Rendering Inhomogeneous Surfaces with Radiosity

L. Mostefaoui, J.M. Dischler and D. Ghazanfarpour
Laboratoire MSI
E.N.S.I.L. – Université de Limoges
TECHNOPOLE 87068 Limoges Cedex- France

Abstract. Natural surfaces are often complex: they nearly always exhibit small scale imperfections such as dirt, dust, cracks, etc., as well as large scale structural elements, as for wickerwork, brick walls, textiles, pebbles, etc., that are generally too complex to be modeled explicitly. In this paper, we propose a new multi-scale periodic texture model adapted to the efficient simulation of the previously mentioned features. This new model combines notions of virtual ray tracing (that we have recently introduced) with bi-directional texture function, while it also considers self-shadowing and inter-reflections at texture scale. In a second step, the texture model is integrated into hierarchical radiosity with clustering. Therefore, an extension of radiosity techniques, currently limited to texture maps, bump maps and general (homogeneous) reflectance functions, is proposed. The final rendering consists of applying a second ray tracing pass, based on a gathering methodology adapted to the model. The method provides images at a significant lower computation and memory consumption cost than with ”explicit” models in the case of periodic features (wickerwork, grids, pavements, etc.) for a similar visual quality.

1 Introduction

Most natural surfaces such as streets, textiles, walls, floors, etc., are not uniform, neither concerning the local geometry (wrinkles, cracks, etc.), nor concerning the reflectance behavior: color variation, and more generally non constant bi-directional reflectance distribution function (BRDF). Various texturing techniques such as texture mapping, ”Bump mapping” Blinn [1], solid texturing [2, 3], displacement mapping [4], hypertexturing [5], texel mapping [6], etc. have been developed in computer graphics to simulate such complex effects. All of these techniques help to enrich the visual quality of the scenes, without increasing the geometric complexity, e.g. the number of patches.

In spite of their importance, only a low number of radiosity approaches have addressed the problem of integrating textures, or more generally ”inhomogeneous” surfaces. In [7], a first method was proposed for texture maps by considering an average color on the patches. [8] proposed an efficient hierarchical technique, based on a Galerkin radiosity. The problem of including bump maps into radiosity simulations was addressed in [9]. Also general reflectance behaviors (general BRDFs) were considered in [10]. However, this model does not consider inhomogeneous cases, that is, a BRDF variation over the patches. Likewise, we recently developed a method allowing for procedural features (displacement maps, hypertextures, fractals, etc.) to be integrated into radiosity simulations without using intermediate ”high precision” meshes [14]. But, the formulation is not hierarchical. In addition ”micro-scale” inter-reflection phenomena are not considered (inter-reflections at texture scale). Currently, complex features are integrated into radiosity simulations using a high rate subdivision into small triangles. The resulting geometric complexity is often enormous and overloads the memory and computation capacity (in spite of clustering strategies [11, 12]). In fact, finite elements techniques still suffer from being limited to scenes of rather poor visual complexity, compared to ray tracing (or more generally path tracing) rendered scenes that already reach impressive performances [13].

The aim of this paper is to propose a texture model adapted to the efficient realistic rendering of the previously mentioned complex surface structures, and to integrate this model into hierarchical radiosity simulations with clustering. Our texture model is adapted to the rendering of periodic features (as for texel maps) such as for example wickerwork. It is based on the notion of bi-directional texture function (BTF), that was introduced in [15], and combines it with our virtual ray tracing technique [19] (see next section).

Once defined, the texture model is integrated into hierarchical radiosity simulations with clustering by generalizing the approaches of [20, 21] to the consideration of non-constant BRDFs over patches. We assume that the entire energy received by a piece of patch (covering a piece of the texture) is scattered towards all directions according to an average BRDF corresponding to that piece of texture. As opposed to radiosity for general reflectance functions [10], we consider different BRDFs at different levels of the patch hierarchy. In practice, we also used a discrete decomposition of the hemisphere into bins, as opposed to spherical harmonics.

Final pictures are computed using a gathering step based on ray tracing, which is necessary because of the inhomogeneous nature of the patches. Since the final gathering step is usually time consuming, we adapted the methodology to the use of bilinear interpolations, which is similar to techniques developed in [22, 23]. However, we also estimate the variance to avoid excessively smoothing shadows and discontinuities.

