A BUTTERFLY ADAPTIVE SUBDIVISION ALGORITHM

SHUJIN LIN*, JIANQI HE, YU GUO, BIN KONG

Computer Application Institute, Sun Yat-sen University.
GuangZhou, 510275,China
*linyu210@163.net**

Abstract

We present an algorithm implementing adaptive subdivision for triangular mesh that makes use of the Modified Butterfly scheme and realizes various subdivision levels by 1-to-4split and 1-to-2split. It computes the new butterfly split vertex by the mask method and constructing virtual split Binary-Quad Tree. The Data Structure bases on the Binary-Quad Tree constructed by nodes of triangle faces. As advantages of the algorithm, it simplifies computation and decreases time cost greatly with a few space redundant. The 1-to-4split and 1-to-2split adaptive subdivision prescripts and rules can avoid producing thiny triangles and control the increased number of extraordinary vertex. In addition, the algorithm is also applicable to many other subdivision schemes for triangular meshes, such as loop scheme, $\sqrt{3}$ scheme and so on. We focus on Modified Butterfly subdivision only for the purpose of perspicuity.

Keywords: Adaptive Subdivision; Modified Butterfly Scheme; Binary-Quad Tree; Face Crack Problem; Even Vertex.

1. Introduction

Usually, subdivision curves and surfaces are visualized by drawing a polygonal chain or mesh on a level of refinement that evokes the impression of sufficient approximation. A disadvantage here is that the mesh is subdivided uniformly everywhere. The required level of subdivision is determined by those locations of the mesh that approximate the limit surface most unfavorably. That may cause unnecessary fine subdivisions at other locations of the surface. Because the combinatorial size of the mesh grows exponentially in the number of levels, this effect is quite undesirable. For the reason, Adaptive Subdivision aims at providing a local subdivision rule that governs whether or not a given face in a mesh needs to be subdivided at the next level of subdivision.

2. Related Work

Amresh[1] have developed an adaptive subdivision scheme based on the Loop scheme using the method of watershed segmentation. Dirc Rose[2] have proposed an adaptive process that stores the next vertex to split and the temporary triangle in advance for Modified Butterfly scheme. Kobbelt[3] have developed adaptive refinement for both his Kobbelt scheme and newly introduced $\sqrt{3}$ subdivision. For his $\sqrt{3}$ subdivision he uses a combination of dyadic refinement, mesh balancing and gap fixing by temporary triangle fans. S. Seeger[4] have developed an adaptive subdivision scheme based on Butterfly Scheme using quark. Xu and Kondo[5] have devised an adaptive subdivision scheme based on the Doo-Sabin scheme. In their method the adaptive subdivision is controlled by the faces of the original mesh.

In our adaptive subdivision, we make use of the Modified Butterfly scheme and realize various subdivision levels by1-to-4split together with 1-to-2split. We compute the new split vertexes by the mask method and construct virtual split Binary-Quad Tree. The Data Structure bases on a Binary-Quad Tree which nodes are triangle faces.

The Modified Butterfly scheme used in our algorithm is an interpolating subdivision surface, which is introduced by Zorin[7]. It subdivides a triangle mesh by 1-to-4 split the faces as shown in figure1 and can ensure overall C^1-continuity. Our algorithm is also applicable to many other subdivision schemes for

triangular meshes, such as loop scheme, $\sqrt{3}$ scheme and so on. We focus on Modified Butterfly subdivision only for the purpose of perspicuity.

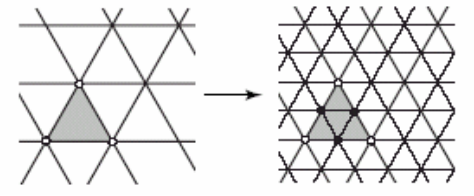

Fig. 1. Subdivision a triangle mesh by face split

To give a demonstration, we adopt the method for approximation calculation introduced by Mueller[6] in our algorithm. In his method the adaptive refinement is controlled by the vertices at every level of subdivision. The approximation is carried by an error calculation at every vertex of the original mesh before it is subdivided. This error is the distance between the vertices of the original mesh and their limit point. All the vertices that lie in the error range are labeled differently and special rules are applied for subdividing a polygon when it contains one or more of these labeled vertices.

