Adaptive Mesh Refinement with Discontinuities for the Radiosity Method

W. Stürzlinger

`wrzl@gup.uni-linz.ac.at`, Johannes Kepler University, A-4040 Linz, Austria, Europe

Abstract

The radiosity method simulates the interaction of light between diffuse reflecting surfaces, thereby accurately predicting global illumination effects. One of the main problems of the original algorithm is the inability to represent correctly the shadows cast onto surfaces. Adaptive subdivision techniques were tried but the results are not good enough for general purposes. The conceptually different discontinuity meshing algorithm produces exact pictures of shadow boundaries but is computationally expensive. The newly presented adaptive discontinuity meshing method combines the speed of adaptive subdivision with the quality of the discontinuity meshing method.

1 Introduction

Radiosity has become a popular method for image synthesis due to its ability to generate images of high realism. It was first introduced to computer graphics by Goral [GTGB84]. The radiosity method models the interaction of light between diffuse surfaces ("patches"). These patches are used to store the radiosity on the respective part of the surface. The global illumination is then approximated by formulating a linear equation system for the interaction of radiosity between the patches. The progressive refinement method [CCWG88] uses a reordering of the solution process to speed up the calculation. This method works as follows: An iteration step distributes ("shoots") the radiosity of the patch with the maximum unshot radiosity to all other patches in the environment. Displaying the results after each iteration step provides the user with progressively refined approximations to the resulting picture.

To describe the influence the illumination of one patch onto another, a value - called formfactor - must be computed. These formfactors were first computed using the relatively inexact hemicube method. Wallace [WEH89] improved the accuracy by using analytic approximations for small parts of the shooting patch and ray tracing for the visibility calculations.

Another noteworthy property of the radiosity method is that it computes the illumination globally, i.e. independent of the camera position. After the radiosity algorithm has finished the scene can be viewed on a graphic workstation from arbitrary viewpoints.

2 Previous Research on Adaptive Subdivision

2.1 Patches and Elements

A drawback of the original radiosity method is the large number of patches needed to capture important detail such as shadow boundaries accurately as more patches increase the computational needs significantly.

Cohen [CGIB86] proposed a two-level hierarchy to partially overcome this problem. The patches are further subdivided into elements. The illumination is computed for the elements and are also used to display the final image. The average of the element radiosities gives a good approximation to the radiosity of the respective patch. The errors caused by shooting the radiosity from patches are sufficiently small.

Hanrahan [HSA91] generalized the subdivision hierarchy to multiple level and used an adaption of n-body problem algorithms to speed up the calculations.

2.2 Adaptive Subdivision

The radiosity algorithm with patches and elements yields unsatisfactory results when applied to general scenes as one cannot determine in advance which level of subdivision will be needed to store the illumination across a surface correctly for every possible viewpoint.

Adaptive subdivision is used to overcome this problem (see e.g. [VP91, LBT92, PWWP93]). The mesh is refined locally if the radiosity values at the element vertices vary to much or in regions with a high radiosity gradient across elements [VP91]. Such elements are considered unevenly lit, the mesh is subdivided and new radiosity values are computed for the added elements. Afterwards the algorithm is recursively applied to the new mesh elements.

This improved technique may still yield visually annoying artefacts because the resulting mesh density varies (too) quickly near shadow boundaries causing interpolation errors in the final image. A restricted quadtree scheme (see e.g. [CW93]) can be used to ensure a gradually varying mesh density.

Fortunately it is not necessary to compute the illumination for all sampling points in every iteration. In surface regions evenly lit by the current shooter only the radiosity of the original elements and associated sampling needs to be calculated. The radiosity of samples created earlier because of uneven illumination by the shooting patch of a previous iteration can be interpolated from the vertices of the original element, if the current shooter lights it evenly. This method avoids a steady increase of sample contributions to be calculated, which would cause a slowdown of subsequent iterations.

2.3 Discontinuity Meshing

A better approach would use the exact shadow boundaries for the mesh subdivision [PWWP93]. The umbra and penumbra regions can be computed exactly for a patch casting a shadow onto a plane when lit by a polgonal lightsource. The penumbra region is separated from the umbra region and the fully lit part by line segments (see Figure 1). The illumination function has discontinuities of first or second order across these line segments (see e.g. [He92, LTG92]).