In the next section, we first present our multi-scale texture model. In section 3, this model is integrated into radiosity simulations. Then, we show how texture-scale light scattering effects can be pre-computed on samples, in particular inter-reflections at texture scale. In section 5, we explain the final rendering technique, based on ray tracing. Finally, in the last section, we show some graphical results, as well as a performance study.

2 Modeling inhomogeneous surfaces

The texture principle presented [15], consists of using a map (like a usual texture map), that, instead of containing a color information on each pixel, contains a ”full” BRDF. Such a map is called a bi-directional texture function. Because of the very important storage requirements of BRDF MIP-maps [16] (a thin 128^2 MIP-map for example requires storing 21845 BRDFs), we proposed in [19] a BTF model limited to the consideration of some finite sets of materials only. The formulation is reduced to a view dependent multi-resolution map of normals, colors, transparencies, etc.

Generally, there are two problems common to nearly all approaches based on discrete multi-resolution samples [17, 18, 19]: memory consumption and the fact that the sampling may become visible in some cases (for example the pixels of a low resolution texture map). For this reason, it becomes interesting in some cases to keep a continuous description of the considered domain. A continuous description avoids the problem of visible sampling and excessive memory consumption, while aliasing can be partly prevented using oversampling.

In this paper, we follow the suggestion of [17] by preserving both levels of the texture description: a continuous one (rendered directly using the virtual ray tracing technique developed in [19]), and a sampled one (a BRDF MIP-map) for efficient antialiasing in extreme cases. The height of the MIP-map pyramid determines the memory consumption / computational efficiency ratio, while the continuous description allows us to avoid the ”pixel” effects. Figure 1 illustrates the principle of combining BRDF MIP- maps with virtual ray tracing. A periodic texture is first modeled by the user using a usual geometric modeler, such that the continuity, when repeating the sample, is preserved (as for the wickerwork model in figure 1). This geometry can be mapped directly onto a 3D surface S using a kind of ”secondary” ray tracing technique that we called ”virtual ray tracing” [19].

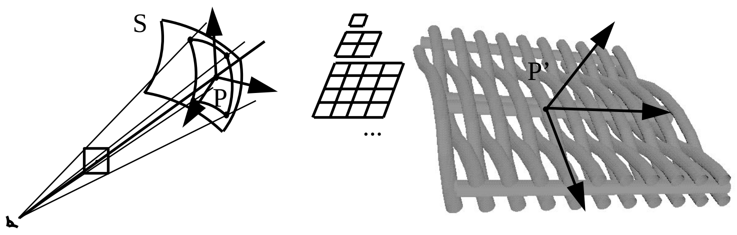

Fig. 1. Combining virtual ray tracing with BRDF MIP-maps (BTFs).

A computer simulation (as it has been done in [25], for example) can be used to compute a BRDF corresponding to the previously modeled texture. In our case, we directly use the radiosity technique developed in the next section. The left part of figure 2 illustrates the BRDF that we obtained for the wickerwork of figure 1. In the same way, it is also possible to compute BRDFs corresponding to sub-parts of the texture, and thus to build a complete BRDF MIP-map pyramid $\Gamma(l, u, v, \lambda)$. l represents the level in the hierarchy, $1 \leq u, v \leq 2^{l-1}$, and λ the wavelength. Figure 1 shows a pyramid of height three. Furthermore, transmittance can be considered (as shown in the middle part of figure 2), by constructing a BTDF (bi-directional transmittance distribution function) MIP-map. We also consider an ”opacity” coefficient, which accounts for ”holes” in the texture (as for a grid). Therefore, we compute a ”transparency” coefficient representing the portion of energy that goes through the texture without any changes (as opposed to the BTDF). The right part of figure 2 shows the transparency (the ratio of holes) for the wickerwork. This coefficient is only view dependent as opposed to the bi-directional nature of the BRDF and BTDF. Though it is possible to include the transparency directly as a part of the transmittance (BTDF), we separated both. The reason is that both can be handled differently during the radiosity simulation, as will be explained in the next section.

We now dispose of two texture descriptions: a continuous geometric model, and a discrete pyramid. Both can be combined during rendering as follows. Oversampling is first applied. Therefore, each pixel is sampled into a certain number of sub-pixels. For each of these sub-pixels, we determine a compression rate to find out the ”appropriate” level into the MIP-map (recent investigations have been made on determining that level [24] efficiently). If the level does not exist (below the bottom of the pyramid), we use for the shading computation the virtual ray tracing technique, which avoids ”pixel” effects. For all of our examples (see results), we used at most pyramids of height two or three (as shown in figure 1).

