Environment Mapping for Efficient Sampling of the Diffuse Interreflection

Erik Reinhard, Lucas U. Tijssen, Frederik W. Jansen

Faculty of Technical Mathematics and Informatics, Delft University of Technology, Julianalaan 132, 2628BL Delft, The Netherlands

Environment mapping is a technique to compute specular reflections for a glossy object. Originally proposed as a cheap alternative for ray tracing, the method is well suited to be incorporated in a hybrid rendering algorithm. In this paper environment mapping is introduced to reduce the amount of computations involved in tracing secondary rays. During rendering, instead of tracing the secondary rays all through the scene, values are taken from the maps for the rays that would otherwise hit distant objects. This way the quality of the image is retained while providing a cheap alternative to stochastic brute force sampling methods. An additional advantage is that due to the local representation of the entire 3D scene in a map, parallelising this algorithm should result in a good speed-up and high efficiency.

1 Introduction

In the past, a number of approaches to rendering have been proposed. Image quality and speed are the two main goals set in the computer graphics community. If speed is the dominant factor, e.g. in flight simulators and architectural walk-throughs where images must be produced in real-time, z-buffer-based algorithms and hardware can be used. By using a radiosity pre-processing, a more realistic shading can be obtained. Disadvantages are that specular reflection can not be modelled and that the quality of the solution strongly depends upon the resolution of the radiosity mesh.

The other approach to rendering is the use of sampling algorithms, such as ray tracing or ray tracing based radiosity. There are several versions of two-pass algorithms which consist of a radiosity and a ray tracing pass. These hybrid algorithms differ in the amount of sampling done during the rendering pass. The standard two-pass algorithm only samples specular reflection during rendering, while the radiosity values are used for diffuse reflection and direct lighting (Sillion and Puech 1989; Wallace et al. 1987). An extended version also samples direct light during rendering (Shirley 1990, 1991; Chen et al. 1991; Kok and Jansen 1991) and finally, diffuse light may be sampled in the rendering stage in addition to specular reflection and direct light (Rushmeier 1988; Chen et al. 1991).

Sampling of indirect light allows the simulation of intricate reflection details. However, in complex environments this method will be very expensive and moreover it may bring about aliasing problems. To make aliasing less visible and to obtain a reasonable estimate of the indirect diffuse reflection, either a huge number of samples has to be taken, or stochastic techniques may be applied. Stochastic sampling effectively turns aliasing into noise, which is less perceptible to the human eye.

However, a large number of samples is still needed. Therefore, using today's hardware, it is not possible to compute an image interactively with these methods. There are two approaches to reduce computation times of high quality rendering algorithms. First the algorithm can be optimised by reducing the number of redundant computations. Examples of such improvements are spatial subdivision techniques (Glassner 1989) and grouping (Rushmeier 1993; Kok 1993). Second, rendering algorithms may provide good opportunities to be efficiently implemented on multicomputers.

An implementation of a hybrid algorithm on a distributed memory MIMD computer will likely present problems when the scenes to be rendered are large. Then the scene database can not be replicated with every processor due to memory restrictions. Some distribution scheme will have to be applied instead. Objects can be assigned to the processors randomly, completely ignoring scene coherence. A better idea is to base the object distribution upon a spatial subdivision structure. Then the objects that are contained within a region of the environment, are assigned to the same processor. A task is executed by the processor that holds the relevant data in its local memory. This data parallel solution may, however, give rise to excessive load imbalances. Another approach is to schedule the computation tasks in a demand driven way, but then processors may need to request data from other processors, resulting in large communication overheads. This is in particular the case for secondary rays where data coherence is low.

In order to solve both sampling and parallelisation problems, we propose to use environment mapping in an adapted form. Where in Blinn and Newell (1976) environment mapping is presented as a cheap alternative for ray tracing, in our approach, which is similar to (Greene 1986), environment mapping is part of a hybrid rendering algorithm to handle secondary rays efficiently. The sampling related problems are solved by pre-filtering the maps. The communication problem is releaved by providing a means to efficiently store information of remote objects in a local environment map. Thus data retrieval becomes a local process, which removes the need to link data management and task scheduling. Therefore, a more flexible parallel implementation may be achieved.