3. The adaptive split rules

When a triangular face(cf. Fig. 2 (a)) need farther subdivision, new vertex must be counted and inserted on each edge by butterfly subdivision which makes the face be split into four triangles. We call the forenamed process as 1-to-4split(cf. Fig. 2 (b)). When splitting, the connective topology between the face and its adjoining face will be destroyed, which is known as Crack problem[10]. Here we define a series of prescripts and rules for settling the crack problem and keeping the smooth of curved surface. In order to settle the problem, some faces need be divided into two triangles as which is named 1-to-2split[8](cf. Fig. 2 (c)). Since the connective topology can be protected by 1-to-4split and 1-to-2split, a face is only allowed to do 1-to-4split and 1-to-2split in our algorithm for the purpose of computation simplified.

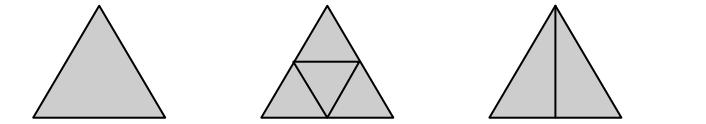

Fig. 2. Adaptive split in the triangular mesh

Then we present the prescripts and rules.

Prescript1. Only 1-to-4split and 1-to-2split are allowed to apply to triangle during subdivision.

Prescript2. The discrepancy of the subdivision level between a triangle face and its adjoining face is 1 or 0.

Rule 1. When the face being farther subdivided is produced by 1-to-4split, the adjoining face will be 1-to-2splited, if the subdivision level of the adjoining face is the same to the one of the face and it is not produced by 1-to-2split.

Rule 2. When the face being farther subdivided is produced by 1-to-4split, the adjoining face's father will be 1-to-4splited, if the subdivision level of the adjoining face is the same to the face's and it is produced by 1-to-2split.

Rule 3. When the face being farther subdivided is produced by 1-to-4split, the adjoining face's father will be 1-to-4splited and the new adjoining child face will be 1-to-2split, if the subdivision level of the adjoining face is less than the one of the face and it is produced by 1-to-2split.

Rule 4. *When the face being farther subdivided is produced by 1-to-2split, 1-to-4split its father.*

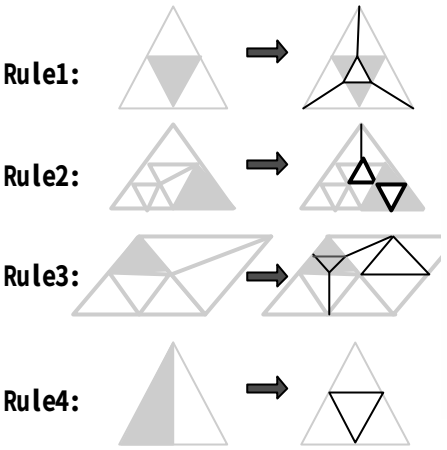

Fig. 3. **The adaptive split rules**

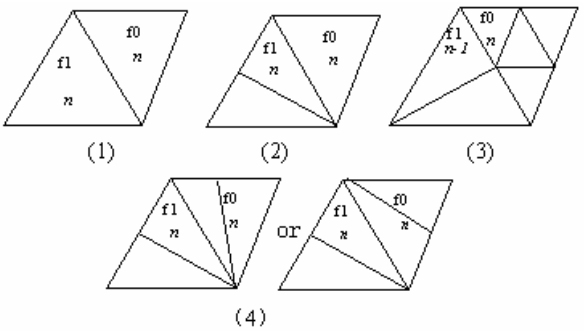

Fig. 4. Four states of adjoining triangle

For only 1-to-4split and 1-to-2split are allowed and father face is split in some condition, the rules can avoid producing thiny triangles and control the increased number of extraordinary vertex.

When a mesh subdivided with the prescripts and rules aforementioned, there are only four states that can occur between a triangular face and its adjoining faces, as is shown in figure 4. So we can only consider the four states when subdividing.

```
1-to-4 subdivide (a triangular face f)
{
        if  f is produced by 1-to-4split  then
          1-to-4split f;
          while  the tree adjoining face g of f  do
            if  g is produced by 1-to-4split  then  // Rule1
                        1-to-2split g;
                      end  if
             if  g is produced by 1-to-2split and its subdivision level is the same to f's  then
                              g=g->father;                          // Rule2
                              1-to-4 subdivide (g);
                      end if
            if  g is produced by 1-to-2split and its subdivision level is less than f's  then
                              g=g->father;                  //Rule3
                              1-to-4 subdivide (g);
                              g=the child of g which neighbor on f;
                              1-to-2split g;
                               end if
                    end while
        end if
        if  f is produced by 1-to-2split  then
                    f=f->father;
          1-to-4split f;         // Rule4
          while  the tree adjoining face g of f  do
                      if  is produced by 1-to-4split and g hasn't child  then // Rule1
                              g=g->father;
                              1-to-4subdivide(g);
                      end if
            if  g is produced by 1-to-2split and it's subdivision level is the same to f's  then
                              g=g->father;                    //Rule2
                              1-to-4subdivide(g);
                              g=the child of g which neighbor on f;
                              1-to-2split g;
                      end if
    }
```

