Interner Bericht

Shadow-Mapping - A Method of Evaluating and Storing Diffuse Radiosities in Static Environments

Philip Jacob

234/93

Fachbereich Informatik

Universität Kaiserslautern · Postfach 3049 · D-67653 Kaiserslautern

Shadow-Mapping - A Method of Evaluating and Storing Diffuse Radiosities in Static Environments

Philip Jacob

234/93

Universität Kaiserslautern
AG Computergraphik
Postfach 30 49
67653 Kaiserslautern

Oct. 1993

Herausgeber: AG Graphische Datenverarbeitung und Computergeometrie
Leiter: Professor Dr. H. Hagen

Shadow-Mapping - A Method of Evaluating and Storing Diffuse Radiosities in Static Environments

Dipl. Inform. P. Jacob

University of Kaiserslautern
234/93
AG Computer Graphics
Postfach 3049
67653 Kaiserslautern
Email:jacob@informatik.uni-kl.de

October 1993

1. Abstract

Most radiosity techniques store radiosities in certain sample points, typically the vertices of polyhedral scenes. As diffuse radiosities are view independent they can be used for an interactive 'walk-through'. This paper presents an algorithm for storing radiosities independent of the representation of the object. A distributed rendering system, which uses this shadow-mapping technique is described. The basic thermophysical definitions, needed to derive a sum formula for a form factor calculation of polygons, are explained.

Keywords: radiosity, texture mapping, form-factors

2. Introduction

The main reason why most illumination models have been developed for diffuse environments is that for a certain configuration of light sources, intensities can be precalculated and stored in sample points. Using interpolation techniques like gouraud shading [1, 2], intensity calculation can be done for points lying between the samples. As many graphics accelerators provide gouraud shading of polygon meshes (SGI RE/Extreme, HP CRX) this has become the most popular technique for rendering radiosity scenes.

3. Models for Illumination

3.1. The Lambert Reflector

A lambert reflector is a surface which has a special bidirectional reflection density function (BRDF) that scatters incident light independent of the direction of the incoming ray. The micro structure of a gaussian surface, which is a lambert reflector is shown in Figure 1.

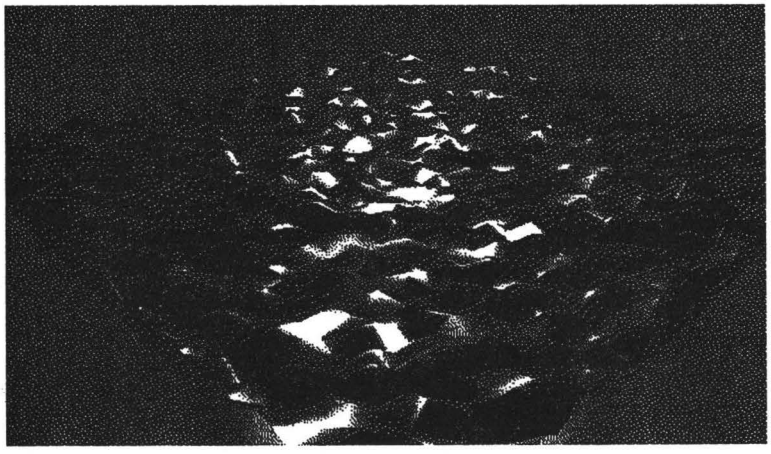


FIGURE 1. Lambert Reflector (gaussian surface)

A black body, a term often used in the physics of heat transfer [3, 4], is an example for such a reflector. The brightness at which it appears does not depend on the point of view. Thus the radiance L is equal for all observation angles.

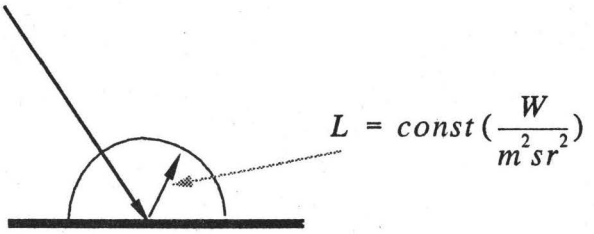


FIGURE 2. The Radiance of a Lambert Reflector

As the (radiant) intensity I depends on the projected area, it can be expressed as a function of the cosine angle between surface normal and reflected ray.

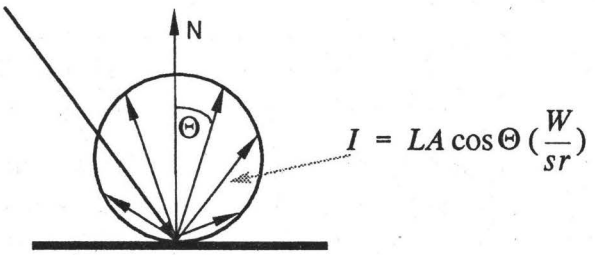


FIGURE 3. The Radiant Intensity (Lambert Circle)

3.2. The Solid Angle Ω

The solid angle between a surface patch A and a point P is equal to the area of the perspective projection of the patch on an unit-sphere, centered in P (Figure 4).