Unfortunately, the previously described combination of virtual ray tracing with

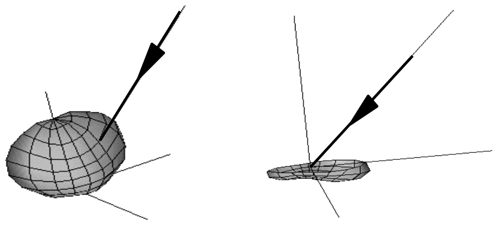

Fig. 2. BRDF, BTDF and view dependent ”transparency” obtained for the wickerwork sample.

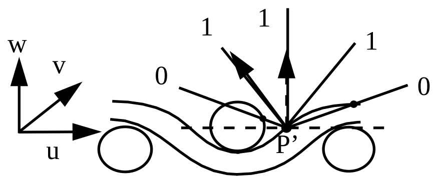

Fig. 3. A self-shadowing directional map.

BRDF MIP-maps might introduce a certain discontinuity at those regions of the textured surface S that are precisely at mid-way between the virtual ray tracing level and the bottom level of the BRDF pyramid. The reason is that the BRDF accounts for all ”internal” light reflection phenomena, as opposed to the virtual ray tracing technique. These phenomena are mainly: inter- reflections at texture scale and self-shadowing. Some parts of the texture may emit energy directly towards neighboring parts, or on the contrary may cast some shadows. The problem of inter-reflections at texture scale will be discussed in section 4. The problem of self-shadowing can be resolved as for bump mapping [29] within the virtual ray tracing computation, by shooting secondary rays in different directions. In fact, it is possible to build a directional shadow map on the texture point P' (see figure 3) to obtain a hemisphere of Boolean values (or a sphere in the case of transparent textures), that can be used as indicator during the shading computation. Results concerning self-shadowing are given in section 4, in parallel with examples concerning inter-reflections.

3 Hierarchical radiosity with clustering

Once the texture has been modeled, it needs to be integrated into realistic rendering and lighting simulation techniques. Radiosity allows us to determine the global lighting of the scene using as subdivision into finite elements. We will integrate our textures into hierarchical radiosity simulations with clustering, including general reflectance functions [20, 21]. However, we do not consider ”mirror-like” specular to diffuse transfers, but only directional diffuse transfers, mainly because of simplicity.

We will consider four kinds of energy transfers: (1) direct transfers, (2) reflected indirect transfers, (3) transmitted indirect transfers, and finally (4) direct transfers going through patches that have holes (in fact, (1) and (4) represent the same type of transfer; the only difference is the way that the visibility is processed). Note that in the case of transmittance, it becomes necessary to consider double-sided patches, which also doubles the radiance information to be stored. Often, the transmittance can be neglected (for example for a grid), the predominance being mainly ”transparency” (the proportion of holes).

The different energy exchanges taking place in a scene can be described as an integral equation, generally called in computer graphics the ”rendering equation” [26]. In the case of scenes, composed of patches, we obtain:

$$L^r_{out} = L_e + \sum_{i=1}^{Np} \int_{A_i} L_{in} \rho_r(\vec{d_{in}}, \vec{d_{out}}, u, v) \frac{\cos(\theta)\cos(\theta')}{r^2} V dA_i$$

where L^r_{out} represents the outgoing radiance, Np the number of patches, ρ_r the BRDF and V the visibility. The same applies for the transmittance with ρ_t, the BTDF. Note that for including the textures, we need to consider an ”inhomogeneous” structure on the patches. Therefore, the BRDF ρ_r, as well as the BTDF ρ_t, were both made (u, v) dependent in our equations. Various radiosity techniques have been developed to resolve the radiance equation presented above. We will combine the method of [10], developed for general reflectance functions, with the clustering strategy of Smits et al. [12]. It turns out that the transcription is straightforward.

