

A NEW CORNER CUTTING SCHEME WITH TENSION AND HANDLE-FACE RECONSTRUCTION

ERGUN AKLEMAN

*Department of Architecture, Texas A&M University
College Station, Texas 77843-3137, United States of America*

JIANER CHEN

*Department of Computer Science, Texas A&M University
College Station, Texas 77843-3112, United States of America*

VINOD SRINIVASAN

*Department of Architecture, Texas A&M University
College Station, Texas 77843-3137, United States of America*

FUSUN ERYOLDAS

*Department of Computer Science, Texas A&M University
College Station, Texas 77843-3112, United States of America*

A recently developed topological mesh modeling approach allows users to change topology of orientable 2-manifold meshes and to create unusual faces. Handle-faces are one of such faces that are commonly created during topology changes. This paper shows that vertex insertion and corner cutting subdivision schemes can effectively be used to reconstruct handle-faces. These reconstructions effectively show the structure of these unusual faces. The paper has three contributions. First, we develop a new corner cutting scheme, which provides a tension parameter to control the shape of subdivided surface. Second, we develop careful and efficient remeshing algorithms for our corner cutting scheme that use only the basic operations provided by our topological mesh modeling approach. This implementation ensures that our new corner cutting scheme preserves topological robustness. Finally, a comparative study shows that the corner cutting schemes create better handles and holes than the well-known Catmull-Clark scheme.

Keywords: Computer Aided Geometric Design, Shape Modeling, Computer Graphics, Visualization, Topology.

1. Introduction and Motivations

With the introduction of subdivision surfaces and the wider usage of solid modeling and implicit surfaces, topology has been becoming an important element of computer graphics research, development and production. There have been various studies in topological modeling during the last decade [15, 16, 19, 11, 21, 14, 12].

We have recently introduced topological graph theory [9, 13, 5] to computer graphics for effective and robust modeling of 2-manifold meshes [1]. We have also introduced a new data structure *Doubly Linked Face List* (DLFL), which guarantees

topological consistency and provides topological changes [1, 5]. We have identified two operations `INSERTEDGE` and `DELETEEDGE` to change mesh structure and manifold topology. Our recent investigation suggests that these two operations with two fundamental Euler operations, MVFS and KVFS [19], which we call `CREATEVERTEX` and `DELETEVERTEX` respectively, form a complete and minimal set of operations for robust and user-friendly modeling of any orientable 2-manifold mesh. In other words, these four operations constitute a necessary and sufficient minimal-set upon which all homeomorphic and topological operations can be implemented [6].

By using our approach, it is easy to change the topology of polygonal meshes. The topological changes we demonstrate include opening and closing holes, creating and deleting handles, connecting two disjoint meshes into one, and separating one mesh into two disjoint ones. During these operations, we also guarantee that polygonal meshes remain valid 2-manifolds.

Motivation for this paper comes from reconstruction of unusual faces that we encounter during topology changes. These unusual faces which we call handle-faces are created by our `INSERTEDGE` operation [1, 5] that can combine two distinct faces with a single edge. In most cases, the resulting handle-face cannot be effectively rendered with current graphics hardware. In order to see the actual structure of this handle-faces, we need to reconstruct them.

In most cases, these handle-faces are at least octagons and it is difficult to reconstruct them using parametric methods [10, 3, 18]. Instead, subdivision schemes [4, 8, 17, 20, 7] provide an attractive alternative for the reconstruction of such handle-faces. In fact, we have recently shown that Catmull-Clark subdivision algorithm [4] over the DLFL structure [2] can effectively be used to reconstruct these handle-faces. Reconstruction with Catmull-Clark subdivision greatly improves the quality of handle-faces. However, in the first iteration of Catmull-Clark, high valence extraordinary vertices are created. Limit surfaces around these vertices generally have ripples that give a G^1 discontinuous look.

In this work, we further study the handle-face reconstruction with subdivision algorithms. We first develop a new corner cutting algorithm and implement it with our minimal set of operations [6]. The new algorithm is similar to Doo-Sabin subdivision scheme, and has the property that users can control the shape of subdivided surface via a parameter. This parameter is independent of the valence of the faces (the valence of a face is the number of sides of the face).

We have also developed two algorithms to remesh a 2-manifold mesh based on the corner cutting scheme. These algorithms are entirely based on our minimal set of operations.

In one of the algorithms, the new mesh is constructed by adding and deleting edges to the old mesh, and the other algorithm creates the new mesh from scratch starting from a single vertex and keeps the old mesh completely intact. Since both algorithms are entirely based on our minimal set of operations, they guarantee the topological robustness of the implementation.

Finally, we compare our new subdivision algorithms with the well-known Catmull-

Clark algorithm which is also implemented on the DLFL structure, and show that the new corner cutting and Doo-Sabin algorithms give a better quality handle-face reconstruction.

2. Topological Operations and Handle-Faces

We have recently identified a minimal set of basic operations on the DLFL structure that are necessary and sufficient for performing all homeomorphic and topological operations on orientable 2-manifolds [6]. A clear identification of the minimal set of operations distinguishes the lower level fundamental operations from the higher level operations, and greatly improves computational efficiency, user interactivity, and topological robustness.

The minimal set consists of four fundamental operations, as follows:

1. `CREATEVERTEX(v)` creates a 2-manifold surface with one vertex v and one face f which we call a *point sphere* as shown in Figure 1. This operation is the same as the Euler operation *MVFS* [19] and effectively adds a new surface component to the current 2-manifold.

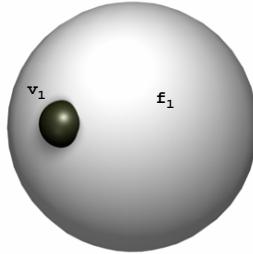


Figure 1: A point-sphere that consists of only one vertex and one face.