3.3. The Differential Solid Angle $d\omega$

The solid angle between a differential area dA and a point P is defined by:

$$d\omega = \frac{\cos\alpha dA}{r^2} = \frac{dA_P}{r^2}$$

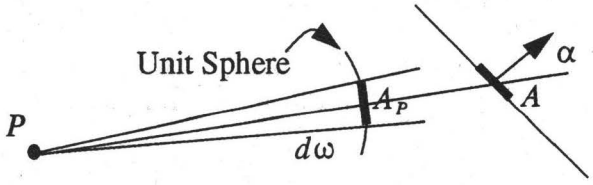


FIGURE 4. The Differential Solid Angle $d\omega$

The $\cos\alpha$ is used to project the differential area dA on a sphere. The solid angle Ω is obtained by integration of $d\omega$ over the area A. The following figure shows how $d\omega$ can be expressed in terms of the elevation angle Θ and the azimuth angle γ.

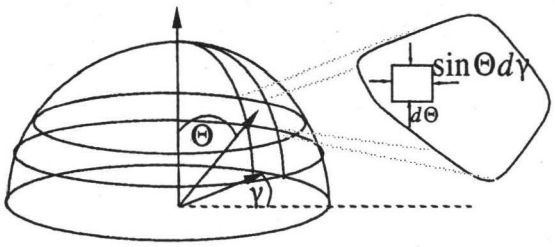


FIGURE 5. Definition of $d\omega$ with Polar Angles

An ideal lambert reflector does not exist because most surfaces show a specular behavior for incident angles of nearly 90°. An example of an almost diffuse reflector are walls painted with chalk. For diffuse surfaces form factors can be defined using the definition of the radiance L.

3.4. Form-Factors

3.4.1 Form-Factors Between two Differential Areas

The radiance L_1 of a diffuse area A_1 is independent of the elevation angle Θ_1.

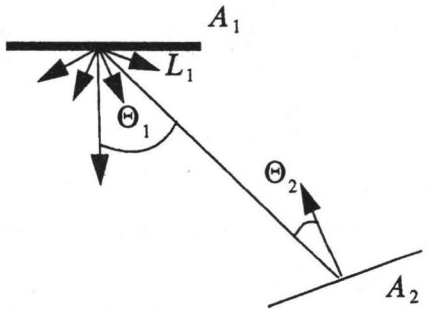


FIGURE 6. The Radiance of a Lambert Reflector

The form-factor $F_{dA1 \rightarrow dA2}$ between two patches is defined as the (total hemispherical) fraction of energy leaving patch 1 that arrives at patch 2 [21].

$$F_{dA1 \rightarrow dA2} = \frac{d^2\Phi_2}{d\Phi_1}$$

EQ. 1

In case of a diffuse source, the directional radiant flux Φ can be defined as follows:

$$d^2\Phi_1 = L_1 d\omega_1 \cos\Theta_1$$

EQ. 2

Via integration over the hemisphere the total emitted energy can be obtained from Equation 2:

$$d\Phi_1 = \int_0^{\pi}\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} L_1 d\omega_1 \cos\Theta_1 d\Theta_1 d\phi_1 = L_1 \int \sin\Theta_1 d(\sin\Theta_1) dA_1 = \pi L_1 dA_1$$

EQ. 3

The radiant flux $d\Phi_2$ reaching the differential area dA_2 is equal to the flux emitted from dA_1 in the direction of dA_2.

$$d^2\Phi_2 = L_1 d\omega_1 dA_1 \cos\Theta_1 = L_1 \frac{\cos\Theta_2 dA_2 \cos\Theta_1 dA_1}{r^2}$$

EQ. 4

By substitution of EQ. 3 and EQ. 2 in EQ. 1 the form-factor between two differential diffuse elements can be defined as follows:

$$F_{dA1 \to dA2} = \frac{\cos\Theta_2 dA_2 \cos\Theta_1}{\pi r^2}$$

EQ. 5

3.4.2 Form-Factors Between a Finite- and a Differential Element

All radiosity methods calculate the radiance of samples lying on the surfaces of objects. The radiance emitted by patch A_j to sample point P, lying on a differential element, is defined by

$$R_P = F_{j \to dP} R_j$$

EQ. 6

if the radiance R_j is assumed to be constant for patch j. If in addition patch j is a diffuse light emitter, the area integral $F_{j \to dP}$ can be expressed as a contour integral over the edge of the surface using Stoke's theorem. For polyhedral surfaces this contour integral can be written as a sum over the edges of the polygon.

$$F_{A1 \to dA2} = \frac{1}{2\pi} \sum_i \Gamma_i N_i$$

EQ. 7

N_i is the surface normal and Γ_i is the cross product between vectors to the corners of the polygon. Γ_i is scaled to a length equal to the angle γ_{i+1} between the vertex-vectors.

Equation 3 and 4 show that the calculation of the radiance of a sample can be reduced to the determination of form-factors. Several algorithms for form-factor calculation [12, 5] have been developed. This paper presents an alternative storage method for radiance values which is independent of the representation of the object. The algorithm works for scenes, which are composed of different types of primitives, like polygon meshes, solids or free-form surfaces and can also be used to store shadows of point-radiators.

3.5. Vertex Based Storage of Radiosities

Radiosity algorithms like the Progressive Refinement Method [12] subdivide the environment into a set of patches for which radiance is nearly constant (analog to FE-models). Cohen and

Greenberg [7] made a distinction between patches, which radiate energy and elements for which intensities are calculated.

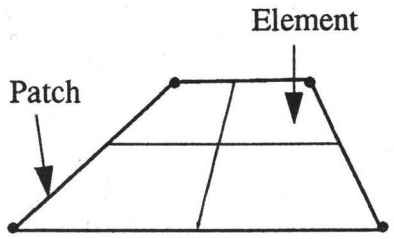


FIGURE 7. Patches and Elements

Elements are used to improve the rendering of detailed, sharp shadows. To save computation time during the energy distribution process (PRR-pass) elements are not considered in this step. Patch radiosities from the PRR-pass determine the vertex radiosities of the patches. The vertex radiosities derived for elements are used to calculate intensities in the rendering pass.

