Meshing for Discontinuity Driven Hierarchical Radiosity

Alois A. Maierhofer Michael Gervautz Konrad F. Karner
Graz University of Technology
Münzgrabenstraße 11
A-8010 Graz
email: ali|gervautz|karner@icg.tu-graz.ac.at

Abstract

A discontinuity driven hierarchical meshing algorithm is presented which keeps polygons well shaped and avoids the generation of T-vertices. The concept of *split-hints* is introduced enabling us to store splitting information for later use rather than immediately anchoring T-vertices. The concept allows to easily combine hierarchical subdivision of a scene and introduction of discontinuity edges representing shadow boundaries. The complete splitting algorithm using split-hints is given in pseudo-code producing a triangle mesh which can be used for fast Gouraud shaded rendering.

Keywords: Radiosity, Hierarchical Refinement, Discontinuity Meshing

1 Introduction

The principal method of many radiosity algorithms is the discretization of 3D–scenes into polygonal patches. A radiosity function (or value) is calculated for each of this patches to approximate the radiosity equation for the entire scene [3]. Mostly, the given scene consists of polygons describing three-dimensional objects. Also mostly, the polygons are as large as possible with respect to the geometric resolution of the scene. To get a realistic image in the rendering step, the radiometric resolution of the scene has to be increased, because the intensity of an initial polygon can change across its face very dramatically. The usual method to take this fact into account is to subdivide the polygons to get a higher radiometric resolution and a more accurate approximation of the radiosity equation. This process is called *mesh generation* or *meshing* and is still an open problem. There are some methods for deciding *if* a patch has to be subdivided (e.g. form factor estimation [4]) but there are only a few methods for deciding *how* to split a certain face. A good description of different meshing methods can be found in [1]. A discontinuity driven meshing method was presented in [5] and in combination with hierarchical radiosity in [6]. In [5] a meshing method is described which derives *where* to split a polygon. Realizing the combination of adaptive subdivision and discontinuity driven meshing and the prevention of T-vertices is the content of our work.

2 Adaptive Subdivision combined with Discontinuity driven Meshing

The goal of all meshing methods is the approximation of the radiosity equation. We used adaptive subdivision to get as many patches as necessary but as few as possible. In the so called *hierarchical radiosity method* [4] polygons are subdivided according to a form factor estimation-criterion. If a form factor exceeds a given threshold the corresponding patch has to be splitted into smaller parts. Further splitting is necessary if shadow boundaries occur across a patch. We used the method of Heckbert described in [5] to calculate the position of discontinuity segments. A fast real-time Gouraud shading polygon-renderer is used to view the resulting images. Since this algorithm is performed in image space, the results are dependent on the orientation of the polygons in the image space [2]. This can be avoided by triangulation of all elements because Gouraud shading is orientation invariant for triangles. So additional splitting of all non-triangle patches has to be forced after the entire subdivision step. These three criteria — form factor, discontinuities, and non-triangles — determine *if* and *where* a patch has to be divided.

Dividing polygons by a given discontinuity edge and dividing simple triangles leads to the splitting of bounding edges. If the neighboring patch is not splitted in the same way a T-vertex is produced. Gouraud shading of polygons containing T-vertices would result in visual errors. In Fig. 1a vertex E of the depicted polygon is a T-vertex. The visual

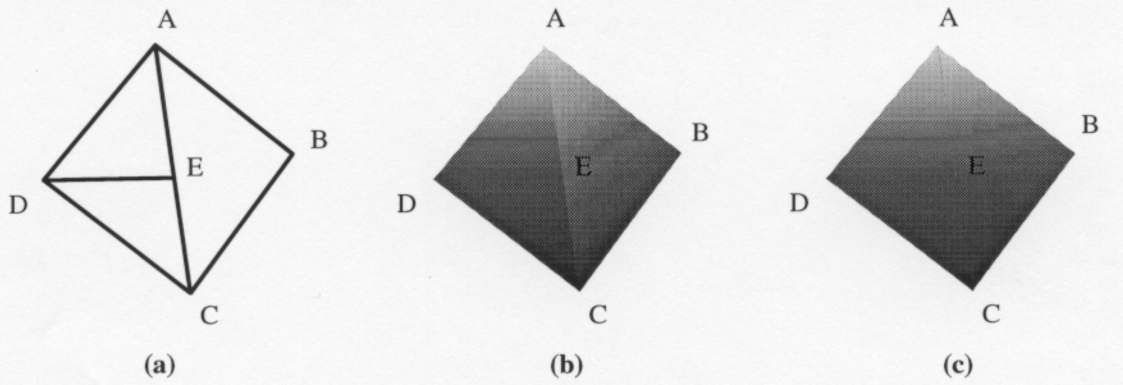

Figure 1: T-vertices. Vertex E is a T-vertex (a), Gouraud shaded polygon(b), Gouraud shaded polygon with additional vertex EB inserted(c).

artifact produced by this vertex is shown in Fig. 1b. By inserting an additional edge EB into the data-structure the visual artifact can be avoided (Fig. 1c). This technique is called anchoring. However, immediate anchoring could lead to not 'well shaped' polygons which are an assumption for an accurate form factor calculation.