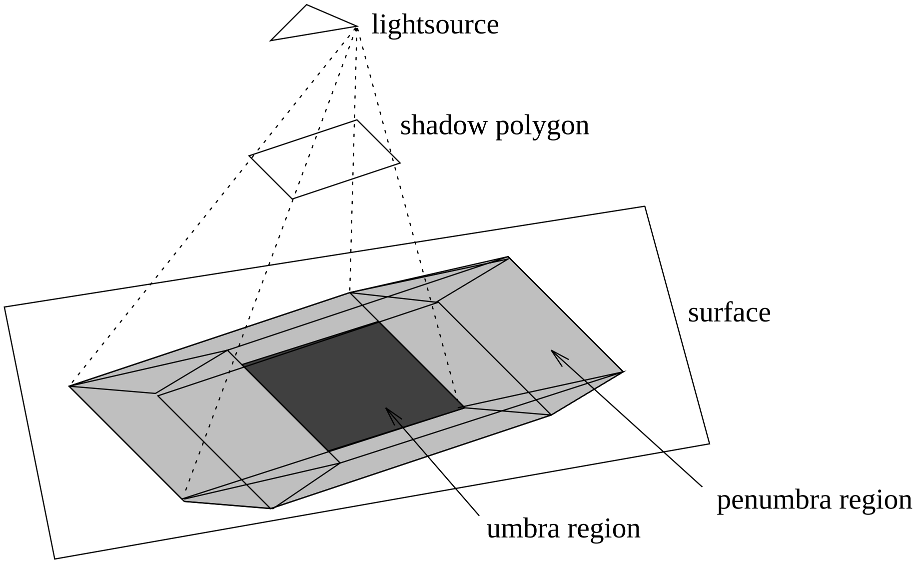

Fig. 1. Penumbra and umbra regions.

Lischinski [LTG92] presented an algorithm which calculates all shadow boundaries. Shadows cast by the current shooter onto all other surfaces are constructed exactly. The use of a BSP-tree (Binary Space Partition tree, see [FKN80]) optimizes this costly operation. The contribution of the current shooter is stored in a separate mesh. This mesh is "added" to the overall mesh after each iteration.

Heckbert [He92] calculated the shadow boundaries using a different approach based on wedges and a sweep line algorithm.

3 Adaptive Discontinuity Meshing

3.1 Motivation

The major drawback of adaptive subdivision is the fact that the required subdivision level depends on the viewing perspective of the final image. Therefore, this method is not well suited for high quality interactive viewing environments. On the one hand artifacts become noticeable at shadow boundaries near the camera (due to insufficient subdivision). On the other hand surfaces far away from the camera can do with much coarser subdivision. Therefore, the distance of a surface to the camera could be used to compute a maxmimum subdivision level. However, this method only applies to static images.

Philips [PWWP93] remarked that adaptive subdivision produces good results if exact shadow boundaries are used to subdivide the mesh but reasoned that they are too expensive to calculate.

The runtimes reported by Lischinski [LTG92] for discontinuity meshing are not far from dominating the whole illumination computation time even though a BSP-tree is used to speed up the calculation. One reason is that all discontinuity segments are calculated, regardless if noticeable or not. Shadow boundaries such as those caused by small objects close to lightsources will be practically invisible in the final images and there is no real need to compute and store them.

3.2 Adaptive Discontinuity Meshing

Adaptive discontinuity meshing is a combination of an adaptive subdivision algorithm and the discontinuity meshing method. Formfactors are calculated by raytracing [WEH89]. The subdivision algorithm is modified to use a different test and subdivision strategy. If a shadow boundary crossing an element is detected the exact shadow boundary is computed and used for mesh subdivision. As an example the shaded surface elements in figure 2 have been subdivided by the shadow boundaries which cross them.

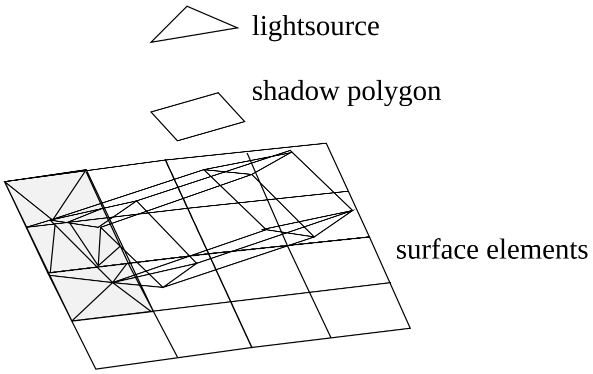

Fig. 2. Adaptive discontinuity meshing in progress.