2. `DELETEVERTEX(v)` is the inverse of the `CREATEVERTEX(v)` operation, i.e., it deletes a point sphere from the current 2-manifold. This operation is the same as the Euler operation *KVFS* [19].
3. `INSERTEDGE(c_1, c_2, e)` inserts a new edge e to the mesh structure between two corners c_1 and c_2 (a *corner* is a subsequence of a face boundary walk consisting of two consecutive edges plus the vertex between them. A corner can also be given as a face-vertex pair). According to our current implementation of the DLFL structure, the operation `INSERTEDGE` takes time logarithmic in the number of vertices in the involved faces.
4. `DELETEEDGE(e)` deletes an edge e from the current 2-manifold mesh represented by the DLFL structure. This is the inverse operation of `INSERTEDGE`.

and also takes time logarithmic in the number of vertices in the involved faces in our current implementation of the DLFL structure.

The above set of operations is minimal and complete in the sense that it is necessary and sufficient for performing all kinds of homeomorphic and topological operations on orientable 2-manifold meshes. The CREATEVERTEX operation is essential in the initial stage of manipulating 2-manifolds and creates a new surface component in the given 2-manifold. In particular, it is necessary when a new surface component is created in an empty manifold. Similarly, the DELETEVERTEX operation is convenient for “cleaning up” a manifold, and is a necessary operation when a surface component needs to be completely removed from the 2-manifold.

The INSERTEDGE is the only operation among the four basic operations that can increase the genus of a manifold. In general, if INSERTEDGE inserts an edge between two corners of the same face, the new edge divides the face into two faces without changing topology, as shown in Figure 2. On the other hand, if INSERTEDGE inserts an edge between corners of two different faces (this includes the situation in which an endpoint or both endpoints of the new edge correspond to point-spheres), the new edge merges the two faces into one and changes the topology of the 2-manifold. This situation is shown in Figure 3. The unusual face shown at the left in this figure is an example of the type of faces which we call “*handle-faces*”.

In case the operation INSERTEDGE is inserting an edge e to corners of two different faces f_1 and f_2 , there is an interesting and intuitive interpretation of the operation as follows: the operation can be decomposed into two steps. In the first step, we cut off along the boundaries of the faces f_1 and f_2 , which results in a 2-manifold with two “open” holes, then the second step runs a new “pipe” between the two holes and allows the pipe ends to “seal” the two holes. An illustration is shown in Figure 4.

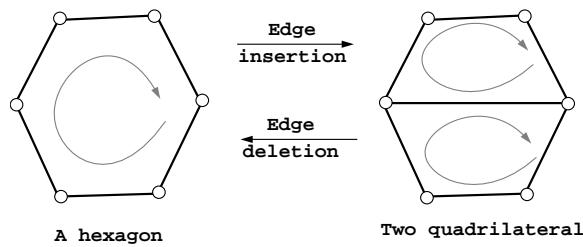


Figure 2: Inserting an edge between two vertices of the same face divides the face into two and deleting an edge between two faces merges the two faces into one.

DELETEEDGE is the inverse operation of the INSERTEDGE. As shown in Figure 2 and Figure 3, DELETEEDGE either merges two faces into a bigger face or splits a handle-face (which forms a pipe) into two faces. When DELETEEDGE splits a handle-face into two faces (as shown in Figure 3), the situation can be described by a 2-step process in which the first step cuts off a pipe from the 2-manifold and

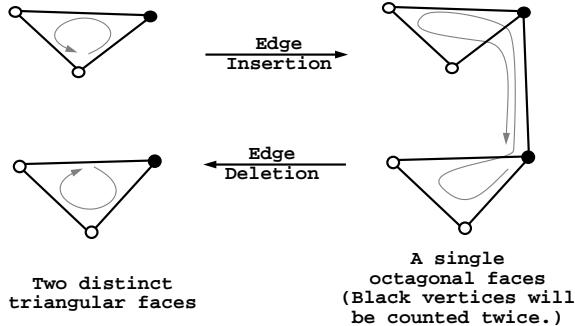


Figure 3: Inserting an edge between two different faces merges the two faces and deleting an edge with both sides in the same face splits the face into two.

leaves two open holes and, the second step seals the two open holes by two disks (which are the two new faces). This operation eliminates either a handle or a hole from the 2-manifold, which is the only, thus also necessary operation among the four that decreases the genus of a 2-manifold.

One important challenge is to efficiently visualize the handle-faces – they are a bit strange looking and current graphics hardware does not support such strange faces. We notice that subdivision schemes can effectively be used to reconstruct such faces and make the structure of the handle become apparent as illustrated in the last figure in Figure 5.

3. Handle-Face Reconstruction with Subdivision

The existing subdivision schemes can be classified based on three criteria [22, 23]: (1) the type of refinement rule (e.g. vertex insertion [4] or corner cutting [8]) (2) the type of generated mesh (e.g. triangular [17] or quadrilateral [4]) (3) whether the scheme is approximating or interpolating. We observed that, for the improvement of handle-faces, triangular or interpolating schemes are not appropriate choices. This observation limited our choices to approximating and quadrilateral schemes. Earlier we have implemented Catmull-Clark subdivision algorithm [4] on the DLFL structure [2] (Catmull-Clark is a vertex insertion, approximating and quadrilateral scheme [22, 23]). Based on this implementation, we have demonstrated how the DLFL structure supports efficiently and effectively topological change and subdivision operations. In particular, we showed how a handle can be added to a cup based on the system (see [2] for details).

