Interpolatory Subdivision on Open Quadrilateral Nets with Arbitrary Topology

Leif Kobbelt

Department of Computer Sciences
University of Wisconsin — Madison
1210 West Dayton Street
Madison, WI 53706-1685
kobbelt@cs.wisc.edu

December 9, 1995

Abstract

A simple interpolatory subdivision scheme for quadrilateral nets with arbitrary topology is presented which generates C^1 surfaces in the limit. The scheme satisfies important requirements for practical applications in computer graphics and engineering. These requirements include the necessity to generate smooth surfaces with local creases and cusps. The scheme can be applied to open nets in which case it generates boundary curves that allow a C^0-join of several subdivision patches. Due to the local support of the scheme, adaptive refinement strategies can be applied. We present a simple device to preserve the consistency of such adaptively refined nets.

Keywords: Curve and surface modeling, Interpolatory subdivision, Adaptive mesh-refinement

1 Introduction

The problem we address in this paper is the generation of smooth interpolating surfaces of arbitrary topological type in the context of practical applications. Such applications range from the design of free-form surfaces and scattered data interpolation to high quality rendering and mesh generation, e.g., in finite element analysis. The standard set-up for this problem is usually given in a form equivalent to the following:

A *net* $N = (V, F)$ representing the input is to be mapped to a *refined net* $N' = (V', F')$ which is required to be a sufficiently close approximation of a smooth surface. In this notation the sets V and V' contain the *data points* $\mathbf{p}_i, \mathbf{p}'_i \in \mathbb{R}^3$ of the input or output respectively. The sets F and F' represent the *topological information* of the nets. The elements of F and F' are finite sequences of points $s_k \subset V$ or $s'_k \subset V'$ each of which enumerates the corners of one not necessarily planar *face* of a net.

If all elements $s_k \in F$ have length four then N is called a *quadrilateral net*. To achieve interpolation of the given data, $V \subset V'$ is required. Due to the geometric background of the problem we assume N to be *feasible*, i.e., at each point $\mathbf{p}_i$ there exists a plane T_i such that the projection of the faces meeting at $\mathbf{p}_i$ onto T_i is injective. A net is *closed* if every edge is part of exactly two faces. In *open* nets, boundary edges occur which belong to one face only.

There are two major 'schools' for computing N' from a given N. The first or classic way of doing this is to explicitely find a collection of local (piecewise polynomial) parametrizations (*patches*) corresponding to the faces of N. If these patches smoothly join at common boundaries they form an overall smooth patch complex. The net N' is then obtained by sampling each patch on a sufficiently fine grid. The most important step in this approach is to find smoothly joining patches which represent a surface of arbitrary topology. A lot of work has been done in this field, e.g., [Pet90], [Loo94], [Pet95] ...

Another way to generate N' is to define a *refinement operator* $\mathcal{S}$ which directly maps nets to nets without constructing an explicit parametrization of a surface. Such an operator performs both, a *topological* refinement

of the net by splitting the faces and a *geometric* refinement by determining the position of the new points in order to reduce the angles between adjacent faces (*smoothing*). By iteratively applying $\mathcal{S}$ one produces a sequence of nets N_i with $N_0 = N$ and $N_{i+1} = \mathcal{S}\, N_i$. If $\mathcal{S}$ has certain properties then the sequence $\mathcal{S}^i\, N$ converges to a smooth limiting surface and we can set $N' := \mathcal{S}^k\, N$ for some sufficiently large k. Algorithms of this kind are proposed in [CC78], [DS78], [Loo87], [DGL90], [DLL92], [HKD93] ...

The scheme which we present here is a *stationary refinement scheme* [Dyn91], [CDM91]. The rules to compute the positions of the new points use simple affine combinations of points from the unrefined net. They are derived from a modification of the well-known four-point scheme [DGL87]. This scheme refines polgons by $\mathcal{S} : (\mathbf{p}_i) \mapsto (\mathbf{p}'_i)$ with

$$\mathbf{p}'_{2i+1} \;:=\; \frac{8+\omega}{16}\,(\mathbf{p}_i + \mathbf{p}_{i+1}) - \frac{\omega}{16}\,(\mathbf{p}_{i-1} + \mathbf{p}_{i+2}) \quad \text{and} \quad \mathbf{p}'_{2i} \;:=\; \mathbf{p}_i \tag{1}$$

where $0 < \omega < 2\,(\sqrt{5} - 1)$ is sufficient to ensure convergence to a smooth limiting curve [DL90]. The standard value is $\omega = 1$ for which the scheme has cubic precision. In order to minimize the number of special cases, we restrict ourselves to the refinement of quadrilateral nets. The faces are split as shown in Fig. 1 and hence, to complete the definition of the operator $\mathcal{S}$, we need rules for new points corresponding to edges and/or faces of the unrefined net. To generalize the algorithm for interpolating arbitrary nets, a precomputing step is needed (cf. Sect. 2).

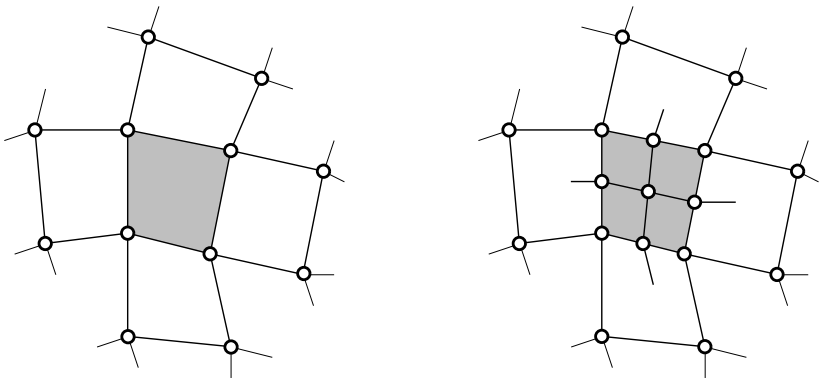

Figure 1: The refinement operator splits one quadrilateral face into four. The new vertices can be associated with the edges and faces of the unrefined net. All new vertices have valency four.