The adaptive discontinuity meshing algorithm proceeds as follows:

First the illumination for a regular grid of sampling points is computed. The visibility of the shooting patch is determined by raycasting and those patches which obstruct the shooting patch are remembered ("light blockers"). In addition a flag stores if the shooter is fully invisible from the sample. Knowing the blocking patches for all vertices of an element it can be determined if it is potentially crossed by a shadow boundary.

Adaptive discontinuity meshing uses the following test and subdivision procedure for each element:

1. If the shooting patch is invisible from all vertices of the element then it is assumed shadowed as a whole and no further action is taken.
2. If the shooter is fully visible from all vertices and the radiosities at the vertices differ significantly the element is subdivided regularily. Then the test is applied to each of the generated (sub-)elements.
3. If the shooting patch is fully visible and the element is uniformly lit no further action is taken.
4. If the maximum difference of the vertex-radiosities is small the element is assumed to be lit uniformly and the algorithm takes no further action.
5. The number of blocking patches for the element is computed.
   (a) If too many patches (e.g. more than 4) cause shadow boundaries on the element it is subdivided regularily. Then the test is reapplied to each of the generated (sub-)elements.
   (b) Otherwise the respective shadow boundary segments for each of the blockers are constructed and the element is subdivided accordingly. This alternative is also chosen if the subdivision level reaches a user-supplied maximum.

The calculation of the shadow boundaries requires a representation of the locations where the visibility of the shooting patch changes. These locations correspond to wedges defined by the geometries of the shooting patch and the patch which causes the shadow and can be stored in a shadow BSP-tree. For a more thorough discussion see [LTG92].

In contrast to Lischinski the approach presented in this paper does not require to construct the shadow BSP-tree for the whole scene. Instead only two patches (the shooter and a "blocker") must be considered for the construction of the shadow BSP-tree caused by the blocking patch.

By intersecting the plane of the element with the BSP-tree a set of discontinuity segments is generated. As elements are small, only a few of these segments will cross the current element. The union of all shadow boundaries of all blocking patches is then used to partition the element into triangles.

The BSP-trees which were used to compute shadow boundaries are stored for the duration of the current iteration, as they might be needed for neighbouring elements. 2d- and 3d-bounding volumina for elements are used to accelerate the detection of potential shadow boundary segments.

The principal advantage of this new algorithm is that it creates considerably less elements than the original adaptive subdivision algorithm. Most shadow boundaries are represented exactly after the first subdivision step thereby avoiding the need for further subdivision in many cases. Only in complicated cases (e.g. multiple intersecting shadow boundaries) the algorithm subdivides deeper.

Instead of relying on heuristics only the algorithm classifies most cases unambiguously by using information already gathered by the visibility test. Therefore, this adaptive subdivision algorithm detects shadow boundaries crossing elements more reliably than the previously published methods.

Due to better shadow boundary detection, the threshold for unevenly lit elements can be set higher than in the original adaptive subdivision algorithm as shadow boundaries are already explicitely accounted for.

Primary lightsources and strong secondary lightsources cause the most prominent discontinuities. These will be detected reliably by the algorithm. Unnoticeable i.e. weak shadow boundaries are ignored and cause no discontinuity segments to be generated. In contrast to Lischinski [LTG92] who used the heuristic of computing the discontinuities only for the primary lightsources, this algorithm computes shadow boundaries only where neccessary, i.e. where they are visible in the final images or walk through environment.

3.3 Improved Adaptive Discontinuity Meshing

Due to the regular grid of the original elements aliasing effects can occur (e.g. small sharp shadow details might be missing).

In a surface region with changing lightning conditions the mesh density will vary abruptly causing interpolation artifacts. Such artifacts can be avoided by using a scheme analogous to the restricted quadtree. If the subdivision levels of neighbouring elements differ by more than one the respective element is subdivided and the algorithm is applied recursively.

Many artifacts appear also at the boundary of a region where elements were split by discontinuity segments and others were not (e.g. figure 3), The artifacts are caused by t-vertices and can be avoided by the anchoring scheme described by Baum [BRW89].