One feature that is not completely satisfactory is that the reconstruction of the handle-face based on a vertex insertion subdivision scheme creates an extraordinary vertex with a high valence. (The “valence” of a vertex is the number of edges incident on the vertex.) This extraordinary vertex occurs at the middle of the handle as shown in Figure 5 in the first iteration of vertex insertion scheme. The valence

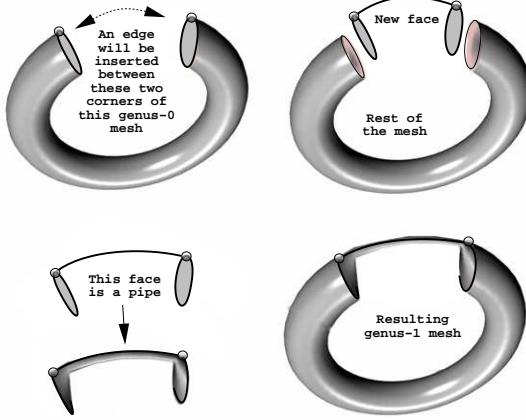


Figure 4: A multiple-step representative illustration of creating a handle by inserting an edge. (Note that cutting off a handle can be considered the reverse order of this process).

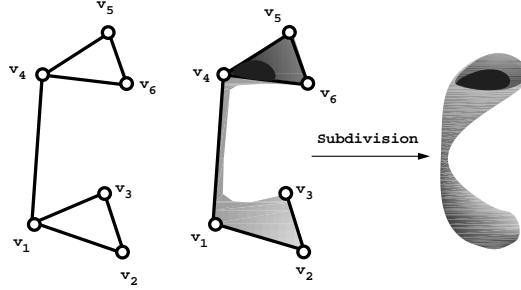


Figure 5: Handle becomes apparent after subdivision operations.

of the extraordinary vertex on a handle is generally higher than 8. The resulting surface does not always look smooth around such an extraordinary vertex because of the ripples created as a result of the unusual structure around the extraordinary vertex as shown in Figure 6*.

In this paper, we show that the quality of the reconstruction of handle-faces can be improved by using corner cutting schemes. Corner cutting schemes do not create additional extraordinary vertices. Under a corner cutting scheme, the handle-face eventually becomes a high-valence planar face, which is a part of C^1 smooth handle [8]. After a few number of iterations, this high-valence face does not create any

*The reconstruction with subdivision is one of the ways to improve the quality of handle-faces. Inserting new edges to a handle-face can also help to remove the ripples and avoid high valence vertices. If a second edge is inserted to connect any two vertices of a handle-face, the new edge separates the handle-face into two faces without changing the topology of the mesh. By inserting more edges, the quality of the handle can be improved further [2].

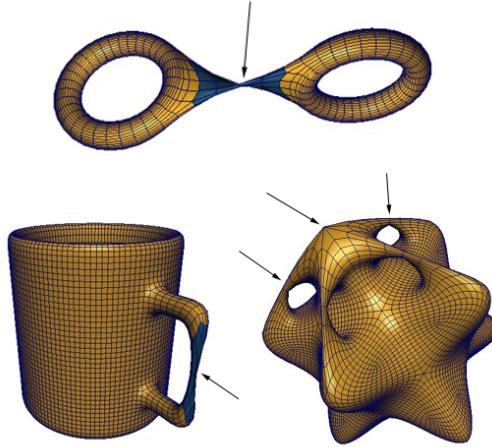


Figure 6: Examples of surfaces that do not look smooth near extraordinary vertices.

visual distraction since it becomes almost planar.

4. Corner Cutting Schemes

The refinement rules for a general corner cutting algorithm are as follows:

- **Step 1 :** For each vertex v_n of a face $f = \{v_0, v_1, \dots, v_n, \dots, v_{N-1}\}$ of the mesh, create a new vertex v'_n . Compute the position of the new vertex as

$$v'_n = \sum_{m=0}^{N-1} a_{n,m} v_m \quad (1)$$

where $a_{n,m}$ are real coefficients described by the corner cutting scheme and N is the valence of the face.

- **Step 2 :** For each old face, create a new face by connecting all the new points that have been generated in that face;
- **Step 3 :** For each old vertex, connect all the new points that have been generated for that vertex (in all faces whose boundary contains the vertex);
- **Step 4 :** For each old edge, connect the four new points that have been generated for the two ends of the edge.
- **Step 5 :** Remove all old vertices and edges.

For any subdivision scheme using approximation, the coefficients $a_{n,m}$ in step 1 in the above algorithm must satisfy the following conditions (we assume that the vertex indices are given in the order of a face traversing):

1. $a_{n,m} \geq 0$ for all n and all m and

$$2. \text{ for all } n\text{'s } \sum_{m=0}^{N-1} a_{n,m} = 1.$$

These conditions guarantee convergence of the algorithm and provide C^0 continuity and affine invariance properties [22, 23].

4.1. Doo-Sabin Algorithm

For example, in the well-known Doo-Sabin subdivision scheme, the coefficients $a_{n,m}$ in equation (1) are computed using the following formulas:

$$\begin{aligned} a_{n,m} &= \frac{1}{4} + \frac{5}{4N} && \text{if } n = m \\ a_{n,m} &= \frac{(3+2\cos(\frac{2(n-m)\pi}{N}))}{4N} && \text{otherwise} \end{aligned}$$

It is easy to verify that in Doo-Sabin scheme, each coefficient is always greater than zero and the coefficients add up to 1. By using Fourier analysis, Doo-Sabin showed that this scheme has three distinct eigenvalues 1, 1/2, 1/4. One eigenvector has the eigenvalue 1, two eigenvectors correspond to eigenvalue 1/2 and the rest of the eigenvectors correspond to the smallest eigenvalue 1/4. This structure of eigenvalues of the Doo-Sabin scheme guarantees that the scheme provides tangent plane continuity [8, 22, 23].

4.2. A New Corner Cutting Algorithm with Tension

Our new corner cutting scheme is inspired by Doo-Sabin algorithm. Let vertex indices in a face be given by a rotation order, we use the following formula to compute the coefficients $a_{n,m}$ to compute the position of v'_n .

$$\begin{aligned} a_{n,m} &= a && \text{if } n = m \\ a_{n,m} &= M \frac{1-a}{3N-5} && \text{otherwise} \end{aligned} \tag{2}$$

where

$$M = 3 + 2 \cos \left(\frac{2(n-m)\pi}{N} \right).$$

In these equations, parameter a in equation (2) is provided by users and it is used as a tension parameter [3]. The new algorithm gives subdivision rules for generating quadratic B-spline tensor-product surfaces for $N = 4$ and $a = 9/16$. Note that, in our case the coefficients add up to 1 and if $0 < a < 1$ each coefficient is greater than zero.