The major advantages that this scheme offers, are that it has the interpolation property *and* works on quadrilateral nets. This seems to be most appropriate to engineering applications (compared to non-interpolatory schemes or triangular nets), e.g., in finite element analysis [Sch95]. The scheme provides the maximum flexibility since it can be applied to *open* nets with *arbitrary* topology. It produces smooth surfaces and yields the possibility to generate local creases and cusps. Since the support of the scheme is local, adaptive refinement strategies can be applied. We present a technique to keep adaptively refined nets C^0-consistent (cf. Sect. 6) and shortly describe an appropriate data structure for the implementation of the algorithm.

2 Precomputing: Conversion to Quadrilateral Nets

It is a fairly simple task to convert a given arbitrary net $\tilde{N}$ into a quadrilateral net N. One straightforward solution is to apply one single *Catmull-Clark-type* split $\mathcal{C}$ [CC78] to every face (cf. Fig. 2). This split operation divides every n-sided face into n quadrilaterals and needs the position of newly computed *face-points* and *edge-points* to be well-defined. The vertices of $\tilde{N}$ remain unchanged. The number of faces in the modified net N equals the sum of the lengths of all sequences $s_k \in \widetilde{F}$.

The number of faces in the quadrilateralized net N can be reduced by half if the net $\tilde{N}$ is closed, by not applying $\mathcal{C}$ but rather its (topological) square root $\sqrt{\mathcal{C}}$, i.e., a refinement operator whose double application is equivalent to one application of $\mathcal{C}$ (cf. Fig. 2). For this split, only new *face-points* have to be computed. For open nets, the $\sqrt{\mathcal{C}}$-split modifies the boundary polygon in a non-intuitive way. Hence, one would have to handle several special cases with boundary triangles if one is interested in a well-behaved boundary curve of the resulting surface.

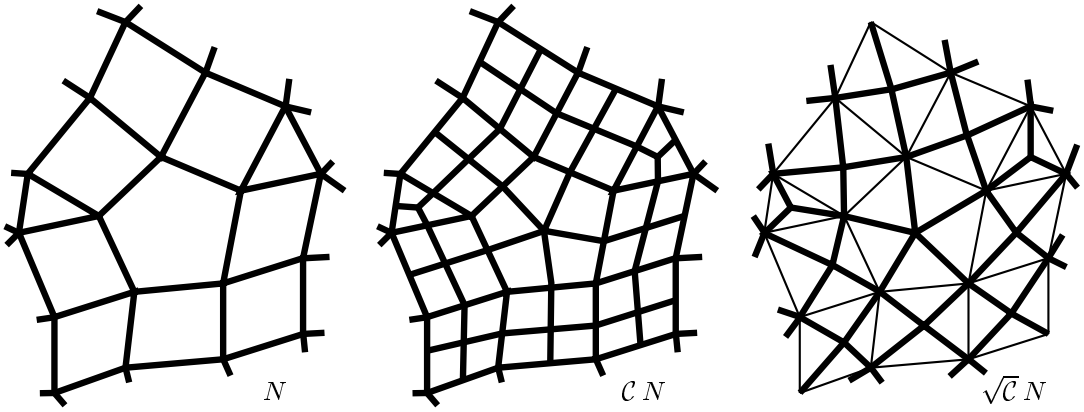

Figure 2: Transformation of an arbitrary net $\tilde{N}$ into a quadrilateral net N by one Catmull-Clark-split $\mathcal{C}$ (middle) or by its square root (right, for closed nets).

3 Subdivision Rules for Closed Nets with Arbitrary Topology

The topological structure of any quadrilateral net after several applications of a uniform refinement operator consists of large regular regions with isolated singularities which correspond to the non-regular vertices of the initial net (cf. Fig. 3). By *topological regularity* we mean a tensor product structure with four faces meeting at every vertex. The natural way to define refinement operators for quadrilateral nets is therefore to modify a tensor product scheme such that special rules for the vicinity of non-regular vertices are found. In this paper we will use the interpolatory four-point scheme [DGL87] in its tensor product version as the basis for the modification.

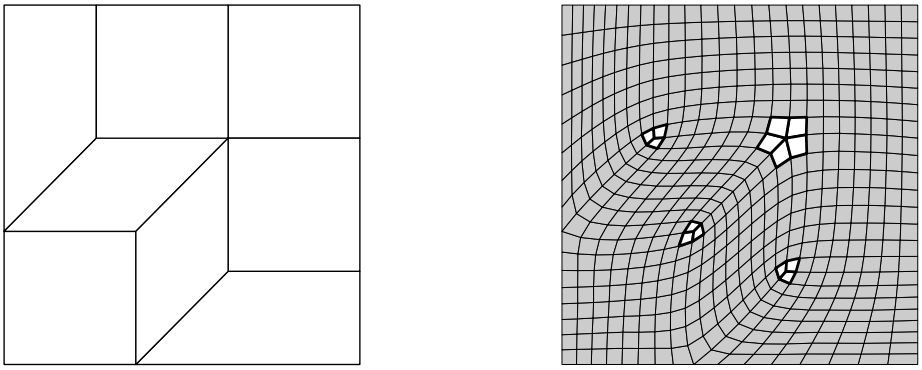

Figure 3: Isolated singularities in the refined net.

Consider a portion of a regular quadrilateral net with vertices $\mathbf{p}_{i,j}$. The vertices can be indexed locally such that each face is represented by a sequence $s_{i,j} = \{\mathbf{p}_{i,j}, \mathbf{p}_{i+1,j}, \mathbf{p}_{i+1,j+1}, \mathbf{p}_{i,j+1}\}$. The points $\mathbf{p}'_{i,j}$ of the refined net can be classified into three disjunct groups. The *vertex-points* $\mathbf{p}'_{2i,2j} := \mathbf{p}_{i,j}$ are fixed due to the interpolation requirement. The *edge-points* $\mathbf{p}'_{2i+1,2j}$ and $\mathbf{p}'_{2j,2i+1}$ are computed by applying the four-point rule (1) in the corresponding grid direction, e.g.,

$$\mathbf{p}'_{2i+1,2j} := \frac{8+\omega}{16}\,(\mathbf{p}_{i,j} + \mathbf{p}_{i+1,j}) - \frac{\omega}{16}\,(\mathbf{p}_{i-1,j} + \mathbf{p}_{i+2,j}). \tag{2}$$

Finally, the *face-points* $\mathbf{p}'_{2i+1,2j+1}$ are computed by applying the four-point rule to either four consecutive edge-points $\mathbf{p}'_{2i+1,2j-2}, \ldots, \mathbf{p}'_{2i+1,2j+4}$ or $\mathbf{p}'_{2i-2,2j+1}, \ldots, \mathbf{p}'_{2i+4,2j+1}$. The resulting weight coefficient masks for these rules are shown in Fig. 4. From the differentiability of the limiting curves generated by the four-point scheme, the smoothness of the limiting surfaces generated by infinitely refining a regular quadrilateral net, follows immediately. This is a simple tensor product argument.