Recall that in the previous section, we have computed a BRDF MIP-map $\Gamma(l, u, v, \lambda)$ (respectively a BTDF map) corresponding to the texture world. These BRDFs (respectively BTDFs) describe the mean reflectance (respectively transmittance) properties of the texture at different scales. As for [10], we need to store a directional energy emission information (an energy hemisphere, or two in the case of transmittance) on each patch (and sub-patch if introduced during the hierarchical radiosity). For practical reasons, we used a discrete decomposition of the hemisphere into bins, as opposed to spherical harmonics. We shall assume that one initial patch matches the size of the modeled texture sample (figure 4.a); that is, the mapping makes the vertices of the patch match the four corners of the texture sample. Thus, the global mean reflectance property of the patch is entirely captured by the corresponding top level of the texture BRDF pyramid, provided that the distortion introduced by the mapping is not too important. Also, the reflectance behavior of eventually introduced sub-patches, is captured by the corresponding lower levels in the pyramid. We note that for triangular patches, that generally cover triangular parts of the texture map, it is also possible to build BRDF pyramids matching these triangular parts. In the remainder of this paper, we mainly consider quadrilateral patches. Now, the energy transfer from one source patch P_i to another receiver patch P_j (figure 4.b), is given by:

$$L_{P_j} = L_{e,P_j} + \int_{P_i} L_{P_i} \Gamma_{P_j} \frac{\cos(\theta)\cos(\theta')}{r^2} V dP_i$$

where L_{P_j} represents the outgoing radiance. As for [10], we may use a disk approximation by sampling the emitter into N sub-elements. Normally, due to the inhomogeneous nature of the patches, each element emits the energy in a different way, which has been discussed in [27] in the case of general light sources. Storing on each patch multiple elements, where each of these elements contains its own energy hemisphere is unfortunately extremely memory consuming. Therefore, we actually stored on the emitter patch (and all of its sub-patches) only the global mean (directional) outgoing energy, plus for patches that have no sub-patches at most four elements (four hemispheres) to get more precise results. Since, sub-patches contain their own energy emission hemispheres, it is not necessary to store additional explicit elements. For example, the emitter P_i of figure 4.b contains four sub-patches, and one of these sub-patches contains again four. If we approximate the energy transfer from P_i to P_j using four disks,

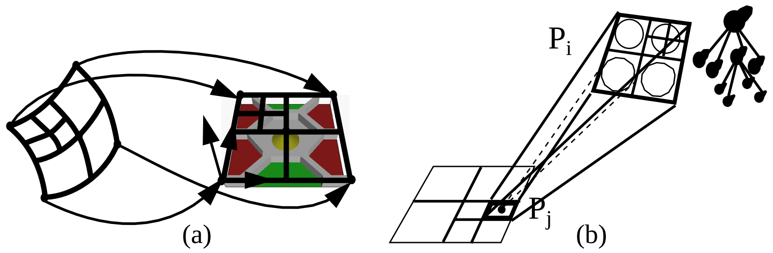

Fig. 4. (a) The vertices of the patch must match the corners of the texture map. (b) extended energy transfer, considering different BRDFs for inhomogeneous surfaces.

then we will consider the energy hemispheres of the four sub-patches, instead of that of P_i. Note also that it is important to use the appropriate BRDF level of the pyramid on the receiver P_j according to its depth in the hierarchy, since the BRDF is not constant across P_j (in the formula we called Γ_{P_j}=$\Gamma_{l,u,v,\lambda}$, where l, u, v are chosen according to the position of P_j in the hierarchy). We finally obtain following approximate transfer formula:

$$L_{P_j} = L_{e,P_j} + \sum_{k=1}^{N} L_{P_i}^k \Gamma_{P_j} \frac{\cos(\theta_k)\cos(\theta'_k)\pi\delta P_i^k}{\delta P_i^k + \pi r^2} V_k$$

Once the transfer is completed, the energy is pushed / pulled inside the patch hierarchy by simple addition of the energy hemispheres (downwards, the energy is split according to the respective areas of the sub-patches). Apart from the transfer between two patches, there is no real difference with usual hierarchical radiosity. The same error bounds and ”oracles” can be used to decide whether to subdivide or not. Also clustering can be processed identically. We used a technique based on α-links (as described in [12]) using a BFA criterion (radiosity, form factor and area). Before shooting the energy from one cluster towards an other cluster or towards a certain patch, the energies of all included patches (the leafs of the cluster) are summed up according to an internal visibility, that is computed using a software Z-buffer technique. Inversely, all energy received by a certain cluster is directly distributed among the enclosed patches. This process allows us to avoid storing a certain spherical energy information on each clusters. For the internal visibility, it is just important to consider the eventual low opacity of some patches. We used a software Z-buffer technique, that first projects all of the opaque patches, and then the transparent ones (with similarity to alpha maps). If transmittance is considered, patches are double-sided (they contain two hemispheres), thus, it is important to take into account the particular orientation of a certain patch.