By using Fourier analysis, similar to Doo-Sabin's approach, we can show that the new scheme also has three distinct eigenvalues

$$1, \frac{(2a+1)N-5}{3N-5}, \frac{3aN-5}{3N-5}$$

and if $a < 1$ then regardless of the value of N the sequence is strictly decreasing. Similar to Doo-Sabin scheme, only one eigenvector corresponds to eigenvalue 1, two eigenvectors correspond to the second largest eigenvalue and the rest of the eigenvectors correspond to the smallest eigenvalue. This result demonstrates that our scheme also provides tangent plane continuity.

There also exists a lower limit over the value of a . To have positive eigenvalues, for a triangular face. (i.e., $N = 3$) a must be larger than $1/3$ (Note that in triangle case, the third eigenvalue will not exist.) For a quadrilateral face (i.e., $N = 4$) a must be larger than $5/12$. Otherwise, the smallest eigenvalue can become negative. It is safer to choose $5/12$ as a lower bound for a since in corner cutting schemes we cannot avoid quadrilaterals (each old edge become a quadrilateral after the application of corner cutting scheme once.)

Figures 7 and 8 show the first four consecutive iterations of the new corner cutting scheme for a cube control shape. In these examples, we used the parameters $a = 9/16$ and $a = 0.9$. As shown in the Figures, the larger a values, the less rounded shape becomes. This is because for larger a values, the position v'_n will geometrically be closer to the position of v_n . By using this property, the user can control the degree of smoothness.

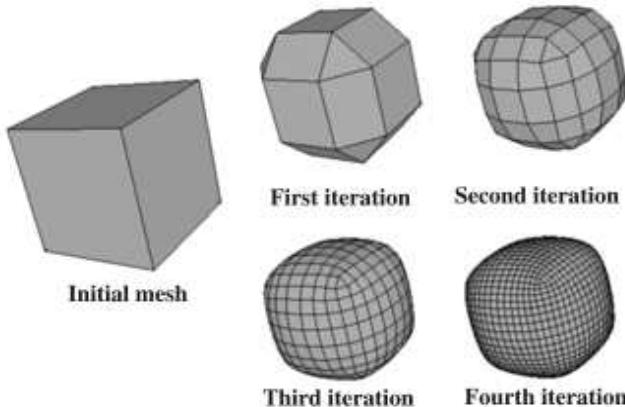


Figure 7: The first four consecutive iterations with $a = 9/16$.

5. Remeshing Algorithms for Corner Cutting

It is straightforward to implement vertex insertion schemes (such as Catmull-Clark) over the DLFL structure by using only minimal set of operations [2] which guarantee topological robustness during the process. Unfortunately, the same cannot be directly done for the development of algorithms to implement corner cutting schemes, since the new mesh structure resulting from corner cutting scheme is completely disjoint from the initial mesh. More seriously, if the new mesh is constructed

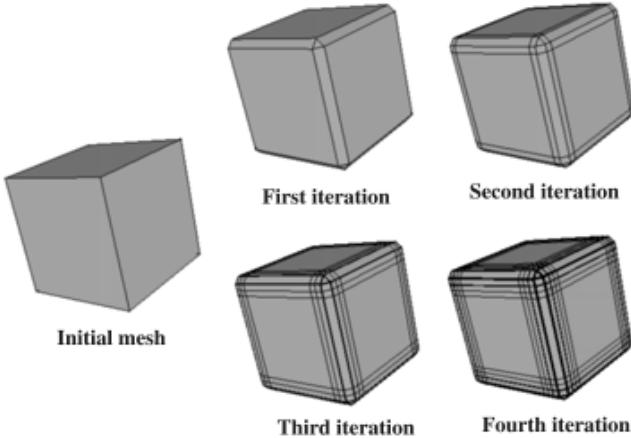


Figure 8: The first four consecutive iterations with $a = 0.9$.

by operations that are not provided by DLFL structure, topological inconsistency might be introduced during the process. In the following, we present two remeshing algorithms for corner cutting schemes that only use the operations provided by DLFL structure. Since DLFL structure and these operations are topologically robust, our algorithm always guarantees topological consistency.

Let \mathcal{M} be a mesh given in the DLFL structure such that we will apply a corner cutting algorithm on \mathcal{M} . If \mathcal{M} has n vertices, m edges and f faces, then the new mesh structure \mathcal{M}' after corner cutting algorithm will have $n + m + f$ faces: each face in \mathcal{M} induces a new face of the same valence in \mathcal{M}' (see Step 2 of the general description of the corner cutting scheme); each edge in \mathcal{M} induces a new face of valence 4 in \mathcal{M}' (see Step 4 of the description); and each vertex in \mathcal{M} induces a new face of valence r in \mathcal{M}' , where r is the valence of the vertex (see Step 3 of the description).

5.1. Remeshing Algorithm - 1

For the development of the first remeshing algorithm we use one high-level operation to simplify the steps of the algorithm:

$\text{SUBDIV}(e, v)$ subdivides the edge $e = (u, w)$ by a new vertex v of valence 2 so that the edge e becomes two new edges (u, v) and (v, w) . This operation can be implemented with the minimal set of operations (first delete the edge (u, w) , then create an isolated vertex v , then insert edges between u and v , and between v and w). In our work, since it is an extremely simple operation, we have implemented it as a stand alone operation.