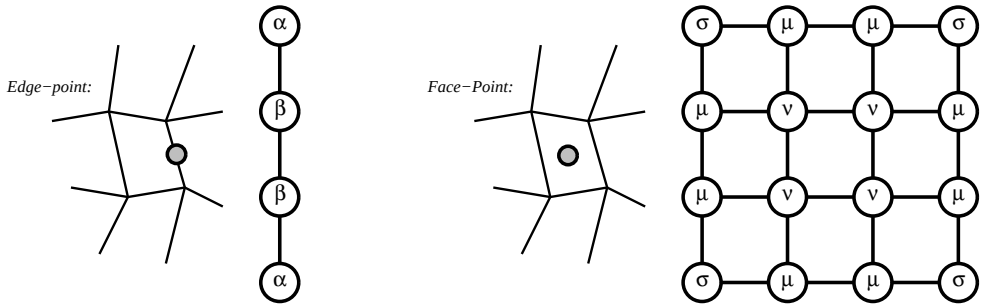

Figure 4: Subdivision masks for regular regions with $\alpha = -\frac{\omega}{16}$, $\beta = \frac{8+\omega}{16}$ and $\sigma = \alpha^2$, $\mu = \alpha\,\beta$, $\nu = \beta^2$.

For the refinement of irregular quadrilateral nets, i.e., nets which include some vertices where other than four faces meet, a consistent indexing which allows the application of the above rules is impossible. If other than four edges meet at one vertex, it is not clear how to choose the four points to which one can apply the above rule for computing the edge-points. However, once all the edge-points are known, there always are exactly two possibilities to choose four consecutive edge-points when computing a certain face-point since the net is quadrilateral. It is an important property of tensor product schemes on regular nets that both possibilities lead to the same result (commuting univariant refinement operators). In order to modify the tensor product scheme as little as possible while generalizing it to be applicable for nets with arbitrary topology, we want to conserve this property. Hence, we will propose a subdivision scheme which only needs *one* additional rule: the one for computing edge-points corresponding to edges adjacent to a non-regular vertex. All other edge-points and all face-points are computed by the application of the original four-point scheme and the additional rule will be such that both possibilities for the face-points yield the same result.

We use the notation of Fig. 5 for points in the neighborhood of a singular vertex $\mathbf{p}$. The index i is taken to be *modulo* n where n is the number of edges meeting at $\mathbf{p}$. Applying the original four-point rule wherever possible leaves only the points $\mathbf{x}_i$ and $\mathbf{y}_i$ undefined. If we require that both possible ways to compute $\mathbf{y}_i$ by applying the standard four-point rule to succeding edge-points lead to the same result, we get a dependence relating $\mathbf{x}_{i+1}$ to $\mathbf{x}_i$

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \frac{w}{8}\,(\mathbf{h}_i - \mathbf{h}_{i+1}) + \frac{w^2}{8\,(4+w)}\,(\mathbf{k}_{i-2} - \mathbf{k}_{i+2}) + \frac{w}{8}\,(\mathbf{l}_{i+2} - \mathbf{l}_{i-1}) + \frac{4+w}{8}\,(\mathbf{l}_{i+1} - \mathbf{l}_i),$$

which can be considered as compatibility condition. In the regular case, this condition is satisfied for any tensor product rule. The compatibility uniquely defines the cyclic differences $\triangle\mathbf{x}_i = \mathbf{x}_{i+1} - \mathbf{x}_i$ which sum to zero (*telescoping sums*). Hence, there always exists a solution and even one degree of freedom is left for the definition of the $\mathbf{x}_i$.

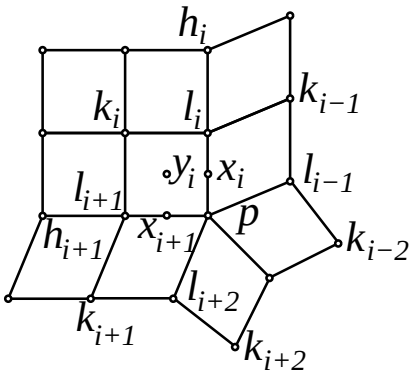

Figure 5: Notation for vertices around a singular vertex P.

The points $\mathbf{x}_i$ will be computed by rotated versions of the same subdivision mask. Thus, the vicinity of $\mathbf{p}$ will

become more and more symmetric while refinement proceeds. Hence, the distance between $\mathbf{p}$ and the center of gravity of the $\mathbf{x}_i$ will be a good measure for the roughness of the net near $\mathbf{p}$ and the rate by which this distance tends to zero can be understood as the 'smoothing rate'. The center of gravity in the regular ($n = 4$) case is:

$$\frac{1}{n}\sum_{i=0}^{n-1}\mathbf{x}_i = \frac{4+w}{8}\,\mathbf{p} + \frac{1}{2n}\sum_{i=0}^{n-1}\mathbf{l}_i - \frac{w}{8n}\sum_{i=0}^{n-1}\mathbf{h}_i.$$

In the non-regular case, we have

$$\frac{1}{n}\sum_{i=0}^{n-1}\mathbf{x}_i \;=\; \mathbf{x}_j + \frac{1}{n}\sum_{i=0}^{n-2}(n-1-i)\,\triangle\mathbf{x}_{i+j}, \qquad j \in \{0,\ldots,n-1\}.$$