The complete rendering algorithm consists of three separate stages. First, a standard radiosity preprocessing is performed. A coarse mesh is used in this stage, as in subsequent stages only a rough estimate of patch radiances is needed. In the second stage, objects are grouped and for each group an environment map is generated. This is a view independent operation. The map generation may be viewed as a pre-processing of secondary rays. Normally, the origin of secondary rays is determined by the intersection point of a primary ray and a surface. During map building, all rays are traced from the centre point. However, as in the rendering stage secondary rays are generally not traced from the centre point, an error will occur. How to keep this error small is discussed in the following section. In the last stage the scene is rendered using environment maps and radiances.

The map generation and rendering stages are presented in greater detail in the next section. Experiments with an implementation of the environment mapping algorithm are discussed subsequently, while in the last section conclusions are drawn.

2 Method

The environment mapping technique was first introduced to computer graphics by Blinn and Newell (1976). It is a method to enhance an object with reflections without explicitly tracing secondary rays. This is accomplished by projecting the 3D environment onto a 2D environment map that surrounds the glossy object. Instead of intersecting secondary rays with objects, an index in the environment map is computed from the surface normal of the object and the angle of the incoming ray (see Figure 1). Environment mapping can also be used to reflect digitised photographs in objects.

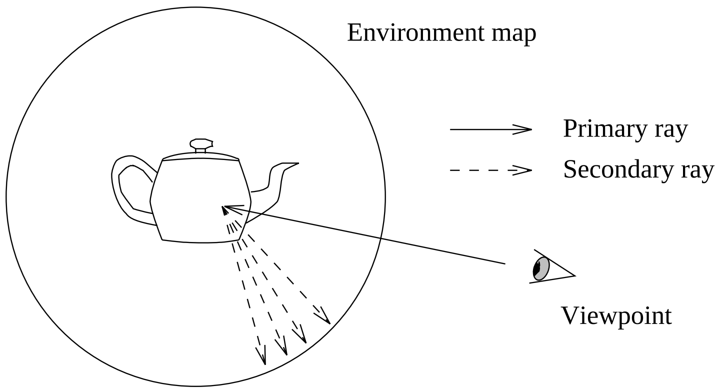

Fig. 1. For every secondary ray, a look-up in the environment map is done, instead of casting secondary rays

Following (Miller and Hoffman 1984), in (Greene 1986) environment mapping was augmented with filtering capabilities to support a more general reflection model with both diffuse and specular reflection. The single environment map was replaced by a cube with six maps that were placed around the object. For each of the six maps, a mipmap was generated. They are indexed according to the specularity of the object that the maps surround. Mip-mapping is a pre-filtering method (Williams 1983) that filters colour arrays, such as texture maps or environment maps.

In the environment mapping algorithm as proposed in this paper, the emphasis is on cost reduction of tracing secondary rays, as suggested in (Hall 1986). To achieve this, the scope of these rays is limited to the boundaries of the cube on which the maps are projected. Within these limits, rays are actually traced, as other objects within the cube may be hit first. Because most of the scene is outside the cube, many intersection computations do not have to be performed. This suggests that the smaller the cube is, the greater the savings, as more objects will be on the outside.

However, this also increases the error that environment mapping introduces, because the maps are generated with respect to the centre of the cube, but due to the size of the object(s) within the cube, secondary rays generally do not originate from the centre of the cube. Therefore, the size of the cube is preferably large with respect to the objects within. Also, the objects for which environment mapping is a suitable technique, are relatively small and cubicly shaped. Another error source is the distance of outside objects to the environment map. The closer these objects are to the environment map, the smaller the error.

As long as the condition of small and cubic objects is satisfied, there is no objection to clustering multiple objects together and to assigning them a single environment map. Good candidates for clustering are for example plants, keyboards and generally small objects with much detail. Not very suitable are long stretched objects such as floors and walls etc.