The first remeshing algorithm proceeds as follows for a given mesh \mathcal{M} under the DLFL structure,

- **Step 1:** for each face $f = \{v_0, v_1, \dots, v_{N-1}\}$ in the mesh \mathcal{M}
 - 1.1. for $i = 0$ to $N - 1$ do
 - subdivide edge $[v_n, v_{(n+1) \pmod N}]$ by a new vertex v'_n ;
 - 1.2. compute the positions of the new vertices $v'_0, v'_1, \dots, v'_{N-1}$ using the formulas (1), and (2);
 - 1.3. for $n = 0$ to $N - 1$ do
 - insert the new edges $[v'_n, v'_{(n+1) \pmod N}]$ into the face;

Remark 1. Traversing the faces of the mesh \mathcal{M} can be simply done by calling the operation FACETRAV provided by the DLFL structure, whose running time is linear in the valence of the face being traversed.

Remark 2. Each edge $[v, w]$ in \mathcal{M} is traversed exactly twice in Step 1, once from v to w and once from w to v . After the first traverse, the edge $[v, w]$ of \mathcal{M} is subdivided into $[v, v', w]$. In consequence, when the edge is traversed in the second time, the new vertex v' is also encountered. Therefore, in the above face traversing, the new added vertices should be simply ignored when faces in \mathcal{M} are traversed. In particular, after Step 1, each edge $[v, w]$ in the mesh \mathcal{M} is subdivided by two new vertices v' and w' of valence 2 and the edge $[v, w]$ becomes three consecutive edges $[v, v']$, $[v', w']$, and $[w', w]$ (see Figure 9B).

Remark 3. It will be helpful to compare the topological view and geometric view for Step 1. Topologically, each edge $[v, w]$ in \mathcal{M} is subdivided by two new vertices v' and w' of valence 2 (See Figure 9B). On the other hand, from geometric point of view, since the locations of the new vertices v' and w' are recomputed, the segment sequence $[v, v']$, $[v', w']$, $[w', w]$ is no longer on a single straight line but becomes zigzag (see Figure 9C).

Remark 4. In Step 1.3, since the new edges $[v'_n, v'_{(n+1) \pmod N}]$ are all inserted in the given face F , the face corners for each new inserted edge are uniquely determined. Also note that after Step 1, each new vertex becomes of valence 3. (see Figure 9D).

- **Step 2 :** Now for each old vertex v in the mesh \mathcal{M} , suppose the neighbors of v , which are all new added vertices, are ordered by the rotation at v as $\bar{v}_0, \bar{v}_1, \dots, \bar{v}_{r-1}$. Insert new edges $[\bar{v}_j, \bar{v}_{(j+1) \pmod r}]$ for $j = 0, 1, \dots, r - 1$ (see Figure 9E).

Remark 5. The DLFL structure also provides a VERTEXTRAV operation that allows to traverse the neighbors of a vertex in the order of the rotation at the vertex in time linear in the valence of the vertex. Therefore, Step 2 can be done very efficiently. Moreover, the face corners for the new inserted edge $[\bar{v}_j, \bar{v}_{(j+1) \pmod r}]$ are in the face that also contains the face corner $(\bar{v}_{(j+1) \pmod r}, v, \bar{v}_j)$, so they can be found conveniently.

- **Step 3 :** Finally, we delete all edges of form $[v', w']$, where v' and w' are the new vertices subdividing an old edge $[v, w]$ in the mesh \mathcal{M} , and delete all old vertices in \mathcal{M} and the edges incident to them (by repeatedly using the edge delete operation followed by a single vertex delete operation).

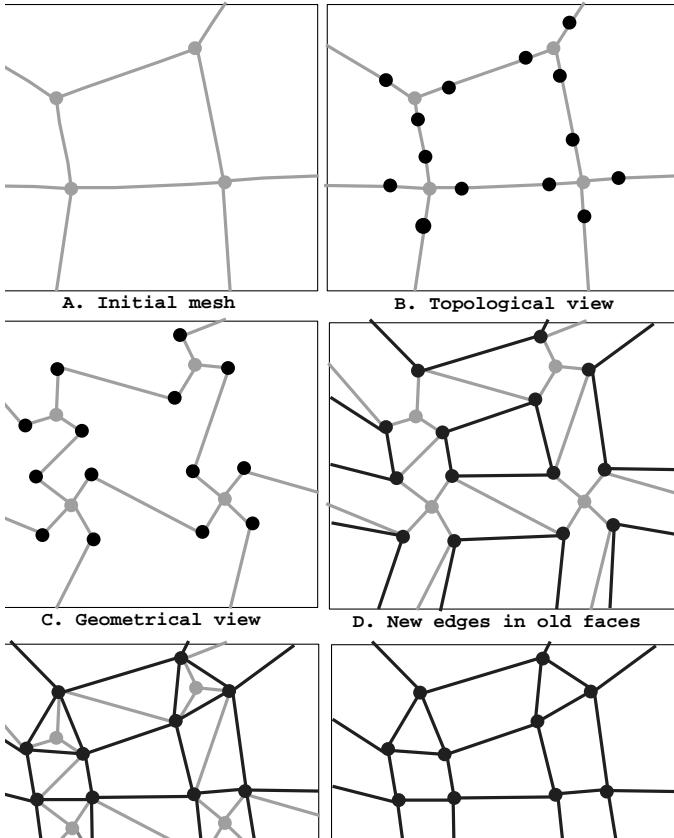


Figure 9: Illustration of first remeshing algorithm.

Remark 6. A careful examination shows that the new mesh structure \mathcal{M}' shares neither common vertex nor common edge with the old mesh \mathcal{M} . Even the new edges resulted from subdividing the old edges of \mathcal{M} are eventually also removed. Therefore, for implementation convenience, we can simply create a separate new Vertex list and new Edge list for the new vertices and edges, respectively, instead of modifying the old Vertex list and old Edge list. The old Vertex list and Edge list will be entirely deleted at the end of the algorithm.

It is not difficult to verify that none of the operations we applied in this algorithm changes the topology of the initial mesh \mathcal{M} . Therefore, the resulting structure \mathcal{M}' must be a corner cutting remesh of the original manifold structure.

