Exact and Error Bounded Approximation of Local Illumination

Gernot SCHAUFLER and Wolfgang STÜRZLINGER
Johannes Kepler University Linz (GUP)
Altenbergerstraße 69, A-4040 Linz, Austria/Europe
Tel.: +43 732 2468 9228, Fax: +43 732 2468 9496
Internet: [schaufler | stuerzlinger]@gup.uni-linz.ac.at

ABSTRACT

Recent approaches to realistic image synthesis split the rendering process into two passes. The first pass calculates an approximate global illumination solution, the second produces an image of high quality (from a user selected view point) using the solution obtained in the first pass by applying the local illumination model to each surface point visible through each pixel. This paper presents a new method how to compute the visible surfaces as seen from a surface point - the hemisphere projection. This method allows the exact evaluation of the local illumination model and facilitates the fast and accurate computation of form factors taking occlusion into account. Using the hemisphere projection an exact local pass solution can be obtained. In addition the hemisphere projection can be used to compute an approximation of a point's local illumination to within given error bounds in significantly less time.

Key Words: image generation, local illumination, local pass, error bounds

1 INTRODUCTION

The light leaving a surface is determined by the incoming light and by the material properties of the surface - mathematically described by the local illumination model. The problem is how to accurately calculate the light reaching the surface which in turn is determined by the light emitted by all other surfaces and taking the scene geometry and occlusion into account.

An approximation to the light leaving all surfaces is calculated by global illumination methods which simulate the distribution of light in an environment. Light emitted by a lightsource is either absorbed by the hit surfaces or reflected to other surfaces and so on. By discretizing the environment into patches (i.e. planar polygons) and under the assumption of diffuse surfaces an equation system can be formulated which describes the mutual influence of patches. Solving this system gives the emitted light (the radiosity) for each patch. For an image of the environment the polygons are rendered from a user selected view point with hidden surfaces removed. However, the chosen patch size limits the quality of the approximation and leads to artefacts in the final image. These artefacts are due to an interpolation of radiosity values at inappropriate places or over unacceptably large areas as neither the final view point nor the image resolution are known at this stage. For an overview of global illumination methods see Cohen *et al.* or Sillion *et al.* [Cohen-Wallace93,Sillion-Puech94].

More correct images are obtained by a so called local pass. First the surface point visible through each pixel is determined, then the light leaving the surface at this point is calculated from the incident illumination (using the results of the global illumination solution) and the material properties.

This paper presents two new methods for local illumination calculation. First a method to represent the surfaces visible in any direction from a surface point is discussed which allows to compute the exact irradiance. The second contribution is a way to approximate the incident light with given error bounds.

2 LOCAL PASS: LOCAL ILLUMINATION AT EACH PIXEL

The local pass method evaluates the local illumination model at each surface point visible through the pixels of an image. This allows to compute high quality images from an approximate global illumination solution and avoids the artefacts of the discretized global illumination solution.

Based on a coarse solution of the global illumination problem a high quality picture is generated by finding the surface point *x* visible at each pixel and obtaining the colour of this pixel by computing the irradiance of *x* and applying the local illumination model.

The local illumination model describes the relationship between the irradiance *H* of a point *x* (the "incoming" light) and the reflected light. For the diffuse case the radiosity *B* equals the irradiance multiplied by the reflectivity of the surface plus its emission:

$$B(x) = E(x) + \rho(x) H(x) \tag{1}$$

where *E* is the emission and ρ is the reflectivity of *x*'s surface. The incoming light for a point *x* is given by

$$H(x) = \frac{1}{\pi} \int_{\Omega} B(\vec{\omega}) \cos\Theta \, d\vec{\omega} \tag{2}$$

which integrates the radiosity *B* coming from all possible directions $\vec{\omega}$ on the hemisphere Ω weighted by the cosine of the angle Θ between $\vec{\omega}$ and the normal vector of *x*'s surface. The radiosity coming from each direction $B(\vec{\omega})$ depends on the surface point *y* visible in the direction $\vec{\omega}$ and its radiosity *B(y)* obtained from a previous global illumination solution.