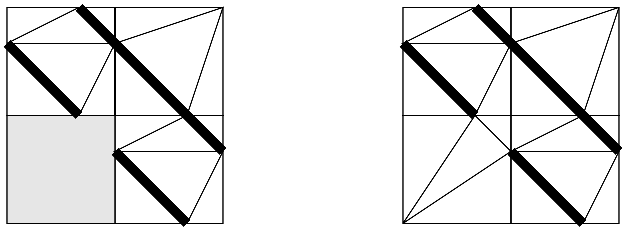

Fig. 3. T-vertices (left) and anchoring (right). Discontinuity segments are drawn in bold.

The techniques presented in section 2.2 are used to avoid a steady increase of sample contributions to be computed.

3.4 Reconstruction of the Illumination Function

One method for reconstructing the illumination function from the radiosity values is by linear interpolation across elements known as Gouraud shading – a polygon display method often implemented in hardware on graphic workstations. The resulting illumination function is continuous in value (C^0). This technique allows an interactive walkthrough of the environment.

As illumination varies smoothly across surfaces (except at shadow boundaries) higher order reconstruction methods can be used, as an alternative. Lischinski [LTG92] used quadratic and Salesin [SLR92] cubic Bezier-triangles. Both approaches exploit the precalculated discontinuities to correctly approximate the illumination function. The reconstruction algorithm of Salesin [SLR92] provides a C^1-smooth interpolation of the surface illumination.

4 Implementation and Results

Our simple test scene consisted of a cube floating above a plane lighted by two lightsources was modeled (analguous to [LTG92]).

Figure 4 shows the mesh in the upper part and a Gouraud shaded version in the lower part of the picture. The pictures were generated without the modifications discussed in section 3.3. The statistics for the two versions are given in the following table, where AS stands for the original adaptive subdivision method and ADM stands for the new adaptive discontinuity meshing method. Times are given in seconds for a SGI Indigo R3000 for the first two interations (corresponding to the two lightsources).

	AS 1. iter	AS 2. iter	ADM 1. iter	ADM 2. iter
Total number of elements	1361	2057	326	828
Iteration time (sec)	4	3	4	5

The mesh created by the adaptive discontinuity meshing algorithm consist of considerably less elements compared to the original adaptive subdivision algorithm, resulting also in less memory usage. Due to the more complex shadow boundary detection and subdivision algorithm the adaptive discontinuity meshing method uses more time per element but the total times used by both algorithms are compareable. As can be seen in figure 4 the visual quality of the result of the adaptive discontinuity meshing method is better, though.

The algorithm with the modifications of section 3.3 was tested with a scene consisting of 4738 patches. The method took about 13 % longer than the adaptive subdivision algorithm. The timings are given for the first iteration.

	AS	ADM
Total number of elements	104689	35897
Iteration time (sec)	559	630

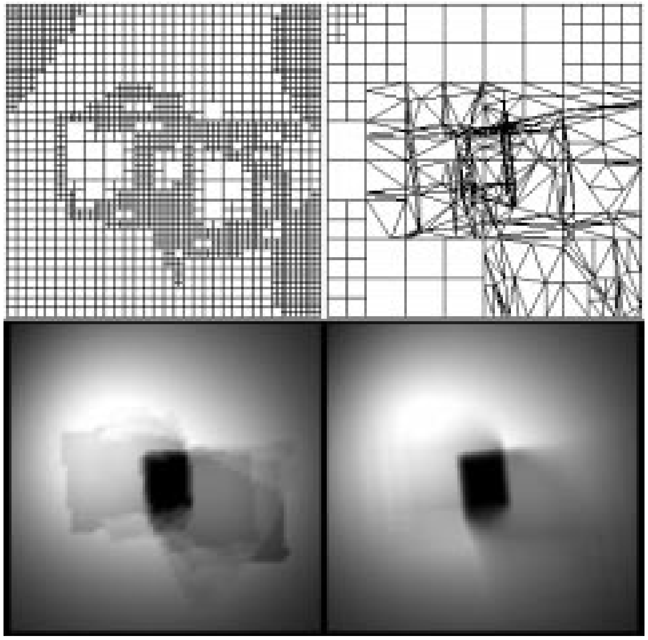

Fig. 4. Adaptive subdivision (left) and adaptive discontinuity meshing (right).

However the visual quality of the resulting image (see figure 5) is superior to the image computed by ordinary adaptive subdivision.

5 Conclusion and Further Extensions

The adaptive discontinuity meshing method presented in this paper combines the speed of adaptive subdivision with the image quality of the discontinuity meshing method.