4 Inter-reflections at texture scale

As we have mentioned in section 2, some parts of the geometric features of the texture might emit some energy directly towards neighboring features. For rather ”flat” textures, this term can be completely neglected, as opposed to rough textures. Computing an inter-reflection term for textures, actually has some similarity with the estimation of irradiance values within complex geometric structures. Often statistical approaches are used. In our case, the textures are periodic, and repeated over the surfaces. This opens

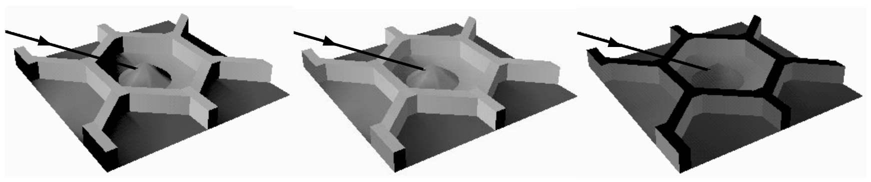

Fig. 5. An example of inter-reflection at texture scale.

the way to a simpler deterministic approach. In fact, we shall simplify the model by considering only the secondary indirect illumination (obtained after one bounce only). Let A be the radiance leaving a point P' of a patch i of the texture world, due to the secondary reflections occurring in the texture, when the latter is illuminated by a certain radiance L. A can be linearly expressed as $A = LF_i(\alpha, \beta)$, where F is a function depending only on the incoming directions (α, β) of the incident radiance L. F is also assumed to be view ”independent”, which excludes high specularity, and relates it to the irradiance.

The function $F_i(\alpha, \beta)$ (one for each patch of the texture) can be directly computed ”on the fly” during the BRDF pyramid construction technique that we have presented in section 2, since we basically apply a radiosity simulation, that can be used to compute also the secondary reflections.

Figure 5 illustrates an example of texture. From left to right, the figures illustrate: first, the initial energy, due to the direct illumination only, second, the result after considering secondary reflections , and last (on the right) the difference between the two previous results, e.g. the total secondary indirect illumination. We obtained this result for an illumination due to a normalized incoming radiance (as shown by the arrow). Since L has been normalized, the result of the right figure can be used directly as value for the function F_i for this direction. Applying the same technique for all directions, we obtain for each patch i of the texture world, the corresponding function $F_i(\alpha, \beta)$.

We note that it is important to duplicate the texture for the computation of F_i, otherwise there would be visual artifacts, especially on the patches at the ”frontiers” of the texture (they don’t have neighbors if the texture is not repeated).

The left part of figure 6 illustrates some tori, rendered using the virtual ray tracing technique (inter-reflections are already implicitly considered when the BRDF pyramid is used, as opposed to virtual ray tracing). The left picture shows the result using no inter-reflection and no self-shadowing. The middle picture integrated self-shadowing, and the last one all effects. Note that there is no real difference between the two pictures on the right, except on the shadow regions in the texture that are completely black for the middle part, as opposed to the right one.

5 Rendering using ray tracing

The radiosity simulation provides a global solution of the rendering equation. Unfortunately, only mean values are given on the patches. These mean values cannot be turned into individual radiance values (at least without introducing serious errors) in the case of important variances (for example of the local BRDF). Final gathering [28] is a technique that helps to improve the quality of the final rendering. It has been used in the context of radiosity with clustering, as for example in [21], or in the context of Monte Carlo techniques, as for example in [23]. This technique is best suited for inhomogeneous surfaces, since it consists in a complete re-computation of the integral equation. In addition, it provides high quality results.

Basically, the global solution computed by the radiosity simulation can be ”re-used” to compute the outgoing radiance on each point P of a surface S, by summing up over the hemisphere of P (a sphere in the case of transmittance). Rays can be ”re-traced” towards all clusters (or patches) to determine more precisely shadows. Unfortunately, using a complete integration on all points of the scene induces excessive computational requirements, making such an ”extremely accurate” approach nearly unusable in practice. To avoid such an excessive computation, we use interpolations in regions of low variance by constructing a data structure similar to the 5D tree of [22], except that we use a 4D tree (two dimensions for the surface u and v parameters and two for the directions). In [14], we also used a similar approach. Nonetheless, to avoid excessively smoothing sharp shadows, we also keep track of those clusters and patches that produce important variations. Therefore, we also add a ”gathering” list l_i to the data structure.