5.2. Remeshing Algorithm - 2

One of the problem with the previous remeshing algorithm is that when the algorithm is completed the initial mesh \mathcal{M} is completely lost. In order to allow operations such as undo, it is better to preserve the initial mesh \mathcal{M} . The following second remeshing algorithm in this section preserve the initial mesh \mathcal{M} and it is also based on our topology operations.

For the development of the second remeshing algorithm we have developed a high-level operation that simplifies the steps of the algorithm. This high level operation given below has been implemented efficiently with the minimal set of operations and thus guarantees topological consistency.

`CREATEFACEMANIFOLD(v_0, v_1, \dots, v_{N-1})`. This operation creates a two sided face (a manifold surface) by the following procedure.

1. for $i = 0$ to $N - 1$ do
 `CREATEVERTEX(v_i);`
2. for $i = 0$ to $N - 1$ do
 `INSERTEDGE($v_i, v_{(i+1) \text{ mod } N}$, e).`

Remark 1. Each of these vertices before the insert edge operation has a valence less than 2. Therefore, we do not have to specify the corners.

The algorithm proceeds as follows for a given mesh \mathcal{M} under the DLFL structure (see Figure 10):

- **Step 1:** for each face $F = \{v_0, v_1, \dots, v_{N-1}\}$ in the mesh \mathcal{M}
 - 1.1. compute the positions of the new vertices $v'_0, v'_1, \dots, v'_{N-1}$ using the formulas (1), and (2);
 - 1.2. `CREATEFACEMANIFOLD($v'_0, v'_1, \dots, v'_{N-1}$)`.

Remark 2. Step 1 creates two new faces $f' = (v'_0, v'_1, \dots, v'_{N-1})$ and $f'' = (v'_{N-1}, v'_{N-2}, \dots, v'_1, v'_0)$ for each old face $f = (v_0, v_1, \dots, v_{N-1})$ (see Figure 10.B). We call f' and f'' “the front face” and “the back face” respectively. Note that each new vertex v'_i the adjacent faces are just the front face and back face. Thus, we can talk about the “front corner” and “back corner” at vertex v'_i . More specifically, the front corner of v'_i is the vertex-face pair $\{v'_i, f'\}$, while back corner at v'_i is the vertex-face pair $\{v'_i, f''\}$.

- **Step 2:** for each old edge $e = \{u, w\}$ the two ends of e induce two pairs of new vertices $\{v', v''\}$ and $\{w', w''\}$. Insert an edge between the back corners of the vertices v' and v'' , and insert another edge between the back corners of the vertices w' and w'' (see Figure 10.C).

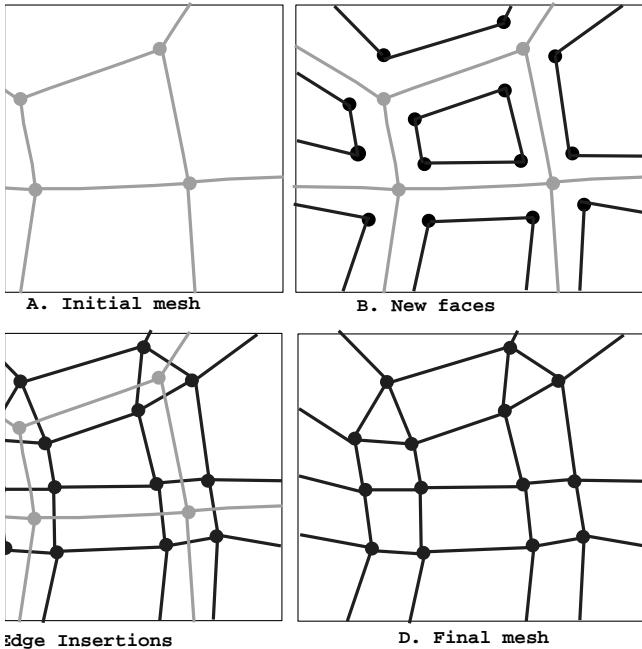


Figure 10: Illustration of second remeshing algorithm.

Let \mathcal{M}'' denote the new mesh structure created by remeshing algorithm 2. Let us compare \mathcal{M}'' with the mesh structure \mathcal{M}' created by the remeshing algorithm 1 (see Figure 9). It is not difficult to verify that they have the same vertex position and the same edge set (because the vertex positions are computed using the same formulas (1) and (2)). Moreover the ‘‘rotation’’ at each vertex (i.e. the cyclic order of the edges incident to the vertex) is also identical in the two mesh structures. According to the principle of rotation systems (see [1, 5] for further explanation) the mesh structures \mathcal{M}' and \mathcal{M}'' have exactly the same topological structure. In consequence, they give exactly the same mesh structure (both topologically and geometrically).

6. Implementation and Results

Both algorithms have been implemented in a C++ program. We applied the new corner cutting algorithm to the control shapes we used for Catmull-Clark subdivision to obtain smooth handles and holes [2]. It is interesting to note that the corner cutting algorithm creates better looking handles and holes than Catmull-Clark algorithm as shown in Figure 11 and Figure 12. Note that in these examples, unlike in the Catmull-Clark case, handles and holes look smoother even for only one edge insertion. Inserting additional edges further improves the quality of the handles and holes.