Combining common terms in the telescoping sum and equating the right hand sides of both formulas leads to

$$\mathbf{x}_j = -\frac{w}{8}\,\mathbf{h}_j + \frac{4+w}{8}\,\mathbf{l}_j + \frac{4+w}{8}\,\mathbf{p} - \frac{w}{8}\,\mathbf{v}_j, \tag{3}$$

where we define the *virtual point*

$$\begin{aligned}\mathbf{v}_j := \frac{4}{n}\sum_{i=0}^{n-1}\mathbf{l}_i - (\mathbf{l}_{j-1}+\mathbf{l}_j+\mathbf{l}_{j+1}) + \\ \frac{w}{4+w}\,(\mathbf{k}_{j-2}+\mathbf{k}_{j-1}+\mathbf{k}_j+\mathbf{k}_{j+1}) - \frac{4w}{(4+w)\,n}\sum_{i=0}^{n-1}\mathbf{k}_i.\end{aligned} \tag{4}$$

Hence, the $\mathbf{x}_j$ can be computed by applying (1) to the four points $\mathbf{h}_j$, $\mathbf{l}_j$, $\mathbf{p}$ and $\mathbf{v}_j$. The formula also holds in the case $n = 4$ where $\mathbf{v}_j = \mathbf{l}_{j+2}$. Such a virtual point $\mathbf{v}_j$ is defined for every edge and both of its endpoints. Hence to refine an edge which connects two singular vertices $\mathbf{p}_1$ and $\mathbf{p}_2$, we first compute the two virtual points $\mathbf{v}_1$ and $\mathbf{v}_2$ and then apply (1) to $\mathbf{v}_1$, $\mathbf{p}_1$, $\mathbf{p}_2$ and $\mathbf{v}_2$. If all edge-points $\mathbf{x}_j$ are known, the refinement operation can be completed by computing the face-points $\mathbf{y}_j$. These are well defined since the auxillary edge-point rule is constructed such that both possible ways lead to the same result.

4 Convergence Analysis

The subdivision scheme proposed in the last section is a stationary scheme and thus the convergence criteria of [BS88] and [Rei95] can be applied. In the regular regions of the net (which enlarge during refinement), the smoothness of the limiting surface immediately follows from the smoothness of the curves generated by the univariate four-point scheme. Hence to complete the convergence analysis, it is sufficient to look at the vicinities of the finitely many isolated singular vertices (cf. Fig. 3).

Let $\mathbf{p}_0,\ldots,\mathbf{p}_k$ be the points from a fixed neighborhood of the singular vertex $\mathbf{p}_0$. The size of the considered neighborhood depends on the support of the underlying tensor product scheme and contains 5 'rings' of faces around $\mathbf{p}_0$ in our case. The collection of all rules to compute the new points $\mathbf{p}'_0,\ldots,\mathbf{p}'_k$ of the same 'scaled' (5-layer-) neighborhood of $\mathbf{p}_0 = \mathbf{p}'_0$ in the refined net can be represented by a block-circulant matrix $\mathbf{A}$ such that $(\mathbf{p}'_i)_i = \mathbf{A}\,(\mathbf{p}_i)_i$. This matrix is called the *refinement matrix*. After [BS88] and [Rei95] the convergence analysis can be reduced to the analysis of the eigenstructure of $\mathbf{A}$. For the limiting surface to have a unique tangent plane at $\mathbf{p}_0$ it is sufficient that the leading eigenvalues of $\mathbf{A}$ satisfy

$$\lambda_1 = 1, \quad 1 > \lambda_2 = \lambda_3, \quad |\lambda_2| > |\lambda_i|, \forall i \geq 4.$$

Table 1 shows theses eigenvalues of the refinement matrix $\mathbf{A}$ for vertices with n adjacent edges in the standard case $\omega = 1$. The computation of the spectrum can be done by exploiting the block-circulant structure of $\mathbf{A}$. We omit the details here, because the dimension of $\mathbf{A}$ is $k \times k$ with $k = 30\,n + 1$.

In addition to a uniquely defined tangent plane we also have to have local injectivity in order to guarantee the regularity of the surface. This can be checked by looking at the natural parametrization of the surface at $\mathbf{p}_0$ which is spanned by the eigenvectors of $\mathbf{A}$ corresponding to the subdominant eigenvalues λ_2 and λ_3. The injectivity of this parametrization is a sufficient condition. The details can be found in [Rei95]. Fig. 6 shows meshes of 'isolines' of these characteristic maps which are well-behaved.

n	λ_1	λ_2	λ_3	$\lambda_{i\geq 4} \leq$
3	1.0	0.42633	0.42633	0.25
4	1.0	0.5	0.5	0.25
5	1.0	0.53794	0.53794	0.36193
6	1.0	0.55968	0.55968	0.42633
7	1.0	0.5732	0.5732	0.46972
8	1.0	0.58213	0.58213	0.5
9	1.0	0.58834	0.58834	0.52180