Let be P_i the patch containing the point P, where the shading must be computed. If P_i is ”visited” for the first time, a certain data structure is initialized (the data structure consists of a hemisphere of energy on each vertex, plus the gathering list l_i). For each cluster Cl_j (or eventually each patch P_j at the lowest level) – starting from the global scene cluster, and refining according to an error bound criterion, with similarity to radiosity – the amount of incoming radiance is computed on the vertices of P_i (as well as on some randomly chosen points to get a better estimation of the variance). If the variance of energy is below a given error threshold ϵ, then the radiance coming from Cl_j (or respectively P_j) is stored in the 4D structure (in the hemisphere defined on each vertex, which has been previously initialized to zero). Otherwise Cl_j (or respectively P_j) is added to the gathering list l_i. Now, the shading on P can be computed by using both the hemispheres on the vertices (using a bilinear interpolation, as done in [22] in a 3D case) and the list l_i. For the latter, stochastic rays are traced, since these clusters (respectively patches) produce important variances (shadow frontiers for example). The data structure on P_i needs only be computed once. For low variances the list l_i will be nearly empty, thus only a very low number of rays will be traced, which considerably accelerates the rendering (compared to usual gathering). On the other hand, some discontinuities may appear in the case of high error bounds around the patches, since for two neighboring patches, the gathering lists may be different.

6 Results

Figure 7 illustrates a comparison between using textures and using ”explicit” models. The two upper pictures correspond to the texturing approach. In this case, the scene contains only 1165 patches. The radiosity simulation required 13 seconds on an PC Pentium II 350MHz with 128Mb RAM for 3 iterations and with a low error bound of 1.0 (this corresponds to 0.4 percent of the initial light source value). 2870 links were created and the memory required for storing the scene and during the radiosity was about 3.1Mb. The left part shows the final result obtained with the previously described ray tracing approach without considering self-shadows. We used a medium error bound $\epsilon = 0.1$ to decide whether or not to add the emitter to the gathering list. The rendering took 33.2 minutes for a resolution of 500x400 pixels, with 9 rays per pixel. An average of 2 gathering rays per patch point was traced. The right part shows the result with self-shadows (the difference is mainly visible on the walls). In this case, the ray tracing took about 85 minutes.

The bottom part of figure 7 illustrates an approach based on ”explicit” patches. In this case, the scene contains 65477 patches. The radiosity simulation required 16.22 minutes for 4 iterations and with an error bound criterion of 2.0 (this corresponds to 1.0 percent of the initial light source value). 12407 links were created and the memory required was about 31.2Mb. The obvious difference concerning the computation time and memory requirement of the radiosity compared to the texturing approach is mainly due to the fact that there are nearly 53 times more patches. The ray tracing required 24.5 minutes. Unfortunately, we could not run the previously described gathering methodology because of memory limitations (storing lists and hemispheres on more than 100000 vertices was not possible). We also tried to apply a gathering step without ”optimization” using interpolations, but in this case the computation time became excessive (we stopped the computation after several hours). The rendering shows only an interpolation on the patches, in particular the shadows are of poor quality. Nonetheless, it allows a visual comparison with the texturing approach. For example, we used a self-shadow map of low resolution (8x4 directions), which may explain the difference on the bottom of the walls. Also the mapping has ”stretched” the spheres.

We note that it would be interesting to develop a fast technique to directly visualize the radiosity solution in the case of textures, without using a final ray tracing approach.

Figure 8 illustrates an additional example. All of the surfaces except the ceiling are textured (even the floor). We show results for two different lighting conditions. The radiosity solution for both figures required approximately 3 minutes, and the final ray tracing 40 minutes (left) and 130 minutes (right). The difference is due to the number of light sources.

7 Conclusion

In this paper, a solution for integrating general inhomogeneous surfaces (surfaces with complex non-constant reflectance behaviors and complex macro-geometry) into radiosity simulations was proposed. Therefore, some light scattering behaviors were precomputed at different scales using simulations similar to BRDF computation techniques. Nearly all inter-reflection phenomena, including inter-reflections at texture scale and self-shadowing, were considered. Aliasing was also addressed using a transition between virtual ray tracing and BRDF MIP-maps (BTFs). The final gathering step provides high precision results, while the hierarchical formulation and the error bounds allow one to control the ratio between quality and computational efficiency. The method can also be extended (with minor changes only) to more advanced texturing techniques such as texel maps for example.