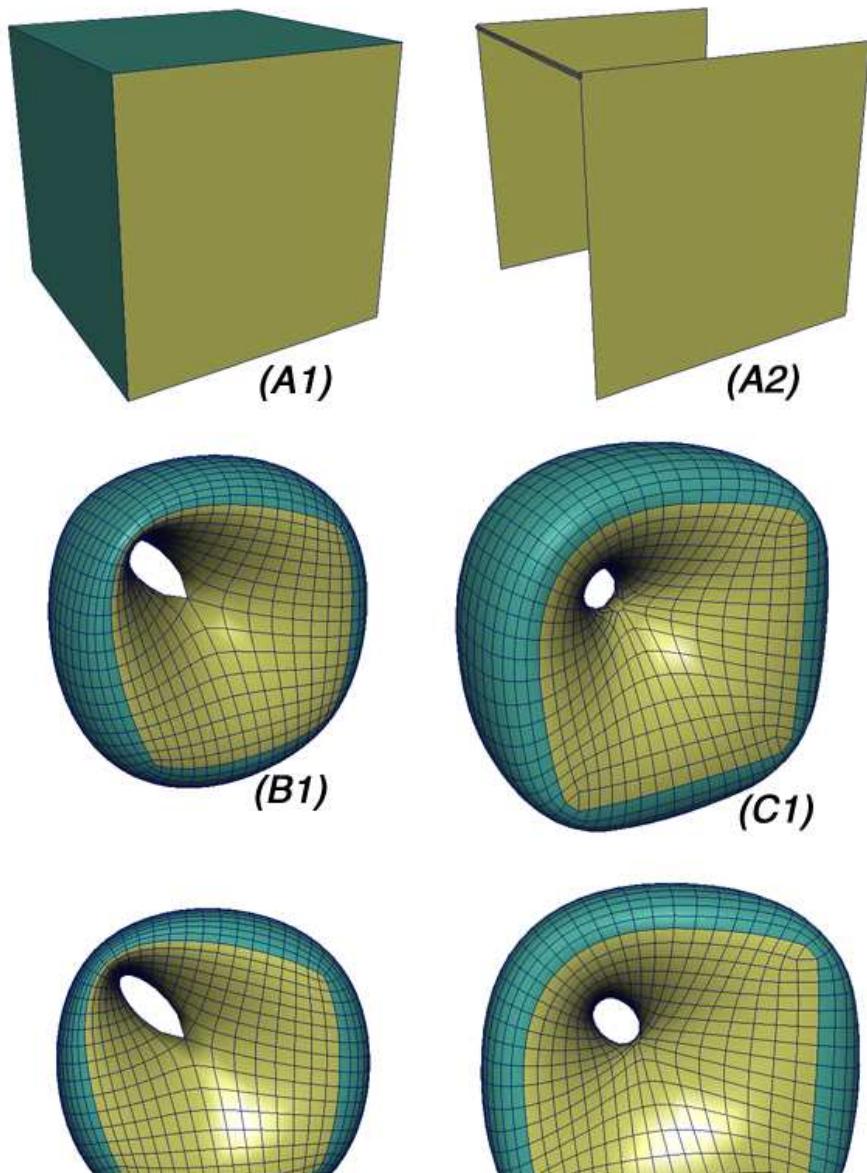


Figure 11: Comparison of handle reconstruction with vertex insertion and corner cutting. (A1) is the control mesh and (A2) shows the handle-face in (A1). (B1) and (B2) are the result of Catmull Clark and give two different views of the same objects. (C1) and (C2) are the result of corner cutting scheme with $a = 9/16$ and give two different views of the same objects.

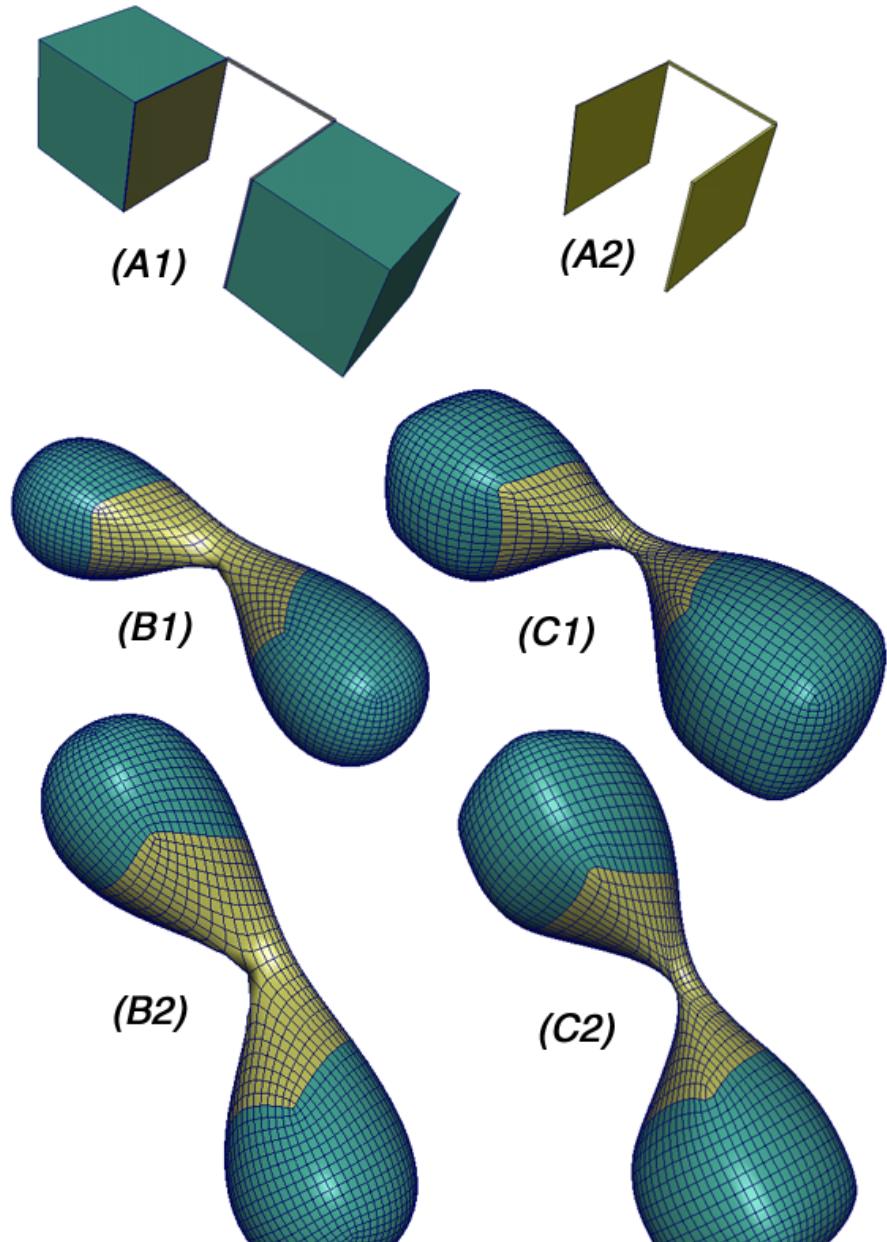


Figure 12: Comparison of handle reconstruction with vertex insertion and corner cutting. (A1) is the control mesh and (A2) shows the handle-face in (A1). (B1) and (B2) are the result of Catmull Clark and give two different views of the same objects. (C1) and (C2) are the result of corner cutting scheme with $a = 9/16$ and give two different views of the same objects.