Table 1: Leading eigenvalues of the subdivision matrix

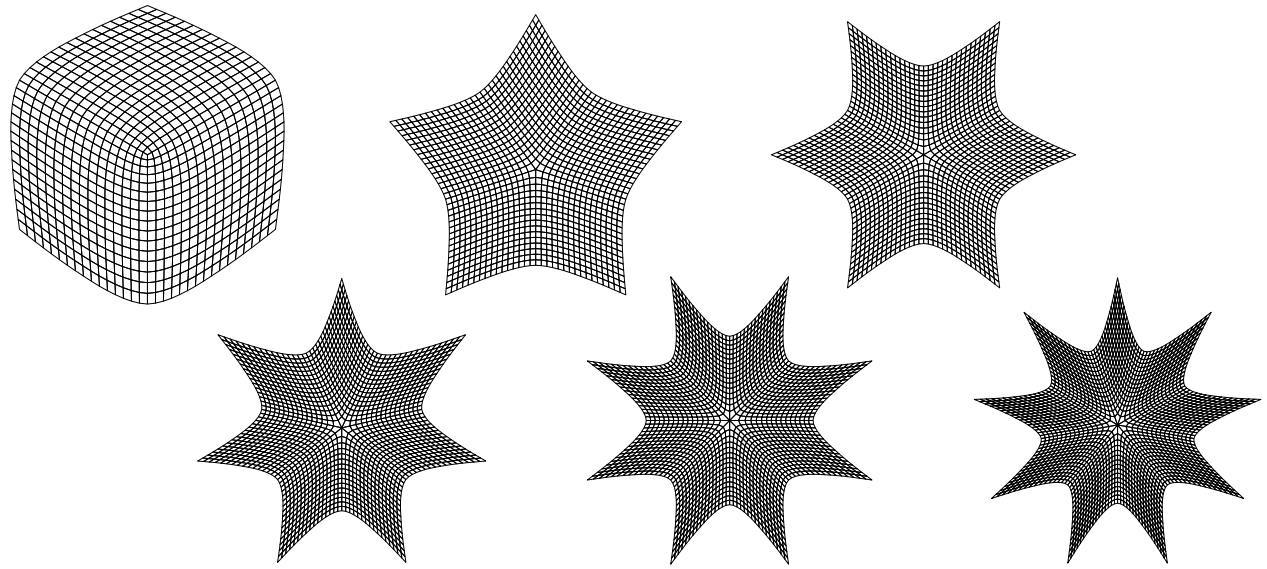

Figure 6: Sketch of the characteristic maps in the neighborhood of singular vertices with $n = 3, 5, \ldots, 9$.

5 Boundary Curves

If a subdivision scheme is supposed to be used in practical modeling or reconstruction applications, it must provide features that allow the definition of creases and cusps [Hop94]. These requirements can be satisfied if the scheme includes special rules for the refinement of *open* nets which yield well-behaved boundary curves that interpolate the boundary polygons of the given net. Having such a scheme, creases can be modeled by joining two separate subdivision surfaces along a common boundary curve and cusps result from a topological hole in the initial net which geometrically shrinks to a single point (cf. Fig. 7).

To allow a C^0-join between two subdivision patches whose initially given nets have a common boundary polygon, it is necessary that their limiting boundary curves only depend on these common points, i.e., they must not depend on any interior point. For our scheme, we achieve this by simply applying the original univariate four-point rule to boundary polygons. Thus, the boundary curve of the limiting surface is exactly the four-point curve which is defined by the initial boundary polygon. Further, it is necessary to not only generate *smooth* boundary curves but rather to allow *piecewise* smooth boundary curves, e.g., in cases where more than two subdivision patches meet at a common point. In this case we have to cut the boundary polygon into several segments by marking some vertices on the boundary as being *corner vertices*. Each segment between two corner vertices is then treated separately as an open polygon.

When dealing with open polygons, it is not possible to refine the first or the last edge by the original four-point scheme since rule (1) requires a well-defined 2-neighborhood. Therefore, we have to find another rule for the point $\mathbf{p}_1^{m+1}$ which subdivides the edge $\overline{\mathbf{p}_0^m\,\mathbf{p}_1^m}$. We define an *extrapolated* point $\mathbf{p}_{-1}^m := 2\,\mathbf{p}_0^m - \mathbf{p}_1^m$. The point $\mathbf{p}_1^{m+1}$ then results from the application of (1) to the subpolygon $\mathbf{p}_{-1}^m, \mathbf{p}_0^m, \mathbf{p}_1^m, \mathbf{p}_2^m$. Obviously, this additional

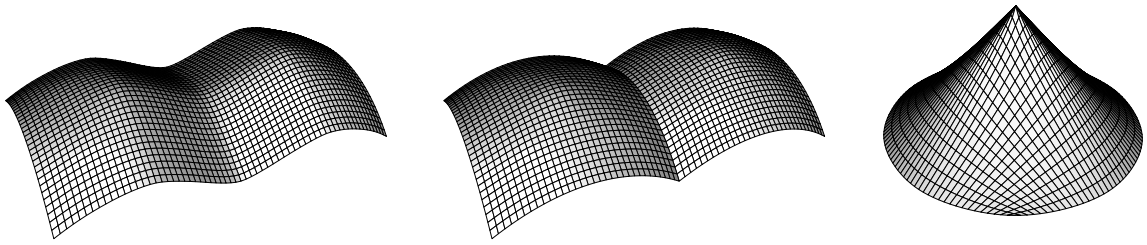

Figure 7: Modeling sharp features (piecewise smooth boundary, crease, cusp)

rule can be expressed as a stationary linear combination of points from the non-extrapolated open polygon:

$$\mathbf{p}_1^{m+1} \;:=\; \frac{8-w}{16}\,\mathbf{p}_0^m + \frac{8+2\,w}{16}\,\mathbf{p}_1^m - \frac{w}{16}\,\mathbf{p}_2^m \tag{5}$$

The rule to compute the point $\mathbf{p}_{2n-1}^{m+1}$ subdividing the last edge $\overline{\mathbf{p}_{n-1}^m\,\mathbf{p}_n^m}$ is defined analogously.

This modification of the original scheme does not affect the convergence to a continuously differentiable limit, because the estimates for the contraction rate of the maximum second forward difference used in the convergence proof of [DGL87] remain valid. This is obvious since the extrapolation only adds the zero component $\triangle^2 p_{-1}^m$ to the sequence of second order forward differences. The main convergence criterion of [Kob95] also applies.

It remains to define refinement rules for inner edges of the net which have one endpoint on the boundary and for faces including at least one boundary vertex. To obtain these rules we use the same heuristic as in the univariate case. We extrapolate the unrefined net over every boundary edge to get an additional layer of faces. When computing the egde- and face-points refining the original net by the rules from Sect. 3, these additional points can be used. To complete the refinement step, the extrapolated faces are finally deleted.

Let $\mathbf{q}_1, \ldots, \mathbf{q}_r$ be the *inner* points of the net which are connected to the boundary point $\mathbf{p}$ then the extrapolated point will be

$$\mathbf{p}^* \;:=\; 2\,\mathbf{p} - \frac{1}{r}\sum_{i=1}^{r}\mathbf{q}_i.$$

If the boundary point $\mathbf{p}$ belongs to the face $s = \{\mathbf{p}, \mathbf{q}, \mathbf{u}, \mathbf{v}\}$ and is not connected to any inner vertex then we define $\mathbf{p}^* := 2\,\mathbf{p} - \mathbf{u}$. For every boundary edge $\overline{\mathbf{p}\,\mathbf{q}}$ we add the extrapolated face $s^* = \{\mathbf{p}, \mathbf{q}, \mathbf{q}^*, \mathbf{p}^*\}$.