However, several topics need to be further investigated. For example, in this paper, we do not consider specular to diffuse transfers, but only directional diffuse transfers. Therefore, caustics (due to metal links [19] for example) cannot be rendered. An other important problem is the fact that we always repeat ”the same” texture sample (periodicity), as for texel mapping. For some random textures (street pavement for example), the repetition effect might become unpleasant.

References

1. Blinn J.F., ”Simulation of wrinkled surfaces”, Computer Graphics 12, 1978, pp. 286–292.
2. K. Perlin, ”An Image Synthesizer”, Computer Graphics 19(3), 1985, pp. 287–296.
3. Peachey D., ”Solid Texturing on complex Surfaces”, Computer Graphics 19(3), 1985, pp. 279–286.

4. R.L. Cook, ”Shade Trees”, Computer Graphics 18 (Siggraph’84), pp. 223–231, 1984.
5. K. Perlin and E.M. Hoffert, ”Hypertexture”, Computer Graphics 23(3), pp. 253–262, 1989.
6. J.T. Kajiya, and T.L. Kay, ”Rendering fur with Three Dimensional Textures”, Computer Graphics 23(3) (Siggraph’89), pp. 271–280, 1989.
7. M.F. Cohen, S.E. Chen, D.S. Immel and P.J. Brock, ”An Efficient Radiosity Approach for Realistic Image Synthesis”, IEEE CGA 6(3), pp. 26–35, 1986.
8. R. Gershbein, P. Schroeder and P. Hanrahan, ”Texture and Radiosity: Controlling Emission and Reflection with Texture Maps”, Computer Graphics (Siggraph’94), pp. 51–58, 1994.
9. H. Chen, and E-H. Wu, ”An Efficient Radiosity Solution for Bump Texture Generation”, Computer Graphics 24(4)(Siggraph’90), pp. 125–134, 1990.
10. F.X. Sillion, J.R. Arvo, S.H. Westin and D.P. Greenberg,”A Global Illumination Solution for General Reflectance Distributions”, Computer Graphics 25(4), pp. 187–196, 1991.
11. F.X. Sillion, ”Clustering and Volume Scattering for Hierarchical Radiosity Calculation”, Fifth EGWR, June 1994.
12. B. Smits, J. Arvo and D.P. Greenberg, ”A Clustering Algorithm for Radiosity in Complex Environments”, Computer Graphics (Siggraph’94), pp. 435–442, 1994.
13. M. Pharr, C. Kolb, R. Gershbein and P. Hanrahan, ”Rendering Complex Scenes with Memory-Coherent Ray Tracing”, Computer Graphics (Siggraph’97), 1997.
14. J.M. Dischler, L. Mostefaoui and D. Ghazanfarpour, ”Radiosity Including Complex Surfaces and Geometric Textures Using Solid Irradiance and Virtual Surfaces”, Computers and Graphics 23(4), 1999 (to appear).
15. Dana K.J., Nayar S.K, van Ginneken B. and Koenderink J.J., ”Reflectance and Texture of Real-World Surfaces”, IEEE Conf. on Comp. Vision and Pattern Recognition, 1997.
16. L. Williams, ”Pyramidal parametrics”, Computer Graphics 17(3), pp. 1–11, 1983.
17. T. Noma, ”Bridging between surface rendering and volume rendering for multi-resolution display”, Proc. EGWR, pp. 31–40, 1995.
18. F. Neyret, ”A general and multiscale method for volumetric textures”, Proc. Graphics Interface’95, pp. 83–91, 1995.
19. J.M. Dischler, ”Efficiently Rendering Macro-geometric Surface Structure with Bi-directional Texture Functions”, Proc. EGWR, pp. 169–180, 1998.
20. F.X. Sillion, G. Drettakis and C. Soler, ”A Clustering Algorithm for Radiance Calculation in General Environments”, Proc. EGWR, pp. 196–205, 1995.
21. P.H. Christensen, D. Lischinski, E.J. Stollnitz and D.H. Salesin, ”Clustering for Glossy Global Illumination”, ACM TOG 16(1), pp. 3–83, January 1997.
22. E. P. Lafortune and Y. D. Willems, ”A 5D Tree to Reduce the Variance of Monte Carlo Ray Tracing”, Proc. of 6th EGWR, 1995.
23. H. W. Jensen and N. J. Christensen, ”Photon Maps in Bi-directional Monte Carlo Ray Tracing of Complex Objects”, Computers and Graphics 19(2), pp. 215–224, 1995.
24. J.P. Ewins, M.D. Waller, M. White and P.F. Lister, ”MIP-map Level Selection for Texture Mapping”, IEEE TVCG 4(4), pp. 317–329, 1998.
25. Westin S. H., Arvo J. R. and Torrance K. E., ”Predicting Reflectance Functions from Complex Surfaces”, Computer Graphics 26(2), pp. 255–263, 1992.
26. Kajiya J., ”The Rendering Equation”, Computer Graphics 20(4), pp. 143–150, 1986.
27. P.M. Deville and J.C. Paul, ”Modeling the spatial Energy Distribution of Complex Light Sources for Lighting Engineering”, Proc. of EGWR, pp. 147–159, 1995.
28. M.C. Reichert, ”A Two-pass Radiosity Method Driven by Lights and Viewer Position”, Master’s thesis, Program of Computer Graphics, Cornell University, 1992.
29. Max N., ”Horizon mapping: shadows for bump-mapped surfaces”, The Visual Computer 4, 1988, pp. 109-117.