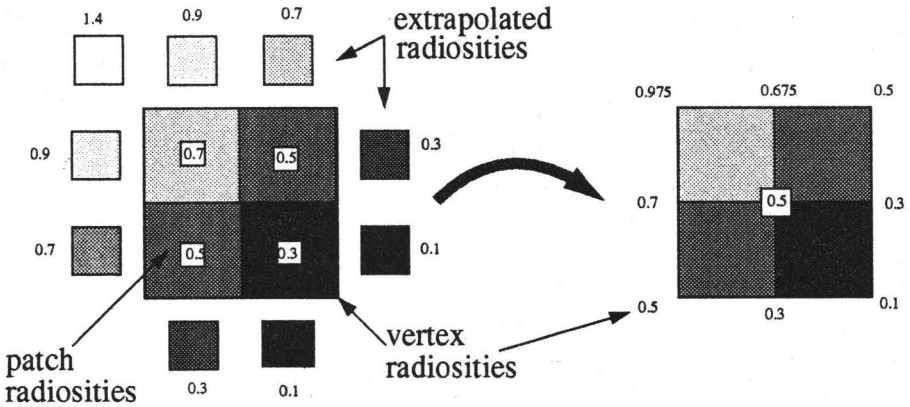


FIGURE 8. From Patch- to Vertex-Radiosities

A disadvantage of this storing method is that detailed shadows increase the size of the scene which reduces rendering speed. The central idea of this paper was to develop an algorithm which is able to store detailed shadows without changing the geometry of the scene.

4. Shadow Maps

The algorithm for preprocessing and storing the shadow maps has been implemented and tested for triangular objects. Therefore the file format has been extended to provide group information for each primitive. All primitives belonging to one group will be referred to as a face in the following text.

4.1. Preprocessing

4.1.1 Clustering

During a first preprocessing step adjacent faces of a scene which have nearly the same average normal are grouped together in clusters. In this implementation clustering is done by a programm named **htfclust**, written in C++. A brief description of the file format and options of this preprocessing tool can be found in appendix 7.1.

Example:

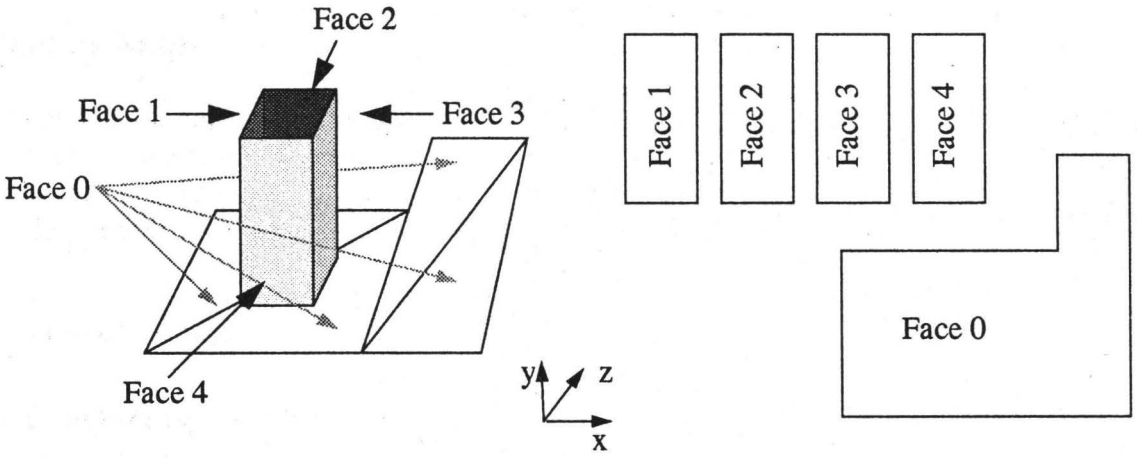


FIGURE 9. Scene with 5 Faces

Patches, having different materials, can be grouped into one face to keep the number of faces and thus the amount of memory needed at a minimum.

4.2. Map Generation

In a second preprocessing step a viewing transformation is defined for each face which was generated during the clustering. This transformation is used to project all samples lying on the corresponding face from 3D on a 2D map (shadow-, radiosity map). Therefore another C++ tool has been developed which reads faces stored in an htf-file* and produces a list of viewing trans-

*. A description of the htf-format which is used to store triangles and polygons (hpf) can be found in Appendix 7.1

formations, one for each face. An epsilon value controls the resolution of each map. A description of **htffac** is given in appendix 7.3.

4.3. Sampling

When a scene is rendered intensities have to be calculated for each visible surface point P. Each surface is composed of a set of facets, where every facet is associated with a specific face. The viewing transformation corresponding to that face is used to transform the (3D) sample P on a 2D map (P').