Again, the tangent-plane continuity of the resulting limiting surface can be proved by the sufficient criteria of [BS88] and [Rei95]. This is obvious since for a fixed number of interior edges adjacent to some boundary vertex $\mathbf{p}$, the refinement of the extrapolated net can be rewritten as a set of stationary refinement rules which define the new points in the vicinity of $\mathbf{p}$ as linear combinations of points from the non-extrapolated net. However the refinement matrix is no longer block-circulant.

At every surface point lying on the boundary of a tangent plane continuous surface, one tangent direction is determined by the tangent of the boundary curve (which in this case is a four-point curve that does not depend on inner vertices). On boundaries, we can therefore drop the requirement of [Rei95] that the leading eigenvalues of the refinement matrix have to be equal. This symmetry is only a consequence of the assumption that the rules to compute the new points around a singular vertex are identical modulo rotations (block-circulant refinement matrix). Although $\lambda_2 \neq \lambda_3$ causes an increasing local distortion of the net, the smoothness of the limiting surface is not affected. This effect can be viewed as a reparametrization in one direction. (Compare this to the distortion of a regular net which is refined by binary subdivision in one direction and trinary in the other.)

We summarize the different special cases which occur when refining an open net by the given rules. In Fig. 8 the net to be refined consists of the solid white faces while the extrapolated faces are drawn transparently. The dark vertex is marked as a corner vertex. We have to distinguish five different cases:

A: Within boundary segments, we apply (1) to four succeeding boundary vertices.

B: To the first and the last edge of an open boundary segment, we apply the special rule (5).

C: Inner edge-points can be computed by application of (3). If necessary, extrapolated points are involved.

D: For every face-point of this class, at least one sequence of four C-points can be found to which (1) can be applied. If there are two possibilities for the choice of these points then both lead to the same result which is guaranteed by the construction of (3).

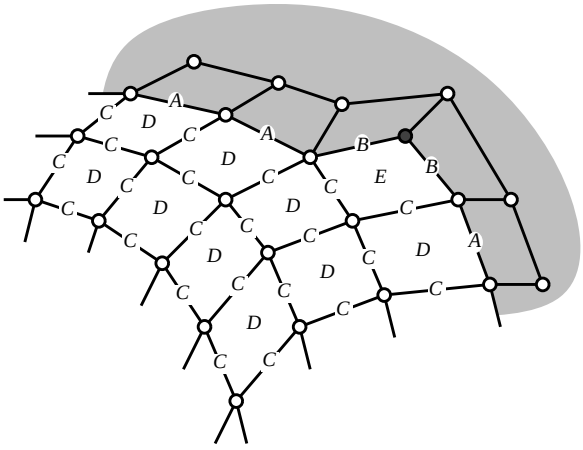

Figure 8: Occurences of the different special cases.

E: In this case no appropriate sequence of four C-points can be found. Therefore, one has to apply (5) to a B-point and the two C-points following on the opposite side of the corner face. In order to achieve independence of the grid direction, even in case the corner vertex is not marked, we apply (5) in both directions and compute the average of the two results.

6 Adaptive Refinement

In most numerical applications, the exponentially increasing number of vertices and faces during the iterative refinement only allows a small number of refinement steps to be computed. If high acuracy is needed, e.g., in finite element analysis or high quality rendering, it is usually sufficient to perform a high resolution refinement in regions with high curvature while 'flat' regions may be approximated rather coarsely. Hence, in order to keep the amount of data reasonable, the next step is to introduce adaptive refinement features.

The decision *where* high resolution refinement is needed, strongly depends on the underlying application and is not discussed here. The major problem one always has to deal with when adaptive refinement of nets is performed is to handle or eliminate C^{-1}-inconsistencies which occur when faces from different refinement levels meet. A simple trick to repair the resulting triangular holes is to split the bigger face into three quadrilaterals in an Y-fashion (cf. Fig 9). However this Y-split does not repair the hole. Instead it shifts the hole to an adjacent edge. Only combining several Y-elements such that they build a 'chain' connecting two inconsistencies leads to an overall consistent net. The new vertices necessary for the Y-splits are computed by the rules of Sect. 3. The fact that every Y-element contains a singular ($n = 3$) vertex causes no problems for further refinement because this Y-element is only of temporary nature, i.e., if any of its three faces or any neighboring face is to be split by a following local refinement adaption, then first the Y-split is undone and a proper Catmull-Clark-type split is performed before proceeding. While this simple technique seems to be known in the engineering community, the author is not aware of any reference where the theoretical background for this technique is derived. Thus, we sketch a simple proof that shows under which conditions this technique applies.

First, in order to apply the Y-technique we have to restrict the considered nets to *balanced* nets. These are adaptively refined nets (without Y-elements) where the refinement levels of neighboring faces differ at most by one. Non-balanced inconsistencies can not be handled by the Y-technique. Hence, looking at a particular face s from the n-th refinement level, all faces having at least one vertex in common with s are from the levels $(n-1)$, n, or $(n+1)$. For the proof we can think of first repairing all inconsistencies between level $n-1$ and n and then proceed with higher levels. Thus, without loss of generality, we can restrict our considerations to a situation where all relevant faces are from level $(n-1)$ or n.