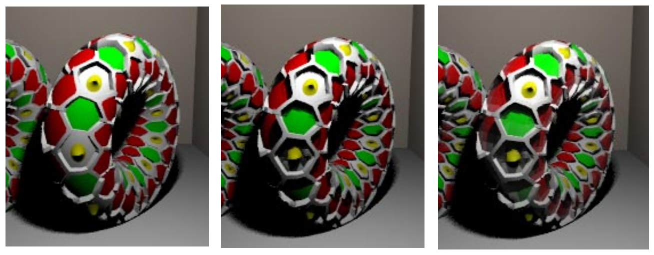

Fig. 6. Three tori showing from left to right the difference between considering: no effect, self-shadowing and self-shadowing plus inter-reflections.

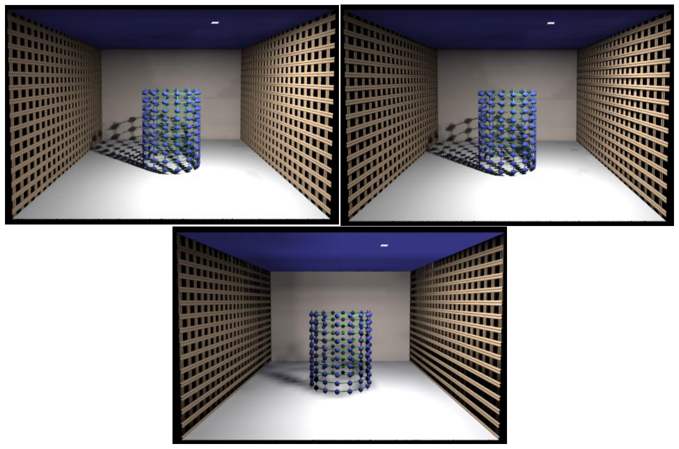

Fig. 7. A Comparison between a texturing approach and an ”explicit” approach.

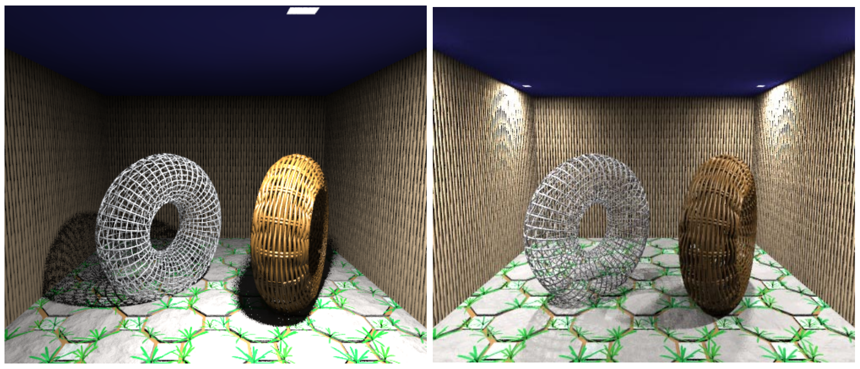

Fig. 8. A scene showing textures under different lighting conditions.