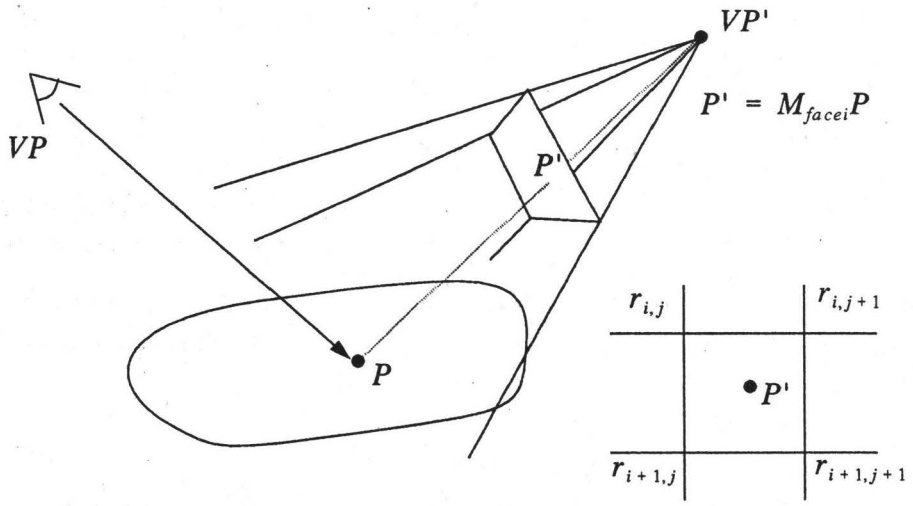


FIGURE 10. Projection Mapping

The intensities of the four neighboring points are used for a (bilinear) interpolation. As the interpolation is done in the projection plane many interpolation techniques which are based on meshes or scattered data can be applied.

If the intensity for a map vertex r_{ij} has to be evaluated a ray is cast from the view-point (VP') through r_{ij} into the scene, to get the intersection with the surface. The intensity of the sample lying on the surface is defined by the following sum over all light emitting patches A_i.

$$I_P = \delta \sum F_{Ai \rightarrow dP} R_i$$

EQ. 8

Several algorithms for calculating form-factors between a patch and a differential element can be used [5, 12]. To improve efficiency during the rendering pass a distributed client-server model has been developed.

4.4. A Distributed Rendering Model

A map server, which reads all map definitions provides a service on a reserved TCP port, which can be used by multiple rendering processes. This map server stores all maps of a scene and is connected with several clients which are used to calculate radiosities.

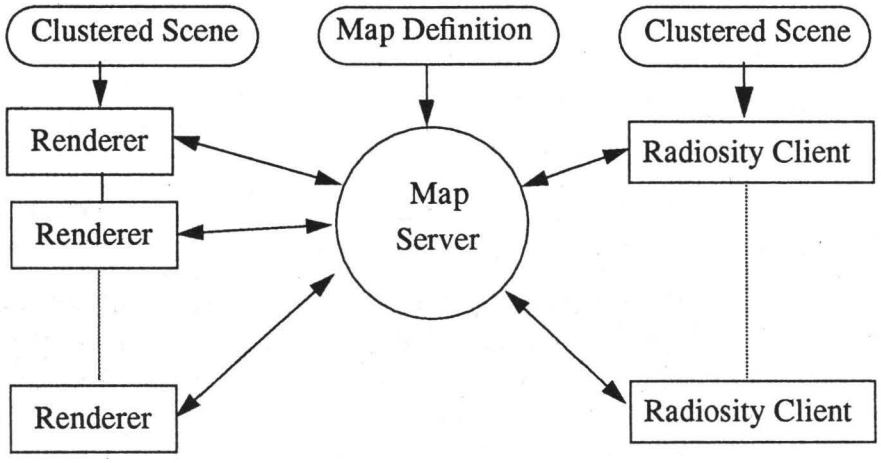


FIGURE 11. Distributed Rendering Model

For communication, stream and datagramm sockets have been used. Sample requests are sent from the renderers to the map server independent whether ray-tracing or any other type of hidden surface algorithm is used. The map-server builds a list of corner vertices $r_{k,l}$ needed to interpolate the sample. These vertices are propagated to the radiosity clients. The radiosities returned by these clients are stored in the associated maps of the server and the interpolated intensity is sent back to the renderer.