We introduce the concept of *split-hints* to keep polygons well-formed taking into account discontinuity edges.

3 Split-Hints

Our strategy is to store additional information at the polygon where to divide it rather then immediately perform a subdivision. We distinguish between point split-hints and edge split-hints. Collecting the splitting information for later use allows to choose from a larger set of possibilities to subdivide a polygon and keep it well shaped.

3.1 Point Split-Hints

A *point split-hint* is the point where an edge has to be splitted. If a patch is subdivided a split-hint is stored at the neighboring patch rather than immediate anchoring this point.

Storing this information leads to a collection of split-hints for each patch. If later on in the process such a patch has to be subdivided due to the form factor-estimation-criterion we can choose the best splitting with respect to the collected hints.

3.2 Edge Split-Hints for Discontinuity Segments

Occluders between light emitters and receivers produce shadows on the light receiving faces. The shadow boundaries are known as segments of discontinuity in the radiosity function [5]. These discontinuity segments (*d-segments*) can be precalculated as described

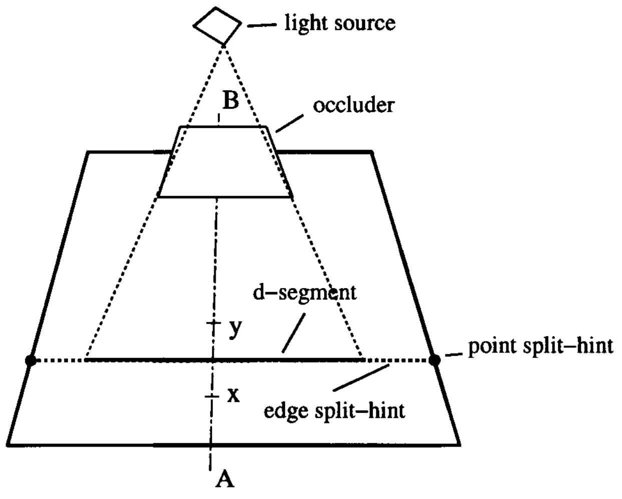

Figure 2: Edge-Vertex-Event

in [5] as a result of so called *edge-vertex* and *vertex-edge-events* leading to additional split-hints for the receiving polygon, see Fig. 2. This hints are edges in contrary to the point split-hints generated by subdivision of neighboring patches as described above. Additionally the edge split-hints are dividing boundary edges of the polygon so that point split-hints are produced for the neighboring patches. Again each patch stores a collection of these edge split-hints. If subdivision is necessary for a patch due to the form factor criterion one of these hints can be used to split the patch.

3.3 Priority of split-hints

In the general case both kinds of split-hints will be stored at a polygon. Therefore we are able to select the best splitting edge for subdivision. However, split-hints which represent discontinuity edges should have higher priority than point split-hints. Also different priorities can be used for discontinuities with different order. Higher order discontinuities should have higher priority. In this way the D^0-discontinuities are used first for splitting.

4 The Meshing Algorithm

Using split-hints the entire meshing algorithm is performed in the following way: In a preprocessing step all discontinuity segments are calculated and stored as edge split-hints.

The form factor is estimated for each pair of polygons and if the estimated value is above a certain threshold, a subdivision is performed hierarchically. According to the priority of split-hints the best splitting edge is chosen and the patch is divided. For patches with only point split-hints or without any split-hints a subdivision strategy is performed as described in section 5.

If all form factor-estimations are below the chosen value, no further splitting is necessary, but patches with split-hints still could be left. D^0-discontinuities cause shadow or light leaks in the resulting image and point split-hints refer to remaining T-vertices. Therefore these hints have to be eliminated. In a post-processing step this can be done very easily by further subdividing these patches. Remaining D^1-discontinuities can be ignored because the estimation-criterion shows that the desired radiometric accuracy is reached. The last step, which can also be performed in combination with the latter one is the division of all polygons into triangles. If this is done the data is ready to be rendered by a common Gouraud shading algorithm.

5 The Splitting Strategy

For the form factor calculation it is assumed that polygons have to be well shaped, that means that the shape of their boundary is similar to the shape of a circle. This assumption should also be fulfilled for all polygons which are produced by the splitting algorithm. In order to not produce further vertices we choose only splitting edges which are connections of two boundary points. The stored split-hints gives us more possibilities to split the polygons without producing additional vertices, we can use them in the same way as boundary points. Fig. 3 shows the possible splitting edges we can choose from.