To calculate the irradiance from a fully visible polygon *A* with constant radiosity *B* it suffices to solve the contour integral for equation (2) which gives (see e.g. [Baum et al. 89]):

$$H(x, A) = B(A) \cdot \underbrace{\frac{1}{2\pi} \sum_{a \in A} N \cdot \Gamma_a}_{FF(x, A)} \tag{3}$$

where *a* are the edges of the polygon and *N* is the normal vector of *x*'s surface. Γ_a is the vector normal to the plane defined by *x* and the edge *a* and length equal to the angle γ_a (see figure 1). The geometry dependent term in equation (3) *FF(x,A)* is also called the form factor. For an alternate way of solving equation (2) using spherical triangles see [Bian92].

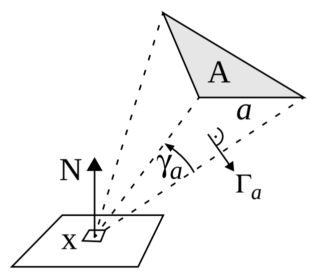


Figure 1: Geometry for form factor of a polygon

General scenes consist of more than one polygon and occlusion must be taken into account when solving the integral over the hemisphere. Let $A^{vis(x)}$ be the part(s) of *A* visible from *x*. The irradiance *H(x)* can then be calculated by

$$H(x) = \sum_{A} B(A) \cdot FF(x, A^{vis(x)}) \tag{4}$$

The visible parts of all polygons can be identified by projecting them onto a virtual unit hemisphere with center *x* above the surface of *x*. The new hemisphere projection described in chapter 3 is a method to obtain and store the surfaces visible in all directions of the hemisphere.

Other solutions to equation (2) have been obtained using stochastic methods (e.g. [Rushmeier88,Shirley91,Ward94]).

3 PROJECTIONS ONTO THE HEMISPHERE

Accurately representing the projection of polygons onto the hemisphere allows to identify the visible parts of all polygons as seen from a point *x*. Considering occlusion allows to compute the exact irradiance for a surface point.

3.1 Previous Work

First approximations to a projection onto the hemisphere were computed using a hemicube [Cohen-Greenberg85]. The polygons are rasterized onto the five faces of a hemicube with z-buffering to account for visibility. Other approaches use a tetrahedron [Beran-Koehn-Pavicic91,Spencer91], discretizations of the hemisphere [Gatenby-Hewitt91] or the projection onto a single plane [Sillion-Puech89,Recker et al. 90].

All above methods have in common that they suffer from aliasing problems due to the rasterization of the projection plane(s). They do not deliver exact results for arbitrary polygonal environments with occlusion as they fail to represent the visible surface points in all directions of the hemisphere accurately.

3.2 Exact Projection onto the Hemisphere

An exact projection of a polygon *A* with vertices v_j onto the unit hemisphere defined by a projection center *x* and its surface normal vector *N* can be calculated in the following way.

The intersection of a ray from the projection center *x* to a polygon vertex v_j with the unit hemisphere gives the projected vertex $\bar{v}_j$, which is calculated by normalizing the ray direction. The projection of the polygon edge defined by v_1 and v_2 onto the hemisphere is curved, it is a segment of a great circle. However the projected vertices $\bar{v}_1$ and $\bar{v}_2$ lie in the plane ε_1 defined by *x*, v_1 and v_2 which can be used to represent a projected edge unambiguously.

Testing beforehand if the polygon *A* faces the projection center *x* removes backfacing polygons. Translating *x* to the origin allows to specify each ε_i by its normal vector only. Clipping the polygon *A* at the plane defined by *x* and *N* before projection assures that all projected vertices fall onto only one half of the unit sphere at *x*. This obviates the need to rotate the coordinate system to align the normal vector *N* with any axis.