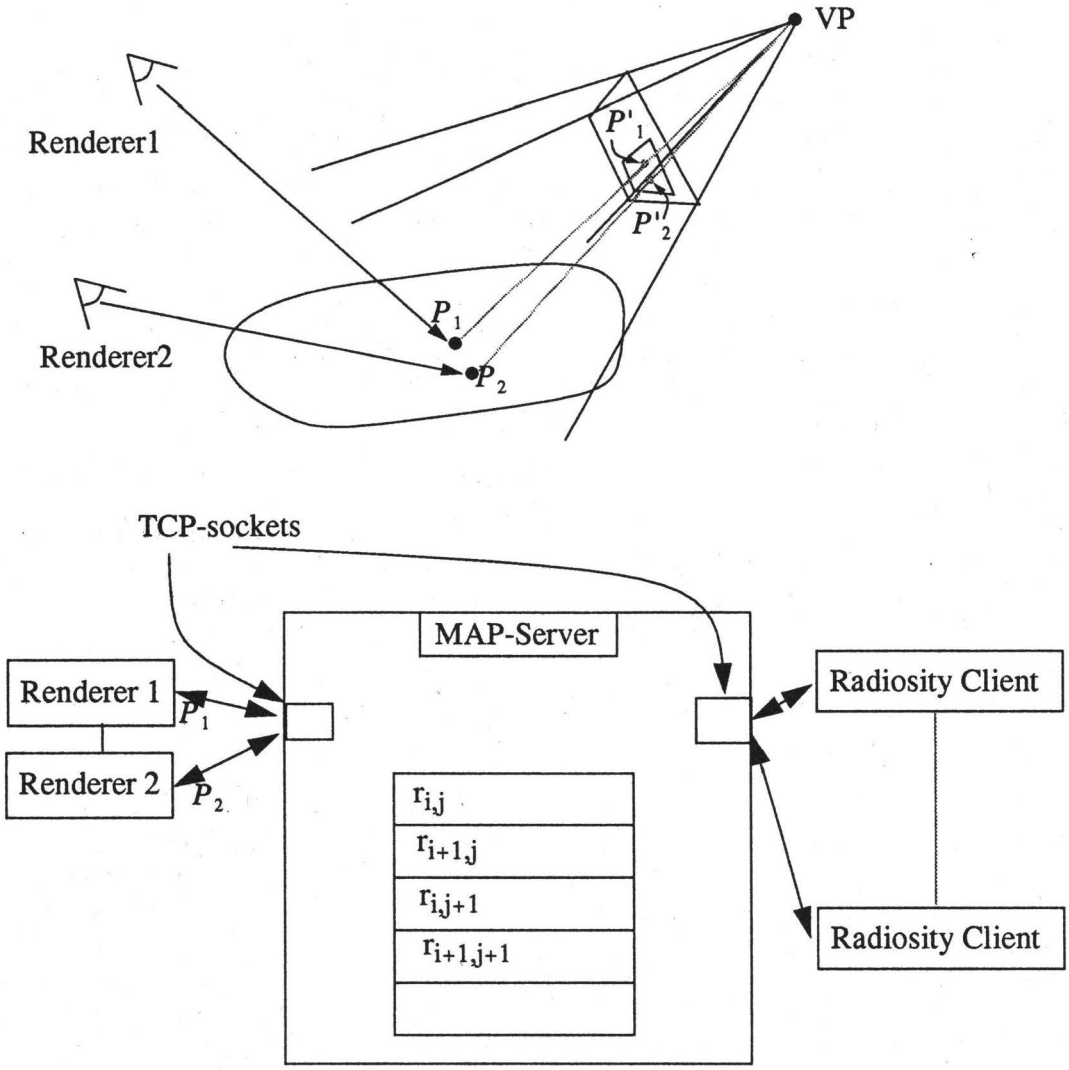


FIGURE 12. Example

The above example shows a situation where two renderers send sample requests to the map-server which need the same corner vertices for interpolation. In this case only four (instead of eight) samples are propagated to the radiosity clients.

5. Implementation

The shadow-mapping technique has been implemented as a C++ library. The basic structure is a single shadow-map, which stores information about the transformation, map-resolution and some additional data. The constructor defined for this class, is simply invoked with the map-resolution, view- and reference-point. The most important method of this class is a function used to get or set radiosity values defined by a 3D-point. Additional functions for loading and saving a map are also provided.

For sample generation and rendering a ray-tracer has been used. The radiosities are calculated from 3D-samples using the analytic form-factor technique described in [5] or alternatively Cohens hemi-cube [7].

6. Future Work

The current implementation uses a regular grid for storing the radiosities in a map, which can be memory-expensive. Instead of this regular grid an adaptive quadtree-like structure could be used, combined with summed area tables to eliminate aliasing caused by detailed shadows.

Other optical phenomena for example high frequency effects like caustics can be included. These effects are difficult to store in the vertices of polyhedral meshes as they are usually very small and have sharp contours.

Finally the entire energy distribution process could be done on shadow maps instead of patches and elements to reduce errors which result from a course meshing.

7. Appendix

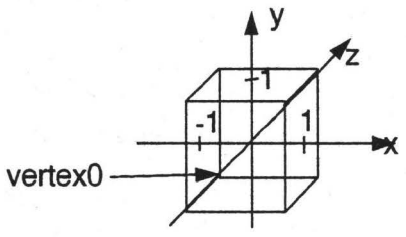
7.1. Hierachical File Format (HTF/HPF)

All tools including renderers, pre- and post-processors, and converters read and write polygons stored in ASCII -files called htf- or hpf-files. Htf-files are a subgroup of hpf-data which may have polygons with an arbitrary number of vertices. In addition to geometry a symbolic information about the material (RGB-) radiosities can be stored for each vertex.

The following htf-scene shows a cube build of 6 faces and 12 triangles where each face has a different material-index which ranges from 1 to 6.

```
8 12
-1.0 -1.0  1.0   0.0 0.0 0.0
-1.0  1.0  1.0   0.0 0.0 0.0
 1.0  1.0  1.0   0.0 0.0 0.0
 1.0 -1.0  1.0   0.0 0.0 0.0
-1.0 -1.0 -1.0   0.0 0.0 0.0
-1.0  1.0 -1.0   0.0 0.0 0.0
 1.0  1.0 -1.0   0.0 0.0 0.0
 1.0 -1.0 -1.0   0.0 0.0 0.0
2 1
 4 5 1
 4 1 0
2 2
 2 6 7
 2 7 3
2 3
 0 3 7
 0 7 4
2 4
 0 2 3
 0 1 2
2 5
 7 6 5
 7 5 4
2 6
 2 1 5
 2 5 6
```

number of points
number of polygons (triangles)
XYZ-vertex-coordinates (0..number of points-1)
RGB-vertex radiosities
number of polygons belonging to that face
symbolic link to material list
list of vertex indices of a polygon



7.2. Preprocessor: htfclust

NAME:

htfclust - A clustering preprocessor for htf-files

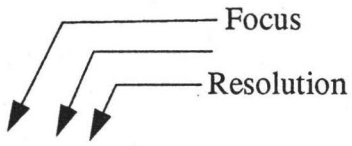
SYNOPSIS:

htfclust <htf_input_file.htf> [<htf_output_file.htf>] [-w<angle>] [-e<epsilon>]

DESCRIPTION:

Htfclust reads and writes to <htf_input_file.htf> if no <htf_output_file.htf> is specified. Groups of triangles are given the same face index, if their normals differ less then <angle> degrees and have common vertices. The vertices do not need to have the same index. They are searched for within an <epsilon> distance.