Environment mapping consists of two distinct stages. First the maps must be generated and then the maps can be used in the rendering phase. During the generation of the environment maps, first the centre of the clustered object is determined. Then around this centre point a large cube is placed on which six environment maps will be projected, one for each side. The resolution of the maps should be high enough to capture sufficient detail for the subsequent rendering stage. Especially specular surfaces need detailed environment maps. From the centre point a number of rays are shot through each map element. Only objects outside the maps are intersected with these rays. A map entry is generated by performing standard ray tracing. After the map entries have been computed, the mip-maps are built by recursively down-filtering the environment maps. This completes the view-independent map generation.

In the rendering stage, for each pixel primary rays are shot, which gives rise to a large number of secondary rays. In our algorithm, with the exception of shadow testing point light sources and selected area light sources, these functions are largely performed by sampling the environment maps. According to the angle of incidence of the primary ray and the bi-directional reflection distribution function (brdf) of a surface, a number of secondary rays are spawned and traced within the scope of the surrounding cube.

An example of possible trace paths is given in Figure 2. Both teapot and table possess an environment map. A primary ray hits the teapot and spawns secondary rays. Some of these rays will hit the table. For these rays according to the brdf of the table, new rays are spawned which are bounded by the environment map belonging to the table.

The distribution of secondary rays is determined by the brdf of the surface, i.e. more rays are sent in the reflected direction than in other directions. Therefore, the angle between neighbouring rays is smaller in the reflected area. Figure 3 shows two brdf's, where the one on the left belongs to a surface that is partly diffuse and partly specular. The brdf on the right belongs to a completely diffuse surface. Because the environment maps are mip-mapped, the filter level must be computed before an entry can be calculated. The brdf determines the angle between successive rays (Figure 3) and this angle provides a means to determine the mip-map level. The closer the rays are together, the less filtering is needed. For more diffuse areas, stronger filtered versions of the environment map should be used. The link between brdf and index level is depicted in Figure 4.

Because no directed shooting is used for both building the environment maps and sampling the maps, there is no guarantee that point light sources are sampled accurately. Even if they are traced separately during the generation of the maps, point light sources are not well represented in the map. Therefore, these light sources are traced in the rendering stage, partly bypassing the environment maps. Also, sources that are selected

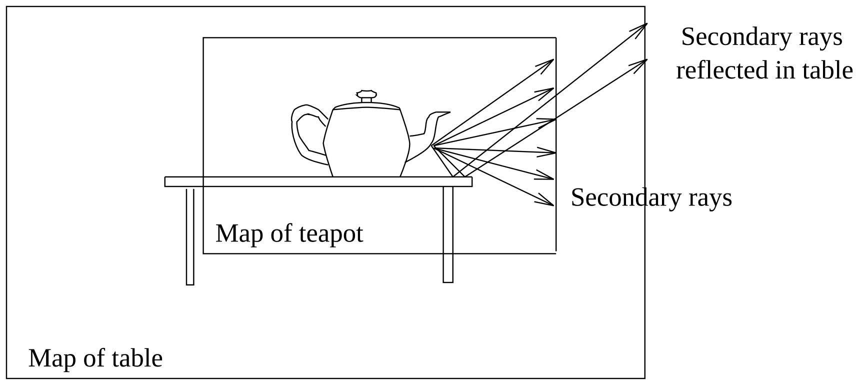

Fig. 2. Example of possible paths of secondary rays

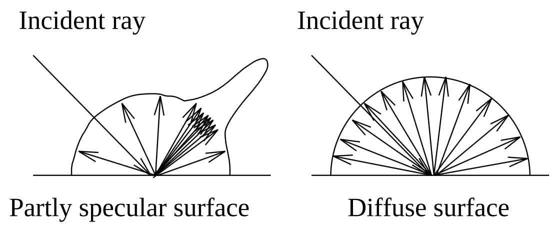

Fig. 3. Secondary rays spawned according to angle of incidence and brdf

for source sampling are kept outside the map in order to keep the amount of noise low.