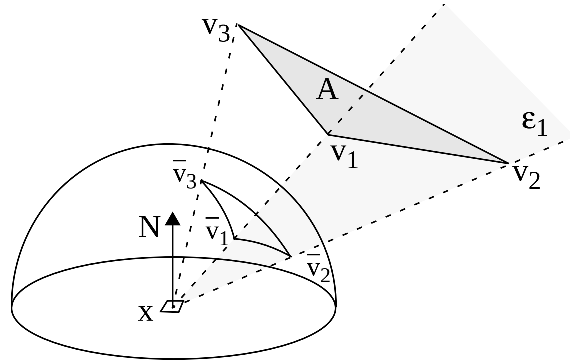


Figure 2: Projection of polygon onto hemisphere

3.3 Intersecting Projections

From Nusselt's analogon it is clear that projections onto the hemisphere can be represented by two dimensional data structures. Intersecting two projected polygons is therefore topologically equivalent to clipping two dimensional polygons against each other which is done using polygon intersection routines e.g. the Weiler-Atherton algorithm [Weiler-Atherton-77]. Modifications are needed for the point-against-line test and for the intersection computation of two edges to work on the hemisphere rather than in two dimensions. To avoid testing a polygon against all previously projected polygons a two dimensional binary space partitioning (BSP) tree [Fuchs80] can be used.

Testing a projected vertex against a projected edge (defined by a plane ε_i) is equivalent to deciding on which side of the plane the vertex lies (at the cost of one dot product).

The intersection of two edges is computed as the line of intersection of the two planes containing the edges and the hemisphere center. This line is defined by the cross-product of the two plane normals and the projection center. Normalizing the cross-product gives the point on the hemisphere $\bar{i}$ corresponding to the intersection of the two lines (see figure 3).

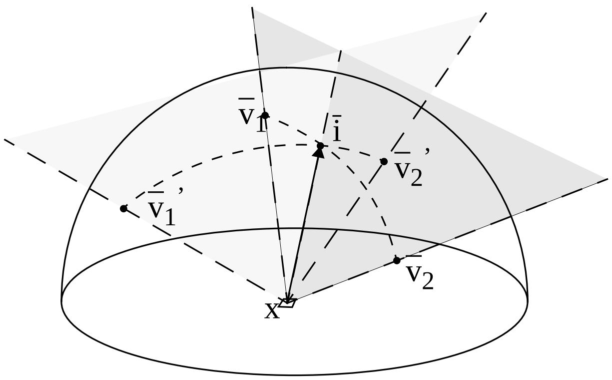


Figure 3: Intersection of two projected edges

3.4 Visible Surface Determination

Assume that with each projected vertex the distance from the projection center *x* to the original vertex v_j is stored as *dist*($\bar{v}_j$) and for each projected polygon a reference to the original polygon is kept as well.

Consider two polygons *A* and *A'* whose projections $\bar{A}$ and $\bar{A}$' onto the hemisphere intersect, and *A'* further away from the projection center *x*. Then $\bar{A}$' has to be clipped to $\bar{A}$ using the modified polygon clipping method described above. If the original polygons do not intersect in space all *dist*($\bar{v}_j$) are smaller than all *dist*($\bar{v}_k$'). (A similar case holds if *A'* lies in front of *A*).

If the polygons do intersect one of them must be partitioned by the base plane of the other polygon and both parts have to be handled separately.

3.5 An Algorithm for the Local Pass

As equation (3) is based on angles between vectors only, it can be used to compute the irradiance at a point *x* coming from the visible parts of a polygon $A^{vis(x)}$. Summing the irradiance of each polygon gives the total irradiance *H(x)* from which the point's radiosity is calculated using equation (1). The calculated radiosity is exact to within the error introduced by the inaccuracy of the global illumination solution.