- Compared to the adaptive subdivision method the adaptive discontinuity meshing algorithm delivers a result in compareable time.
- Significantly fewer elements are generated which reduces memory requirements.
- Due to the improved shadow boundary representation the resulting images are visually more accurate.
- This method introduces an new approach for reliable adaptive subdivision criteria.
- Adaptive discontinuity meshing can be used as an extension to existing radiosty systems and/or parallel implementations of the radiosity algorithm (e.g. [St94]).

Fig. 5. Complex scene with adaptive discontinuity meshing.

In this paper only Gouraud shading was used for the resulting images. The potential effect of higher order interpolation methods remains to be investigated.

Hierarchical discontinuity meshing [LTG93] is another practicable way to improve image quality and to obtain the image quickly as well. A noteable disadvantage of all hierarchical methods reported by [Ca94, To94] are significant problems in parallelizing them for distributed memory machines.

5.1 Acknowledgements

The author thanks G. Schaufler for reading an early version.

References

[BRW89] Baum, D. R., Rushmeier, H. E., Winget, J. M.: Improving Radiosity Solutions through the Use of Analytically Determined Form-Factors. Computer Graphics (SIGGRAPH '89) **23**:3 (1989) 51–60

[Ca94] Carter, M.: Private communications.

[CGIB86] Cohen, M., Greenberg, D. P., Immel, D. S., Brock, P. J.: An Efficient Radiosity Approach for Realistic Image Synthesis. IEEE Computer Graphics and Applications **6**:3 (March 1986) 26–35

[CCWG88] Cohen, M., Chen, S. E., Wallace, J. R., Greenberg, D. P.: A Progressive Refinement Approach to Fast Radiosity Image Generation. Computer Graphics (SIGGRAPH '88) **22**:4 (1988) 75–84

[CW93] Cohen, M. F., Wallace, J. R.: Radiosity and Realistic Image Synthesis. Academic Press (1993)

[FKN80] Fuchs, H., Kedem, Z. M., Naylor, B.: On Visibile Surface Generation by a Priori Tree Structures. Computer Graphics (SIGGRAPH '80) **14**:3 (July 1980) 124–133

[GTGB84] Goral, C. M., Torrance, K. E., Greenberg, D. P., Battaile, B.: Modelling the Interaction of Light Between Diffuse Surfaces. Computer Graphics (SIGGRAPH '84) **18**:3 (1984) 212–222

[He92] Heckbert, P.: Discontinuity Meshing for Radiosity. Third Eurographics Workshop on Rendering (May 1992) 203–226

[HSA91] Hanrahan, P., Salzman, D., Aupperle, L.: A Rapid Hierarchical Radiosity Algorithm. Computer Graphics (SIGGRAPH '91) **25**:4 (July 1991) 197–206

[LBT92] Languenou, E., Bouatouch, K., Tellier, P.: An adaptive Discretization Method for Radiosity. Proceedings of Eurographics 1992 **11**:3 (1992) 205–216

[LTG92] Lischinski, D., Tampieri, F., Greenberg, D. P.: A Discontinuity Meshing Algorithm for Accurate Radiosity. IEEE Computer Graphics and Applications **12**:6 (Nov 1992) 25–39

[LTG93] Lischinski, D., Tampieri, F., Greenberg, D. P.: Combining Hierarchical Radiosity and Discontinuity Meshing Computer Graphics (SIGGRAPH '93) **27**:4 (August 1993) 199–208

[PWWP93] Philips, S., Worrall, A., Willis, C., Paddon, D.: Adaptive Mesh Refinement for the Radiosity Method. Proceedings of Compugraphics (1993) 178–186

[SLR92] Salesin, D., Lischinski, D., DeRose, T.: Reconstructing Illumination Functions with Selected Discontinuities. Third Eurographics Workshop on Rendering (May 1992) 99–112

[St94] Stürzlinger, W., Wild, C.: Parallel Progressive Radiosity with Parallel Visibility Computations, Winter School of Computer Graphics 94 (Plzen, CZ) (Feb. 1994) 66–74

[To94] Tobler, R. F.: Private communications.

[VP91] Vedel, C., Puech, C.: A testbed for adaptive subdivision in progressive radiosity, Second Eurographics Workshop on Rendering (May 1991)

[WEH89] Wallace, J. R., Elmquist, K. A., Haines, E. A.: A Ray Tracing Algorithm for Progressive Radiosity. Computer Graphics (SIGGRAPH '89) **23**:3 (July 1989) 315–324

This article was processed using the LaTeX macro package with LMAMULT style