7. Discussion, Conclusion and Future Work

In this paper, we continue to study the relationship between the topological mesh modeling and subdivision algorithms. We have developed a new corner cutting algorithm, which provides a tension parameter to control the shape of subdivided surface. We have also implemented the new corner cutting algorithm, over the DLFL structure.

We have observed that the corner cutting schemes are more appropriate for the reconstruction of handle-faces as they create better handles and holes than Catmull-Clark. It can also be informative to compare reconstruction of other types of unusual faces that are created during topological modeling. Our initial investigation strongly suggest that for other types of unusual faces, limit surfaces created by the Doo-Sabin scheme also behave better than those created by the Catmull-Clark scheme. It can be interesting to develop vertex insertion schemes that can effectively reconstruct such unusual faces.

This paper, along with [2], shows that the DLFL structure is subdivision friendly. Since subdivision operations are essential for the improvement of the quality of meshes, performing the topological change operations and the subdivision operations alternatively provides a powerful shape modeling approach that supports a hierarchy of topology changes and quality improvements at different levels of details.

8. Acknowledgments

We are grateful to the reviewers of Shape Modeling International 2001 conference who provided valuable ideas for the improvements of the paper. This work is partially funded by the Research Council of College of Architecture, Interdisciplinary Program of the Texas A&M University and NSF under grant CCR-0000206.

1. E. Akleman and J. Chen, "Guaranteeing the 2-Manifold Property for meshes with Doubly Linked Face List", *International Journal of Shape Modeling* Volume 5, No 2, (2000) 149-177.
2. E. Akleman, J. Chen, and V. Srinivasan, "A New Paradigm for Changing Topology During Subdivision Modeling," *Pacific Graphics 2000*, (October 2000) 192-201.
3. R. H. Bartels, J. C. Beatty, and B. A. Barsky, *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann Publishers, Los Altos, CA, 1987.
4. E. Catmull and J. Clark, "Recursively Generated B-spline Surfaces on Arbitrary Topological Meshes", *Computer Aided Design*, **10** (September 1978) 350-355.
5. J. Chen, "Algorithmic Graph Embeddings", *Theoretical Computer Science*, **181** (1997) 247-266.
6. J. Chen and E. Akleman, "Topologically Robust Mesh Modeling: Concepts, Data Structures and Operations", *In Preparation*.
7. T. deRose, M. Kass and T. Truong, "Subdivision Surfaces in Character Animation", *Computer Graphics*, **32** (August 1998) 85-94.
8. D. Doo and M. Sabin, "Behavior of Recursive Subdivision Surfaces Near Extraordinary Points", *Computer Aided Design*, **10** (September 1978) 356-360.
9. J. Edmonds, "A Combinatorial Representation for Polyhedral Surfaces", *Notices American Mathematics Society*, **7** (1960) 646.

10. G. Farin, *Curves and Surfaces for Computer Aided geometric Design, A Practical Guide*, Academic Press, Inc, London, (1988).
11. H. Ferguson, A. Rockwood and J. Cox, "Topological Design of Sculptured Surfaces", *Computer Graphics*, **26** (August 1992) 149-156.
12. A. T. Fomenko and T. L. Kunii, *Topological Modeling for Visualization*, Springer-Verlag, New York, (1997).
13. J. L. Gross and T. W. Tucker, *Topological Graph Theory*, Wiley Interscience, New York, (1987).
14. B. T. Stander and J. C. Hart, "Guaranteeing the Topology of an Implicit Surface Polygonization for Interactive Modeling", *Computer Graphics*, **31** (August 1997) 279-286.
15. C. M. Hoffmann, *Geometric & Solid Modeling, An Introduction*, Morgan Kaufman Publishers, Inc., San Mateo, Ca., (1989).
16. C. M. Hoffmann and G. Vanecek, "Fundamental techniques for geometric and solid modeling", *Manufacturing and Automation Systems: Techniques and Technologies*, **48** (1990) 347-356.
17. C. Loop, "Smooth Subdivision Surfaces Based on Triangles", Master's Thesis, Department of Mathematics, University of Utah (1987).
18. C. Loop and T. DeRose, "Generalized B-spline Surfaces with Arbitrary Topology", *Computer Graphics*, **24** (August 1991) 101-165.
19. M. Mantyla, *An Introduction to Solid Modeling*, Computer Science Press, Rockville, Ma., (1988).
20. T. W. Sederberg, D. Sewell, and M. Sabin, "Non-Uniform recursive Subdivision Surfaces", *Computer Graphics*, **32** (August 1998) 387-394.
21. S. Takahashi, Y. Shinagawa and T. L. Kunii, "A Feature-Based Approach for Smooth Surfaces", in *Proceedings of Fourth Symposium on Solid Modeling*, (1997) 97-110.
22. D. Zorin and P. Schröder, co-editors, *Subdivision for Modeling and Animation*, ACM SIGGRAPH'99 Course Notes no. 37, August, 1999.
23. D. Zorin and P. Schröder, co-editors, *Subdivision for Modeling and Animation*, ACM SIGGRAPH'2000 Course Notes no. 23, July, 2000.