The local pass method applies equation (1) at each pixel to compute an exact image of a scene with global illumination. Instead of projecting all patches onto the hemisphere they can be rendered front to back until the whole hemisphere is covered. This can be done traversing a spatial subdivision hierarchy from the point *x* outwards (see e.g. [Wang-Davis90]). The form factors of the visible parts of the patches are determined using the results of the hemisphere projection and their radiosity weighted by the form factor is accumulated to the irradiance of point *x*.

```
for all pixels
      find surface point x visible through pixel
      Covered = 0
      while Covered < 1.0
            v = next voxel front to back
            project patches of voxel v onto the hemisphere
            Covered += area covered by patches of voxel v
      endwhile
      H(x) = 0
      for each patch-part A visible on the hemisphere
            H(x) += FF(x,A) * B(A)
      endfor
      B(x) = ρ(x) * H(x)
endfor
```

Figure 4: Pseudocode for local pass

4 APPROXIMATE LOCAL ILLUMINATION

The complexity of a brute force local pass is proportional to the number of pixels times the cost of projecting all patches onto the hemisphere. Projecting all patches may take $O(n^2)$ time, as scenes exist where the projection has $O(n^2)$ parts (see e.g. [Foley90]).

One approach to deal with this cost is to project less patches onto the hemisphere, namely only those having significant influence on the illumination of the point under consideration. For the other patches a coarse approximation of their illumination can be used to estimate their influences. This is also motivated by the fact that ray tracing often achieves acceptable results by simply point sampling the light sources.

In computing an approximation to the irradiance decision criteria are needed which polygons to project accurately and the contribution of which to approximate. These criteria can be based on bounds on the contribution of a part of the scene for the irradiance of a point *x*.

4.1 Bounding the Energy

Assume that the results of the global illumination simulation are stored in an octree (any other hierarchical space subdivision scheme could be used as well).

For a voxel *v* of the octree an upper bound for the radiosity it can radiate can be defined as:

$$\lceil B(v) \rceil = \max_i B(A_i) \quad (5)$$

which says that the whole voxel emits a radiosity equal to the maximum radiosity of any patch in the voxel.

The octree is built by a recursive procedure which considers all patches at the root node. Either the node is subdivided into eight children with patches split if necessary, or the patches are stored with the current voxel if the following criterion is met:

$$(\lceil B(v) \rceil \cdot \mathrm{diagonal} < \mathrm{EnergyThreshold}) \vee (\mathrm{level} > \mathrm{maxlevel}) \quad (6)$$

where *diagonal* is the voxel diagonal, *EnergyThreshold* bounds the energy one voxel can radiate, *level* is the current hierarchy level, and *maxlevel* is the maximal allowable depth of the octree. The left part of the criterion tries to put equal amounts of energy into each octree leaf.

With a view dependent energy bound the voxels can be sorted by the influence they have when computing the illumination of a surface point.

4.2 View Dependent Energy Bounds

An upper bound to the maximum contribution of a voxel *v*'s energy to the irradiance of a view point *x* can be given by:

$$\lceil B(x, v) \rceil = \lceil B(v) \rceil \cdot FF(x, v) \quad (7)$$

where *FF(x,v)* is the form factor of the projection of the silhouette of the voxel (see figure 5).

If the view point *x* is inside the voxel the form factor is defined to be 1.

A tighter upper bound for the voxels contribution takes the orientation of the patches with respect to point *x* into account:

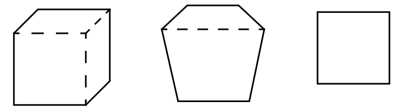


Figure 5: Possible silhouettes of a voxel

$$\lceil B(x, v) \rceil = FF(x, v) \max_i B(A_i) \left(\vec{xc} \bullet N(A_i) \right) \quad (8)$$

where $N(A_i)$ is the normal of patch A_i and $\vec{xc}$ is the vector from *x* to the center of A_i.

This upper bound for the voxels contribution can be used to sort the voxels by decreasing potential influence on point *x*. The total possible contribution of the patches in the octree is bounded by:

$$\lceil B(x, \mathrm{Octree}) \rceil = \sum_{v \in \mathrm{Octree}} \lceil B(x, v) \rceil \quad (9)$$

4.3 Error Bounded Approximation of the Local Pass

An error bounded approximation to the irradiance of point *x* can be calculated by accumulating the contribution arriving from each voxel *v*. The voxels are sorted in descending order of $\lceil B(x, v) \rceil$ to guarantee that the voxels are processed in the order of decreasing possible contribution.