4. Search related even vertices for computing new vertices

Since there are only four states shown in figure 5 that can occur between a triangular face and its adjoining faces, the new vertices of the △BDE in the figure5 as below contain all instances about searching even vertices.

For instance, when compute the new vertex of the edge DE, the eight even vertices found by traditional Butterfly scheme are the vertices ? in shadow faces (cf. Fig. 5). But in adaptive subdivision, to keep smooth and consistency we must find the eight even vertices ¦ in the same level with the face △BDE, which affect the new vertex ? . We compute the even vertex C by setting virtual vertices, and construct virtual split Binary-Quad Tree if need keep the information of the virtual split. For searching the other seven related even vertices, we can find them in the Binary-Quad Tree of the mesh directly. The Binary-Quad Tree is the data structure that will be introduced in the next section.

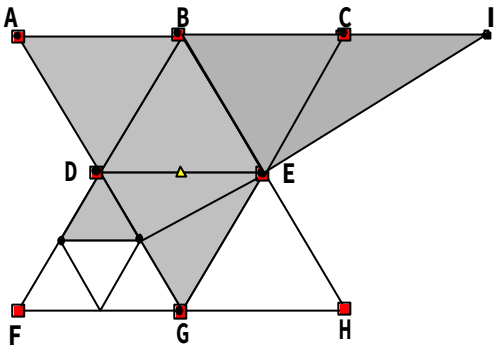

Fig. 5. Look for related even vertices for compute new vertices

```
findevenvertex(a triangular face ff)
{
    while the tree adjoining face g of f do
        if g is produced by 1-to-4split and its subdivision level is the same to f's then
                the vertex on g is one of the even vertices;
                end if
        if g is produced by 1-to-2split and its subdivision level is the same to f's then
                g=g->father;
                the vertex on g is one of the even vertices;
        end if
        if g is produced by 1-to-2split and its subdivision level is less than f's then
              //must compute virtual vertices and virtual split the face
                g=g->father;
              construct virtual split Binary-Quad Tree for subdivide g;
              the virtual vertex on g is one of the even vertices;
                g=the adjoining face of f in the virtual split Binary-Quad Tree;
        end if
        if the edge that correspond to the new vertex?  is a edge of g then
                f=g;
          goto the while sentence;
        end if
    end while
    }
```

5. The data structure base on Binary-Quad Tree

We employ a Binary-Quad tree to model the dynamic adjoining relationships of the triangles generated via subdivision (cf. Fig. 6). In a Binary-Quad tree, each parent point only has two feasibilities of four child points or two child points, which represent the new face after *1-to-4 split* and *1-to-2 split* respectively. The Binary-Quad tree stores all level information of every face. Thus each triangle in the controlled mesh will

store all of its subdivision progeny in a unique Binary-Quad tree. Insertions and removals are local, simple, and efficient. Furthermore, any level of subdivision that has already computed is readily available by just descending the tree to the desired subdivision depth. Any triangle neighbours local to the quad tree can be found by using a *Nearest Common Ancestor* traversal algorithm. And the expected cost is actually *E(1)*[2]. So we can find the adjoining faces of the same layer and confirm the eight even vertices quickly, which determine the geometric position of new vertices.