SEE ALSO:



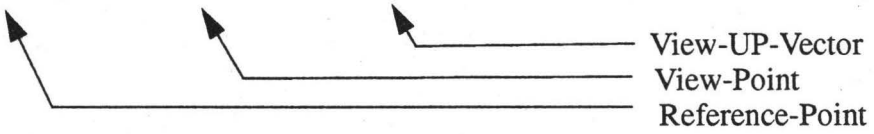
DIAGNOSTICS:

BUGS:

None

NOTES:

The current implementation is limited on triangles rather than on polygons.



7.3. Preprocessor: htffac

NAME:

htffac - A face preprocessor for htf-files

SYNOPSIS:

htffac <htf_file.htf> [-e<epsilon>]

DESCRIPTION:

Htffac reads <htf_file.htf> and generates a view for each face. The output is written to a file named <htf_file.fac> in a line orientated ASCII-format. The value for <epsilon> determines the resolution of the maps generated. The output of htffac for a cube centered in the origin and aligned with the coordinate system would be:

Example: htffac cube.htf

0	-1.0 0.0 0.0	-3.0 0.0 0.0	0.0 1.0 0.0	0.95 1 2 2
1	1.0 0.0 0.0	3.0 0.0 0.0	0.0 1.0 -0.0	0.95 1 2 2
2	0.0 -1.0 0.0	0.0 -3.0 0.0	0.0 -0.0 -1.0	0.95 1 2 2
3	0.0 0.0 1.0	0.0 0.0 3.0	0.0 1.0 0.0	0.95 1 2 2
4	0.0 0.0 -1.0	0.0 0.0 -3.0	-0.0 1.0 0.0	0.95 1 2 2
5	0.0 1.0 0.0	0.0 3.0 0.0	-0.0 0.0 1.0	0.95 1 2 2

SEE ALSO:

DIAGNOSTICS:

BUGS:

None

NOTES:

The current implementation is limited on triangles rather than on polygons.

7.4. Form-Factor Between a Polygon and a Differential Area Receiver

The form-factor between a polygon which is a diffuse emitter and an unoccluded (diffuse) differential area receiver can be evaluated by a sum-formula over the vertices of the polygon [6]. This paragraph gives a derivation of this formula starting from the elementary thermophysical definition (EQ. 9) of a form-factor between two differential patches derived in paragraph 3.4.1.

$$F_{dA_1 \to dA_2} = \frac{\cos\Theta_1 \cos\Theta_2}{\pi r^2} dA_2$$

EQ. 9

Integration over the area of the light emitting patch A_1 which is a lambert source leads to the definition of $F_{A1 \to dA2}$:

$$F_{A_1 \to dA_2} = \int_{A_1} \frac{\cos\Theta_1 \cos\Theta_2}{\pi r^2} dA_2$$

EQ. 10

Equation 10 can be rewritten as follows:

$$F_{A_1 \to dA_2} = \int_{A_1} [\,(x_1 - x_2) f \cos\alpha_2 + (y_1 - y_2) f \cos\beta_2 + (z_1 - z_2) f \cos\gamma_1\,]\, dA_2$$

$$f = \frac{(x_2 - x_1) + (y_2 - y_1) + (z_2 - z_1)}{\pi r^4}$$

EQ. 11

Figure 13 shows this configuration with two differential areas where A_2 is the energy receiving element.

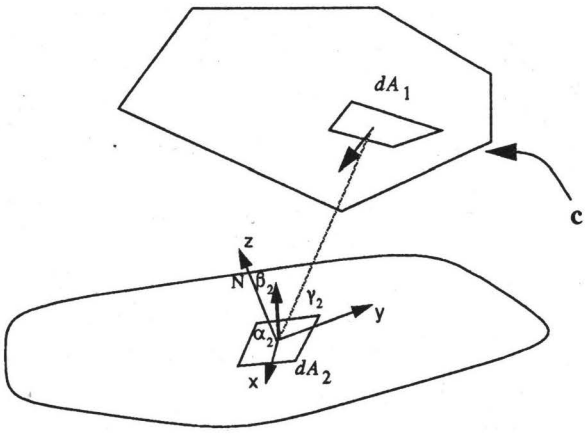


FIGURE 13. Two Differential Element Configuration

Stoke's theorem in three dimensions can be used to transform the area integral in a contour integral along the boundary curve **c**, which are linear segments in case of a polygon.

$$\int_{A_1}\left[\left(\frac{dR}{dy}-\frac{dQ}{dz}\right)\cos\alpha_2+\left(\frac{dP}{dz}-\frac{dR}{dx}\right)\cos\beta_2+\left(\frac{dQ}{dx}-\frac{dP}{dy}\right)\cos\gamma_2\right]=\oint_C(Pdx+Qdy+Rdz)$$