A *critical* edge is an edge, where a triangular hole occurs due to different refinement levels of adjacent faces. A sequence of Y-elements can always be arranged such that two critical edges are connected, e.g., by surrounding one endpoint of the critical edge with a 'corona' of Y-elements until another critical edge is reached (cf. Fig. 10). Hence, on closed nets, we have to require the number of critical edges to be even. (On open nets, any boundary

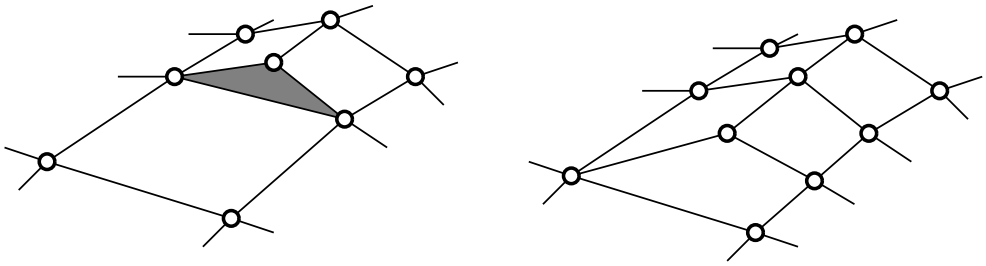

Figure 9: A hole in an adaptively refined net and an Y-element to fill it.

edge can stop a chain of Y-elements.) We show that this is always satisfied, by induction over the number of faces from the n-th level within an environment of $(n-1)$-faces. Faces from generations $> n$ or $< (n-1)$ do not affect the situation since we assume the net to be balanced.

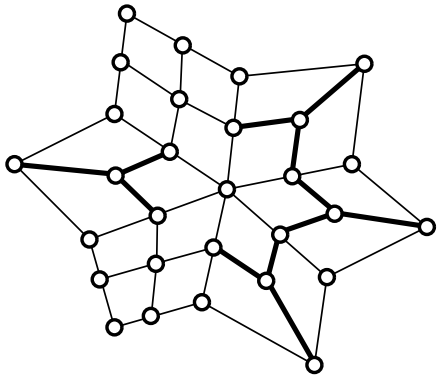

Figure 10: Combination of Y-elements

The first adaptive Catmull-Clark-type split on a uniformly refined net produces four critical edges. Every succeeding split changes the number of critical edges by an even number between -4 and 4, depending on the number of direct neighbors that have been split before. Thus the number of critical edges is always even. However, the n-faces might form a ring having in total an even number of critical edges which are separated into an odd number 'inside' and an odd number 'outside'. It turns out that this cannot happen: Let the inner region surrounded by the ring of n-faces consist of r quadrilaterals having a total number of $4r$ edges which are candidates for being critical. Every edge which is shared by two such quadrilaterals reduces the number of candidates by two and thus the number of boundary edges of this inner region is again even.

The only situation where the above argument is not valid, occurs when the considered net is open and has a hole with an odd number of boundary edges. In this case, every loop of n-faces enclosing this hole will have an odd number of critical edges on each side. Hence, we have to further restrict the class of nets to which we can apply the Y-technique to *open balanced nets which have no hole with an odd number of edges*. This restriction is not serious because one can transform any given net in order to satisfy this requirement by applying an *initial uniform refinement step* before adaptive refinement is started. Such an initial step is needed anyway if a given arbitrary net has to be transformed into a quadrilateral one (cf. Sect. 2).

It remains to find an *algorithm* to place the Y-elements correctly, i.e., to decide which critical edges should be connected by a corona. This problem is not trivial because interference between the Y-elements building the 'shores' of two 'islands' of n-faces lying close to each other, can occur. We describe an algorithm which only uses local information and decides the orientation separately for each face instead of 'marching' around the islands.

The initially given net (level 0) has been uniformly refined once before the adaptive refinement begins (level 1). Let every vertex of the adaptively refined net be associated with the generation in which it was introduced. Since all faces of the net are the result of a Catmull-Clark-type split (no Y-elements have been placed so far), they all have the property that three of its vertices belong to the same generation g and the fourth vertex belongs to a generation $g' < g$. This fact yields a unique orientation for every face. The algorithm starts by marking

all vertices of the net which are endpoints of a critical edge, i.e. if a $(n-1)$-face $\{\mathbf{p}, \mathbf{q}, \ldots\}$ meets two n-faces $\{\mathbf{p}, \mathbf{r}, \mathbf{s}, \ldots\}$ and $\{\mathbf{q}, \mathbf{r}, \mathbf{s}, \ldots\}$ then $\mathbf{p}$ and $\mathbf{q}$ are marked. After the *marking-phase*, the Y-elements are placed. Let $s = \{\mathbf{p}, \mathbf{q}, \mathbf{u}, \mathbf{v}\}$ be a face of the net where $\mathbf{p}$ is the unique vertex which belongs to an elder generation than the other three. If neither $\mathbf{q}$ nor $\mathbf{v}$ are marked then no Y-element has to be placed within this face. If only one of them is marked then the Y-element has to be oriented as shown in Fig. 11 and if both are marked this face has to be refined by a proper Catmull-Clark-type split.

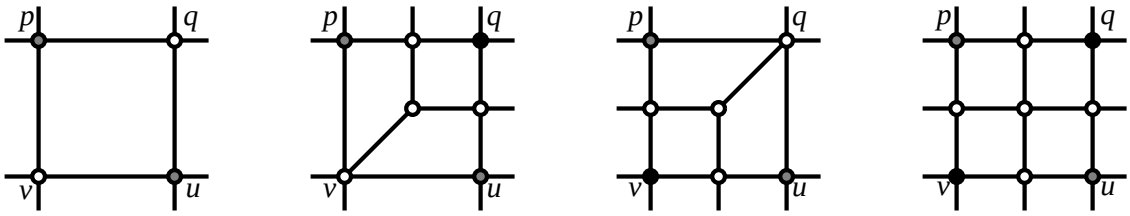

Figure 11: The orientation of the Y-elements depends on whether the vertices $\mathbf{q}$ and $\mathbf{v}$ are marked (black) or not (white). The status of vertices $\mathbf{p}$ and $\mathbf{u}$ does not matter (gray).

The correctness of this algorithm is obvious since the vertices which are marked in the first phase are those which are common to faces of different levels. The second phase guarantees that a corona of Y-elements is built around each such vertex (cf. Fig. 10).