After the visible parts of the patches in voxel *v* as seen from point *x* have been determined their contribution is added to the irradiance of point *x*. Let B_{Octree} be the upper bound to the energy remaining in the octree. After a voxel *v* has been processed B_{Octree} can be decreased by $\lceil B(x, v) \rceil$.

In order to approximate the irradiance of point *x* up to a certain tolerance, it suffices to project voxels until the maximum energy remaining in the octree has fallen below a given percentage of the current approximation of the irradiance *H(x)*. This criterion can be enhanced by using the fact that the voxels are projected front to back and taking into account the percentage of the hemisphere already covered by (previously) projected patches:

$$B_{Octree} \cdot (1 - Covered) < Tolerance \cdot B(x) \quad (10)$$

The algorithm is summarized as pseudocode in figure 6 .

```
for all pixels
     find surface point x visible through pixel
     H(x) = 0
     Covered = 0
     B_Octree = ⌈B(x, octree)⌉
     while B_Octree * (1-Covered) > Tolerance * H(x)
          v = next voxel in order of decreasing ⌈B(x, v)⌉
          compute projection v̄
          B_Octree -= ⌈B(x, v)⌉
          if depth(v̄) < depth(polygons of v̄ on hemisphere)
               SolveVisibility(x,v)
               project patches of voxel v onto hemisphere
               H(x) += Σ_{A ∈ v} B(A) · FF(x, A)
               Covered += area covered by patches of voxel v
          endif
     endwhile
     H(x) += approximation of portion of B_Octree arriving at x
     B(x) = ρ(x) * H(x)
endfor

SolveVisibility(x,v)
     project all voxels v_i possibly hiding v onto hemisphere
     remember v_i has been projected (do not project it again)
```

Figure 6: Pseudocode for optimized local pass

The visibility of the patches of each voxel *v* is determined by projecting all voxels onto the hemisphere which lie between *x* and *v*. By remembering which voxels have already been projected, projecting them again either in `SolveVisibility` or in the main loop is avoided. After `SolveVisibility` all patches possibly hiding patches in voxel *v* have been projected. Therefore, by projecting the patches of voxel *v* the correct visibility of the patches is determined.

5 IMPLEMENTATION AND RESULTS

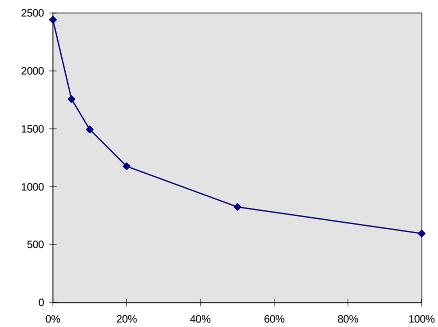
The current implementation of the hemisphere projection and local pass method runs on a Silicon Graphics Indigo Workstation (R3000). The results were obtained using a scene consisting of approx. 1300 polygons. which were split into 2985 patches by the octree (see figure 8).

The exact method presented in chapter 3 was implemented to compute reference solutions. To speed up the projection a BSP tree was used.

Test showed that the error bound provided by equation (7) overestimates the contribution by several orders of magnitude in general which results in poor algorithm performance.

Using equation (8) up to 56 percent of the upper bound of the energy contribution actually contributed to the irradiance in the test runs.

The average number of projected polygons and the relative maximum image error are shown in relation to the tolerance parameter (see figure 7).



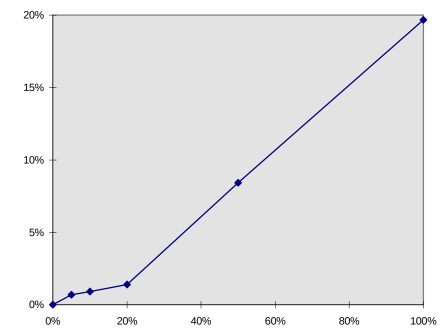


Figure 7: Number of patches projected per pixel, maximum error in image in relation to tolerance parameter

The upper diagram shows e.g. that on the average approx. 1200 (=40%) patches have to be projected so that the maximum remaining energy falls below 20%. As can be seen from the lower diagram the corresponding maximum image error was 2% which is accurate enough for most images. Tests with other scenes showed that the given results are representative.