Pure specular surfaces may be handled separately as well, given the fact that the error introduced by using environment maps is largest for perfect mirrors. Finally, the advantage of having a limited scope for secondary rays becomes less when the objects outside the environment map are located close to the environment map. These examples show that environment mapping is most suitable in those cases where sampling does not significantly influence the quality of the image.

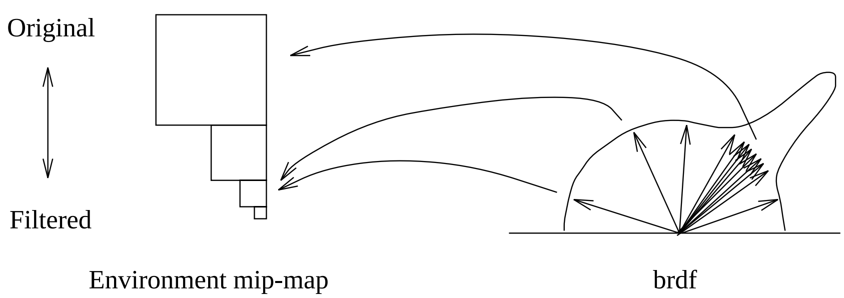

Fig. 4. Brdf determines mip-map index level

3 Experiments

Tests have been performed to establish the image quality that can be achieved using environment mapping. Also, the method is compared with respect to quality and rendering times with two two-pass radiosity algorithms. The first shoots primary rays and uses the pre-computed radiosity values to avoid any diffuse secondary rays. Also source selection is performed to limit the number of secondary shadow rays. In the remainder, this algorithm will be called 'source selection' (Kok and Jansen 1991). The second algorithm taking part in the comparison, shoots primary rays and for each object/primary ray intersection, both diffuse and specular secondary rays are shot. However, instead of shooting tertiary diffuse rays, the pre-computed radiosity values are used. The number of shadow rays is limited by using source selection. This algorithm is also known as 'one-level path tracing' (Rushmeier 1988); in this paper to be abbreviated as 'path tracing'.

All tests have been performed on a test scene consisting of a table on which a ball is placed. The table is in a room with a textured ceiling and four area light sources. The table has an environment map, so that the walls and the ceiling are outside the map and the table and the ball are inside the map. The objects outside are projected onto the map, which is shown in Figure 5.

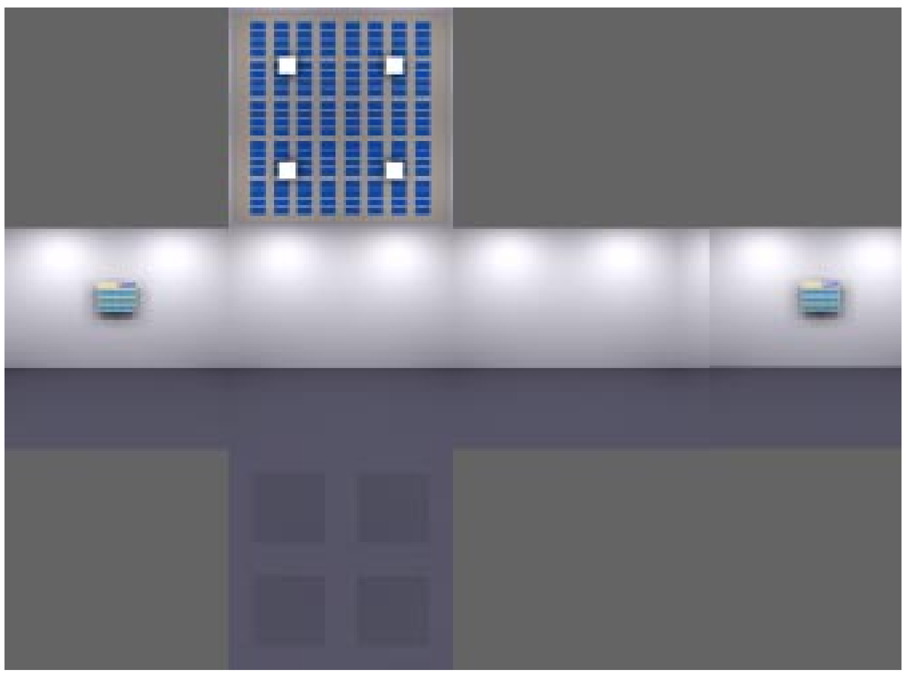

Fig. 5. Environment map generated for table in test environment

Two potential sources for errors have been defined in the preceding paragraph.