(**Stoke's theorem**) EQ. 12

Sparrow [6] proposes the following substitutions for P, Q and R.

$$P=\frac{-(z_2-z_1)+(y_2-y_1)}{2\pi r^2}$$

$$Q=\frac{(z_2-z_1)-(x_2-x_1)}{2\pi r^2}$$

$$R=\frac{-(y_2-y_1)+(x_2-x_1)}{2\pi r^2}$$

EQ. 13

Solutions for P, Q and R have to be twice differentiable and fulfill the following relation:

$$(\frac{dR}{dy} - \frac{dQ}{dz}) = (x_1 - x_2)f; (\frac{dP}{dz} - \frac{dR}{dx}) = (y_1 - y_2)f; (\frac{dQ}{dx} - \frac{dP}{dy}) = (z_1 - z_2)f$$

EQ. 14

Using Equation 12 the form factor $F_{A1 \to dA2}$ can be rewritten as:

$$F_{A1 \to dA2} = \cos\alpha_2 \oint_c \frac{z_1 dy_1 - y_1 dz_1}{2\pi r^2} + \cos\beta_2 \oint_c \frac{x_1 dz_1 - z_1 dx_1}{2\pi r^2} + \cos\gamma_2 \oint_c \frac{x_1 dy_1 - y_1 dx_1}{2\pi r^2}$$

EQ. 15

If A_1 is defined by a polygon each edge can be considered separately, and the contour c is split into n linear pieces, which are convex combinations of the polygon vertices P^i.

$$F_{A1 \to dA2} = \sum_{i=1}^{n} F^i_{A1 \to dA2}$$

EQ. 16

Index i indicates that the integration of $F^i_{A_1 \to dA_2}$ is done along edge i which is defined by P^i and ΔP^i.

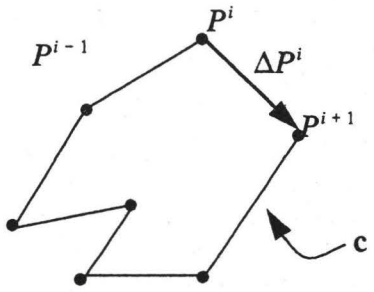


FIGURE 14. Contour Integration for a Polygon

Substituting the values for P^i, and ΔP^i in equation 15 the form-factor from an edge of a polygon to a differential area is:

$$F^i_{A_1 \to dA_2} = \frac{1}{2\pi}\oint_0^1 \begin{bmatrix}\cos\alpha_2\\ \cos\beta_2\\ \cos\gamma_2\end{bmatrix}\begin{bmatrix}z_1dy_1 - y_1dz_1\\ x_1dz_1 - z_1dx_1\\ x_1dy_1 - y_1dx_1\end{bmatrix}\frac{1}{r^2}dt = \frac{1}{2\pi}\oint_0^1 \begin{bmatrix}\cos\alpha_2\\ \cos\beta_2\\ \cos\gamma_2\end{bmatrix}\frac{[P^i \times \Delta P^i]}{|P^i + \Delta P^i|^2}dt$$

EQ. 17

The angles α_2, β_2 and γ_2 are the direction angles of the normal N.After substituting the values u, v and w integrating the form-factor of an edge can be rewritten as ($u = \langle P^i, P^i\rangle$ $v = 2\langle P^i, \Delta P^i\rangle$, $w = \langle \Delta P^i, \Delta P^i\rangle$):

$$F^i_{A_1 \to dA_2} = \frac{N \cdot CR^i}{\pi\sqrt{4uw - v^2}}\left[\operatorname{atan}\frac{2u + v}{\sqrt{4uw - v^2}} - \operatorname{atan}\frac{v}{\sqrt{4uw - v^2}}\right]$$