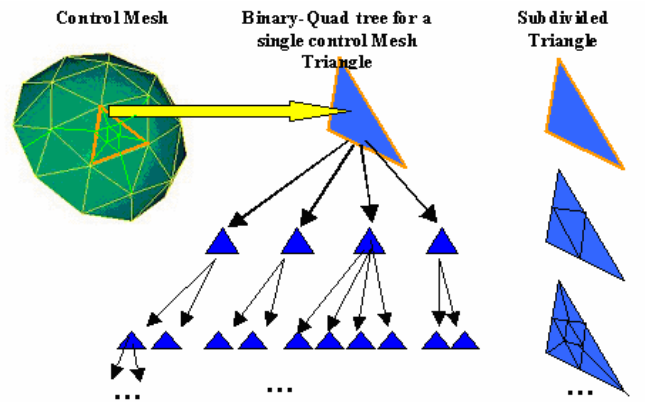

Fig. 6. A portion of a Binary-Quad Tree

6. Algorithm analysis

When subdividing the n level triangular mesh and calculating the new generated vertexes in n+1 level in Butterfly Subdivision, we must find the eight even vertexes of the Butterfly mask. But in the Adaptive Butterfly Subdivision, Not all triangular faces are subdivided in n level. So some of the even vertexes may be not exist and they must be calculated. After resolved the Crack, the level difference between two adjoining faces will be within 1. Then the level of the three adjoining faces of the face that is subdivided will be less then n. Four of the eight even vertexes are known, and no more than four vertexes need to be calculated. Because the relativity of the vertexes, it is not more than 6 virtual vertexes of n level that must be calculated when subdividing the n level triangular mesh. As a result, the number of the virtual vertexes needed to calculate is not more than C*n*(n+1). The worst time complexity of resolving the crack is O (n^2).

When searching the correlative even vertexes, if the vertex is existent, it can be found in constant time. If the vertex isn't existent, virtual vertexes will be calculated. When subdivides the n level triangular mesh, the worst time complexity of searching the correlative even vertexes is O (n^2).

So the worst time complexity of the algorithm is O (n^2).

7. Conclusions

In the paper we present an algorithm of adaptive subdivision for triangular mesh that makes use of the Modified Butterfly scheme and realizes vary subdivision level by1-to-4split together with 1-to-2split. The 1-to-4split and 1-to-2split adaptive subdivision prescripts and rules can avoid producing thiny triangles and control the increased number of extraordinary vertex. It computes the new split vertex by dint of the mask method and constructs virtual split Binary-Quad Tree. The Data Structure bases on a Binary-Quad Tree whose nodes are triangle faces.

As an advantage of the algorithm, it achieves great computation simplified and time cost decreased with a few space redundant. The space complicacy degree of this algorithm is equivalent to Kobbelt's[3] and S. Seeger's[4] but far less than Dirc. Rose's[2]. Under the general circumstance the time complicacy degree is also better than Kobbelt's and S. Segger's. In addition, the algorithm is also applicable to many other

subdivision schemes for triangular meshes, such as loop scheme, $\sqrt{3}$ scheme and so on. We focus on Modified Butterfly subdivision only for the purpose of perspicuity.

In the future work we are going to extend our algorithm to more schemes with comprehensive applicability. As we store the points of a node's child and it's father, the data capacity can be ulterior optimized. Under the worst circumstance the time complicacy degree is still bigger than Kobbelt's[3] and S. Segger's[4]. These can also be optimized in the future work.

References

1. A. Amresh, G. Farin. A. Razdan, "Adaptive Subdivision Schemes for triangle Meshes", Arizona State University, October 5, 2000.
2. D. Rose, M. Kada, T. Ertl, "On-the-Fly Adaptive Subdivision Terrain", *VMV 2001*, pp.87-92, 2001.
3. L. Kobbelt. "$\sqrt{3}$ Subdivision", *SIGGRAPH '00 Proceedings*, pp. 103-112, 2000.
4. S. Seeger, K. Hormann, G. Hausler, G. Greiner, " A Sub-Atomic Subdivision Approach", *VMV 2001*,Stuttgart, Germany, 77-86, 2001.
5. Z. Xu, K. Kondo, " Adaptive refinements in subdivision surfaces", Eurographics'99, Short papers and demos, 239-242, 1999.
6. H. Mueller, R. Jaeschke, " Adaptive subdivision curves and surfaces", *Proceedings of Computer Graphics International* '98, 48-58, 1998.
7. Zorin D, Schroder P, Sweldens W, " Interpolating subdivision for meshes with arbitrary topology [A]", *Computer Graphics*(S IGGRA PH'96 Proceedings) [C], ACM , 1996: 189～ 192.
8. K. Huang, " Quarks, Leptons & Gauge Fields", World Scientific, 1992.
9. Vasily Volkov, Ling Li, "Real-Time Refinement and Simplification of Adaptive Triangular Meshes", *14th IEEE Visualization Conference*, October 22 - 24, p.21, 2003.
10. C. Loop, " Smooth Subdivision Surfaces Based on Triangles", Master's Thesis, University of Utah, Department of Mathematics, 1987.