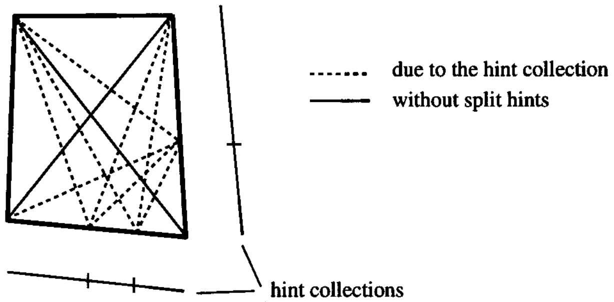

Figure 3: Possible splitting edges

For convex patches with more than three boundary points the best splitting edge is the edge that minimizes the product of the difference of the area of the resulting patches and the length of the edge:

$$|A_1^{\mathbf{e}} - A_2^{\mathbf{e}}| * |\mathbf{e}| \rightarrow_{\mathbf{e}} \min, \qquad (1)$$

with

$A_1^{\mathbf{e}} \ldots$ area of the first patch splitted with inner edge $\mathbf{e}$,
$A_2^{\mathbf{e}} \ldots$ area of the second patch splitted with inner edge $\mathbf{e}$.

This formula results from the goal to keep the new patches well formed: The size of the new patches should be as equal as possible and the length of the splitting edge should be as short as possible.

If the patch has only three boundary points we are forced to generate an additional point by splitting the longest edge into two parts. This results in two new triangle-

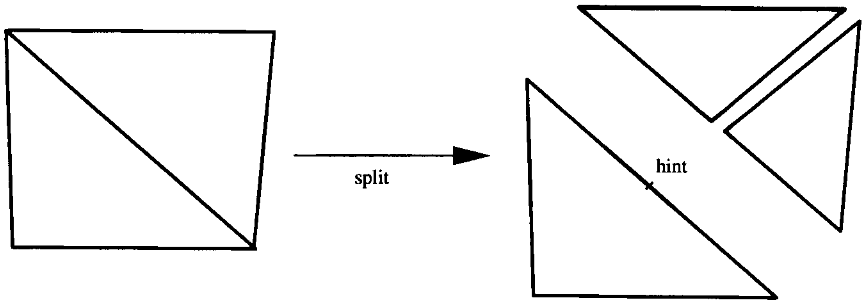

Figure 4: Split of triangle without split-hints

shaped polygons. Furthermore a point split-hint has to be produced and stored at the neighboring polygon, see Fig. 4.

6 Pseudo code of the presented algorithm

```
generate_mesh() {
    calculate all discontinuity segments
    store segments as edge split-hints to the patches
    for each pair of patches p,q with p != q
        subdivide(p,q);
    split patches until all point split-hints and all edge split-hints
      caused by d-segments of order 0 are removed
    for all polygons p with more than three vertices
        point_split(p)
}

subdivide(patch p, patch q) {
    if (formfactor_estimate(p,q) > Fmax) {
        r=max(p, q);        // choose larger patch
        s=min(p, q);        // for hierarchical subdivision
        if (r too small)
            return;         // don't split if too small
        else {
            split(r);
            subdivide(r.subpatch1, s);   // subdivide hierarchically
            subdivide(r.subpatch2, s);
        }
    }
}
```

```
split(patch p) {
    if (p has two subpatches)
        return;
    else if (p has edge split-hints)
        edge_split(p);
    else if (p has more than three boundary points)
        point_split(p);
    else
        triangle_split(p);
}

edge_split(patch p) {     // split-hints are produced
    e=edge-hint with highest priority
    split p using e as splitting edge
    store split-hints to neighboring patches
}

point_split(patch p) {     // no hints are generated
    c=collection of all polygon vertices and point split-hints of p
    e=selected edge from c as described in Equ. (1)
    split p using e as splitting edge
}

triangle_split(patch p) { // a split-hint is produced
    split the longest edge of p and generate two subpatches
    store point split-hint to neighboring patch
}
```

7 Conclusions and further work

We have implemented the algorithm described above in C++ on an SGI Indy Workstation and have tested the splitting strategy. Our experience out of the tests are that the splitting performs best for a polygon mesh consisting of initially well shaped polygons. However, further improvements can be done by introducing point split-hints into very badly shaped polygons if they occur. This will be our next step but currently we cannot report on any results of this idea.

References

[1] Daniel R. Baum, Stephen Mann, Kevin P. Smith, and James M. Winget. Making radiosity usable: Automatic preprocessing and meshing techniques for the generation of accurate radiosity solutions. In *Computer Graphics (Proc. Siggraph '91)*, volume 25, pages 51–60. ACM Press, July 1991.

[2] Michael F. Cohen and John R. Wallace. *Radiosity and Realistic Image Synthesis.* Academic Press Professional, 1993.

[3] Andrew S. Glassner. *Principles of digital image synthesis.* Morgan Kaufmann, San Francisco, 1995.

[4] Pat Hanrahan, David Salzman, and Larry Aupperle. A rapid hierarchical radiosity algorithm. In *Computer Graphics (Proc. Siggraph '91)*, volume 25, pages 197–206. ACM Press, July 1991.

[5] Paul S. Heckbert. Discontinuity meshing for radiosity. In A. Chalmers, D. Paddon, and F. Silion, editors, *Proceedings of the Third Eurographics Workshop on Rendering*, pages 203–216. Consolidation Express, April 1992.

[6] Dani Lischinski, Filippo Tampieri, and Donald P. Greenberg. Combining hierarchical radiosity and discontinuity meshing. In *Computer Graphics (Proc. Siggraph 93)*, volume 12 of *Annual Conference Series*, pages 199–208. ACM Press, August 1993.