Figure 8: Living room scene

6 CONCLUSION AND FURTHER EXTENSIONS

This paper presents a new exact method to compute and store the surfaces visible from an arbitrary surface point. This allows to compute the exact illumination of a surface point. Furthermore error bounded approximations to the irradiance were calculated and it was shown that considerable savings are possible.

A different way to reduce the cost of the local pass is to evaluate the local illumination integral less often and use some kind of interpolation scheme if applicable. Arvo [Arvo94] introduced a method to compute irradiance gradients which can be computed simply by using the results of the hemisphere projection method introduced in this paper. Gradients have already been applied successfully to identify areas where irradiance can be interpolated (see [Ward94]) which shows the feasibility of the above approach.

7 REFERENCES

Arvo J., *The Irradiance Jacobian for Partially Occluded Polyhedral Sources*, Computer Graphics (SIGGRAPH 94), pp. 343-350, July 1994.

Bian B., *Hemispherical Projection of a triangle*, In Graphics Gems III, D. Kirk, Ed. Academic Press, San Diego, 1992, pp. 314-317.

Beran-Koehn J. C., and Pavicic M. J., *A cubic tetrahedal adaption of the hemi-cube algorithm*, In Graphics Gems II, J. Arvo, Ed. Academic Press, San Diego, 1991, pp. 229-302.

Baum D. R., Rushmeier H. E., and Winget J. M., *Improving radiosity solutions through the use of analytically determined form-factors*, Computer Graphics (SIGGRAPH 89), pp. 325-334, July 1989.

Cohen M. F., and Greenberg D. P., The hemicube: *A radiosity solution for complex environments*, Computer Graphics (SIGGRAPH 85), pp. 31-40, July 1985.

Cohen M. F., and Wallace J. R., *Radiosity and Realistic Image Synthesis*, Academic Press, 1993.

Foley J. D., van Dam A., Feiner S. K., and Hughes J. F., *Computer Graphics: Principles and Practice*, Second Edition, Addison-Wesley, 1990.

Fuchs H., Kedem, Z. M., and Naylor B. F., *On Visibile Surface Generation by A Priori Tree Structures*, Computer Graphics (SIGGRAPH 80), pp. 124-133, July 1980.

Gatenby N., and Hewitt W. T., Radiosity in Computer Graphics: *A Proposed Alternative to the Hemi-cube Algorithm*, Eurographics Workshop on Rendering 1991.

Recker R. J., George D. W., and Greenberg D. P., *Acceleration techniques for progressive refinement radiosity*, Computer Graphics (1990 Symposium on Interactive 3D Graphics), pp. 59-66, Mar. 1990.

Rushmeier H. E., *Realistic Image Synthesis for Scenes with Radiatively Participating Media*, Ph.D. thesis, Cornell University, 1986.

Shirley P., *Physically Based Lighting Calculations for Computer Graphics*, Ph.D. thesis, University of Illinois at Urbana-Champain, 1991.

Spencer S. N., *The hemisphere radiosity method: A tale of two algorithms*, In Photorealism in Computer Graphics (Proceedings Eurographics Workshop on Photosimulation, Realism and Physics in Computer Graphics, 1990), K. Bouatouch and C. Bouville, Eds., 1991, pp. 127-135.

Sillion F., and Puech C., *A general two-pass method integrating specular and diffuse reflection*, Computer Graphics (SIGGRAPH 89), pp. 335-344, July 1989.

Sillion F. X., and Puech C., *Radiosity and Global Illumination*, Morgan Kaufmann Publishers, 1994.

Ward G. J., *The RADIANCE Lighting Simulation and Rendering System*, Computer Graphics (SIGGRAPH 94), pp. 459-472, July 1994.

Weiler K., and Atherton P., *Hidden Surface Removal Using Polygon Area Sorting*, Computer Graphics (SIGGRAPH 77), pp. 214-222, 1977.

Wang Y., and Davis W. A., *Octant Priority for Radiosity Image Rendering*, Graphics Interface ‘90, pp. 83-91, 1990.