 37–40 1, 2, 4.

[BFK*16] BRAINERD, WADE, FOLEY, TIM, KRAEMER, MANUEL, et al. "Efficient GPU rendering of subdivision surfaces using adaptive quadtrees". *ACM Transactions on Graphics* 35.4 (2016), 1–12 1.

[CC78] CATMULL, E. and CLARK, J. "Recursively generated B-spline surfaces on arbitrary topological meshes". *Computer-Aided Design* 10.6 (1978), 350–355. ISSN: 0010-4485 1.

[DKT98] DEROSE, TONY, KASS, MICHAEL, and TRUONG, TIEN. "Subdivision surfaces in character animation". *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. 1998, 85–94 1, 4.

[DV21] DUPUY, J. and VANHOEY, K. "A Halfedge Refinement Rule for Parallel Catmull-Clark Subdivision". *Computer Graphics Forum* 40.8 (2021), 57–70 4.

[HDD*94] HOPPE, HUGUES, DEROSE, TONY, DUCHAMP, TOM, et al. "Piecewise smooth surface reconstruction". *Computer graphics and interactive techniques*. 1994, 295–302 1.

[KMDZ09] KOVACS, DENIS, MITCHELL, JASON, DRONE, SHANON, and ZORIN, DENIS. "Real-time creased approximate subdivision surfaces". *Proceedings of the 2009 symposium on Interactive 3D graphics and Games*. 2009, 155–160 1.

[KSD14a] KOSINKA, JIŘÍ, SABIN, MALCOLM, and DODGSON, NEIL. "Semi-sharp Creases on Subdivision Curves and Surfaces". *Computer Graphics Forum*. Vol. 33. 5. 2014, 217–226 1.

[KSD14b] KOSINKA, JIŘÍ, SABIN, MALCOLM, and DODGSON, NEIL. "Subdivision surfaces with creases and truncated multiple knot lines". *Computer Graphics Forum*. Vol. 33. 1. 2014, 118–128 1.

[Loo87] LOOP, CHARLES. "Smooth subdivision for surfaces based on triangles". MA thesis. University of Utah, 1987 1.

[LWY08] LING, RUOTIAN, WANG, WENPING, and YAN, DONGMING. "Fitting Sharp Features with Loop Subdivision Surfaces". *Computer Graphics Forum* 27.5 (2008), 1383–1391 1.