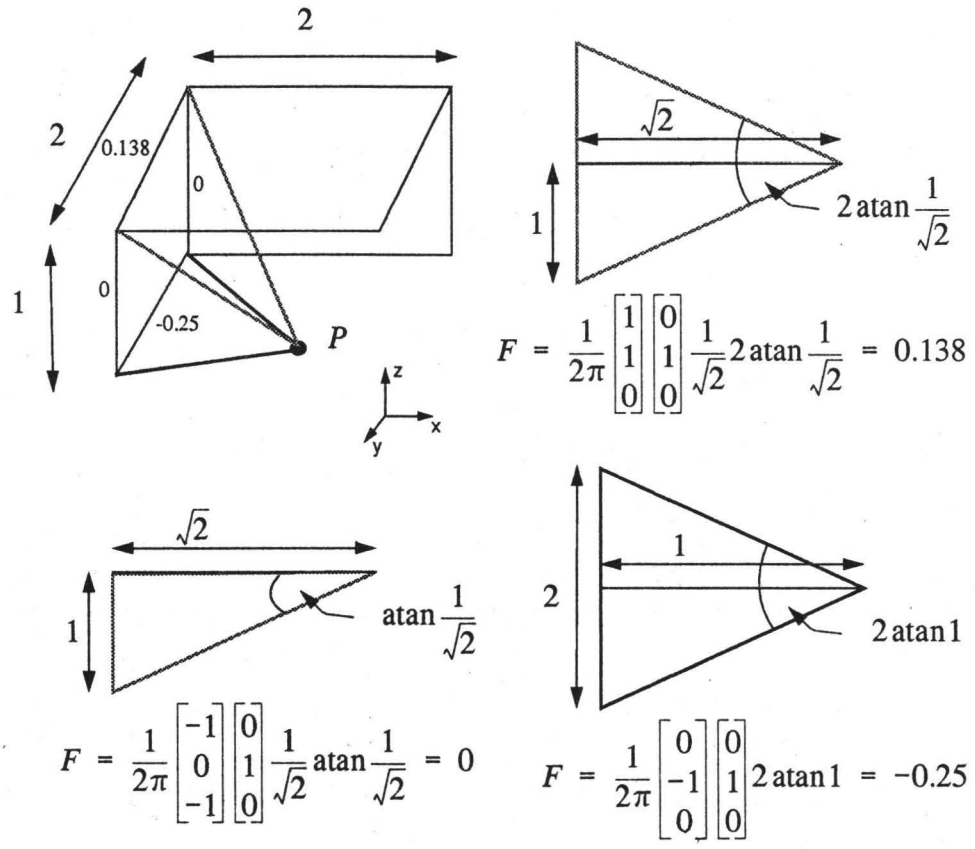
EQ. 18

In Equation 18 and 19 the cross-product between two vertex vectors P^i and P^{i+1} is abbreviated with CR^i.

$$F^i_{A_1 \to dA_2} = \frac{N \cdot CR^i}{2\pi|CR^i|}\operatorname{acos}\frac{\langle P^i, P^{i+1}\rangle}{|P^i||P^{i+1}|}$$

EQ. 19

Equation 19 in combination with Equation 16 is used to define the final sum formula (EQ. 7) which is an easy and efficient way to calculate unoccluded form-factors for arbitrary polygons. Figure 15 shows how this formula can be used to evaluate the form-factors needed for a hemicube.



$F_{top} = 0.552$ $F_{side} = 0.112$ $F_{total} = 1$

FIGURE 15. Form-Factor of a Hemi-Cube

8. Reference

[1] H.Gouraud: *Continous shading of curved surfaces*, IEEE Transactions on Computers, C-20, pp 623-629, June 1971

[2] P. Jacob: An Improved Shading Algorithm For Radiosity Based Renderers, Focus on Scientific Visualization,Springer-Verlag 1993,pp 317-324

[3] R.Siegel, J.Howell: *Thermal Radiation Heat Transfer*, Hemisphere Publishing Corp. Washington DC 1981

[4] Hott, Sarofim: *Radiative Transfer*, Mc Graw-Hill, New York, 1982

[5] P.Jacob: Analytical Form-Factor Calculation using BSP-Trees, Conference Proceedings, Graphicon'93, Saint-Petersburg, Russia

[6] E.M. Sparrow: *A New and Simpler Formulation for Radiative Angle Factors, Journal of Heat Transfer*, Vol 85 no. 2 pp 81-88, 1963

[7] M.Cohen, D.Greenberg: *The Hemi-Cube, A Radiosity Solution For Complex Environments,* ACM Computergraphics Vol.19, Number 3, pp 31-40

[8] R.Baum,H.Rushmeier & J.Wingert: *Improving Radiosity Solutions Through the Use of Analytically Determined Form-Factors*, ACM Computergraphics Vol. 23. Number 3, pp325-334

[9] James T.Kajiya: *The Rendering Equation,* ACM Computergraphics, Vol. 20, Number 4, pp.143-150

[10] F.Crow: *Shadow Algorithms for Computer Graphics*, ACM Computergraphics Vol. 11 Number 3, pp.242-248

[11] J.Wallace, K.Elmquist, E.Haines: *A Raytracing Algorithm for Progressive Radiosity*, ACM Computergraphics Vol. 23 Number 3, pp.315-324

[12] Cohen,Michael F, S.Chen, J.Wallace, D.Greenberg: *A Progressive Refinement Approach To Fast Radiosity Image Generation*, ACM Computergraphics Vol. 33 Number 3 pp.75-84

[13] Urs Meyer, Hemi-Cube Ray Tracing: *A Method for Generating Soft Shadows*, Proc. Eurographics 1990, pp.365-376

[14] Fuchs, Kendem, Naylor: *On Visible Surface Generation by A Priori Tree Structures*, ACM Computergraphics Vol. 14, Number 3, pp 124-133

[15] N.Chin, S. Feiner: *Near Real-Time Shadow Generation Using BSP-Trees*, ACM Computergraphics Vol. 23, Number 3, pp.99-106

[16] Thibault, Naylor: Set Operations on Polyherda Using Binary Space Partitioning Trees, ACM Computergraphics Vol. 21, Number 4, pp 153-162

[17] A.Woo,P.Poulin,A.Fournier: *Survey of Shadow Algorithms*, IEEE Computer Graphics, Nov. 1990

[18] Goral,Cindy,Torrance,Kenneth,Greenberg,Donald,Battaile,Bennet: *Modeling The Interaction of Light Between Diffuse Surfaces,* ACM Computergraphics (SIGGRAPH'84) pp.213-222

[19] ‡ J.R Howell: *A Catalog of Radiation Configuration Factors*, MC Graw-Hill, New-York,1982, B-98

[20] W. Nusselt: *Graphische Bestimmung des Winkelverhältnisses bei der Wärmestrahlung,* VDI Z. Vol 72 , p.673 ,1928

[21] S.Westin,J.Arvo,K.Torrance:*Predicting Reflectance Functions from Complex Surfaces*,ACM Computergraphics (SIGGRAPH'92) Vol.26 pp.255-264