7 Implementation and Examples

The described algorithm is designed to be useful in practical applications. Therefore, besides the features for creating creases and cusps and the ability to adaptively refine a given quadrilateral net, efficiency and compact implementation are also important. Both can be achieved by this algorithm. The crucial point of the implementation is the design of an appropriate data structure which supports an efficient navigation through the neighborhood of the vertices. The most frequently needed access operation to the data structure representing the balanced net, is to enumerate all faces which lie around one vertex or to enumerate all the neighbors of one vertex. Thus every vertex should be associated with a linked list of the objects that constitute its vicinity. We propose to do this implicitely by storing the topological information in a data structure `Face4Typ` which contains all the information of one quadrilateral face, i.e., references to its four corner points and references to its four directly neighboring faces. By these references, a doubly linked list around every vertex is available.

Since we have to maintain an adaptively refined net, we need an additional datatype to consistently store connections between faces from different refinement levels. We define another structure `Face9Typ` which holds references to nine vertices and eight neighbors. These *multi-faces* can be considered as 'almost' split faces, where the geometric information (the new edge- and face-points) is already computed but the topological split has not yet been performed. If, during adaptive refinement, some n-face is split then all its neighbors which are from the same generation are converted into `Face9Typ`'s. Since these faces have pointers to eight neighbors, they can mimic faces from different generations and therefore connect them correctly. The `Face9Typ`'s are the candidates for the placement of Y-elements in order to re-establish consistency. The various references between the different kinds of faces are shown in Fig. 12.

To relieve the application program which decides where to adaptively refine, from keeping track of the balance of the net, the implementation of the refinement algorithm should perform recursive refinement operations when necessary, i.e., if a n-face s is to be refined then first all $(n-1)$-neighbors which have at least one vertex in common with s must be split.

The following pictures are generated by using our experimental implementation. The criterion for adaptive refinement is a discrete approximation of the Gaussian curvature. The running time of the algorithm is directly proportional to the number of computed points, i.e., to the complexity of the output-net. Hence, since the number of regions where deep refinement is necessary usually is fixed, we can reduce the space- and time-complexity from exponential to linear (as a function of the highest occurring refinement level in the output).

Each of the appended colored pictures shows a given quadrilateral net N (gray) and a corresponding adaptively refined net N' (colored) where additionally the Gaussian curvature is printed (red: positive, blue: negative).

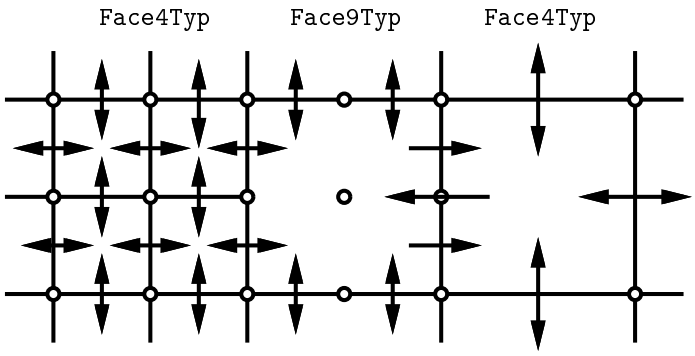

Figure 12: References between different kinds of faces.

Figure 13: Examples for adaptively refined nets.

References

[BS88] A. Ball / D. Storry, *Conditions for Tangent Plane Continuity over Recursively Generated B-Spline Surfaces*, ACM Trans. Graph. 7 (1988), pp. 83–102

[CC78] E. Catmull, J. Clark, *Recursively generated B-spline surfaces on arbitrary topological meshes*, CAD 10 (1978), pp. 350–355

[CDM91] A. Cavaretta / W. Dahmen / C. Micchelli, *Stationary Subdivision*, Memoirs of the AMS 93 (1991), pp. 1-186

[DS78] D. Doo / M. Sabin, *Behaviour of Recursive Division Surfaces Near Extraordinary Points*, CAD 10 (1978), pp. 356–360

[Dub86] S. Dubuc, *Interpolation Through an Iterative Scheme*, Jour. of Mathem. Anal. and Appl., 114 (1986), pp. 185–204

[DGL87] N. Dyn / J. Gregory / D. Levin, *A 4-Point Interpolatory Subdivision Scheme for Curve Design*, CAGD 4(1987), pp. 257–268

[DGL90] N. Dyn / J. Gregory / D. Levin, *A Butterfly Subdivision Scheme for Surface Interpolation with Tension Controll*, ACM Trans. Graph. 9 (1990), pp. 160–169

[DL90] N. Dyn / D. Levin, *Interpolating subdivision schemes for the generation of curves and surfaces*, Multivar. Approx. and Interp., W. Hauŝmann and K. Jetter (eds.), 1990 Birkhäuser Verlag, Basel

[Dyn91] N. Dyn, *Subdivision Schemes in Computer Aided Geometric Design*, Adv. in Num. Anal. II, Wavelets, Subdivisions and Radial Functions, W.A. Light ed., Oxford Univ. Press, 1991, pp. 36–104.

[DLL92] N. Dyn / D. Levin / D. Liu, *Interpolatory Convexity-Preserving Subdivision Schemes for Curves and Surfaces*, CAD 24 (1992), pp. 221–216

[HKD93] M. Halstead / M. Kass / T. DeRose, *Efficient, fair interpolation using Catmull-Clark surfaces*, Computer Graphics 27 (1993), pp. 35–44

[Hop94] H. Hoppe, *Surface Reconstruction from unorganized points*, Thesis, University of Washington, 1994

[Kob95] L. Kobbelt, *Using the Discrete Fourier-Transform to Analyse Interpolatory Subdivision Schemes*, submitted

[Loo87] C. Loop, *Smooth Subdivision Surfaces Based on Triangles*, Thesis, University of Utah, 1987

[Loo94] C. Loop, *A G^1 triangular spline surface of arbitrary topological type*, CAGD 11 (1994), pp. 303–330

[Pet90] J. Peters, *Smooth mesh interpolation with cubic patches*, CAD 22 (1990), pp. 109–120

[Pet95] J. Peters, *Smoothing polyhedra made easy*, ACM Trans. on Graph., Vol 14 (1995), pp. 161–169

[Rei95] U. Reif, *A unified approach to subdivision algorithms near extraordinary vertices*, CAGD 12 (1995), pp. 153–174

[Sch95] K. Schweizerhof, Universität Karlsruhe *private communication*