These are the finite resolution of the maps and the discrepancy between the origin of secondary rays during map generation and rendering. The test scene (with a specular reflecting table top) has been rendered using different map resolutions, results of which are shown in Figure 6. The rendering times for these pictures are given in Table 1[1]. The effect of differences between the origin of rays during map generation and rendering can be varied by enlarging the environment map with respect to the table. In Figure 7, the size of the environment map is increased from left to right. Corresponding rendering times are shown in Table 1.

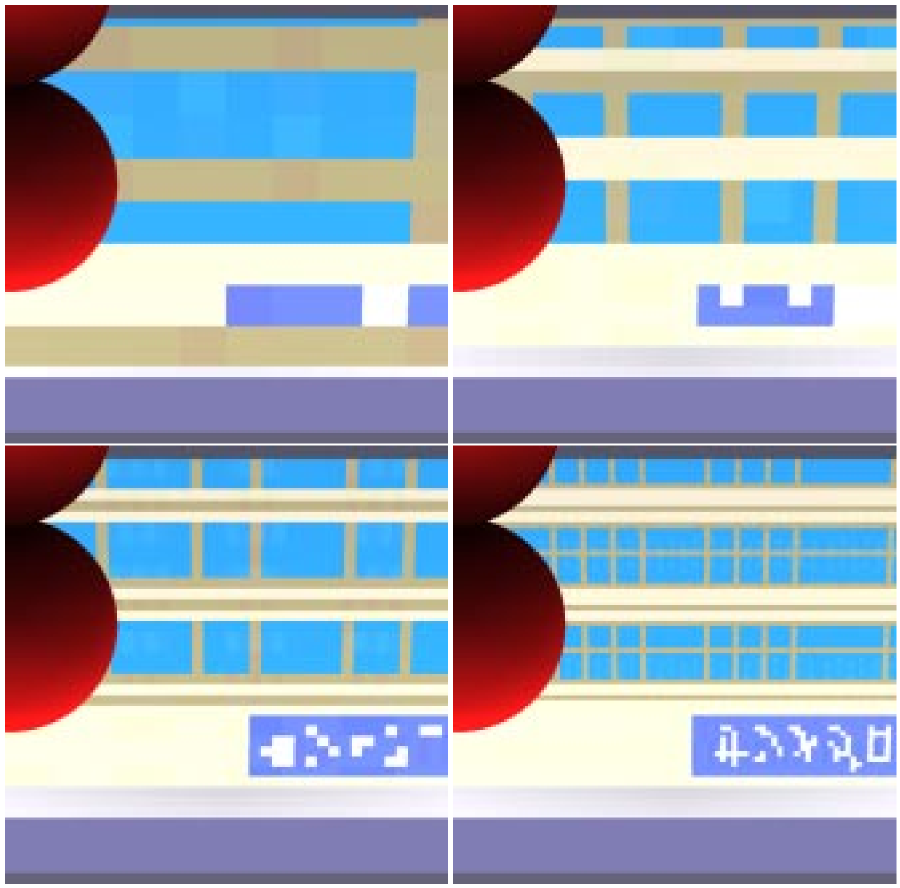

Fig. 6. Test scene rendered with increasing environment map resolution. The table top is a purely specular reflector

These images show that when the resolution is chosen too low, aliasing occurs. For this test environment, which is a worst case in the sense that the camera-point is zoomed

[1] All rendering has been performed on a Silicon Graphics Indigo.

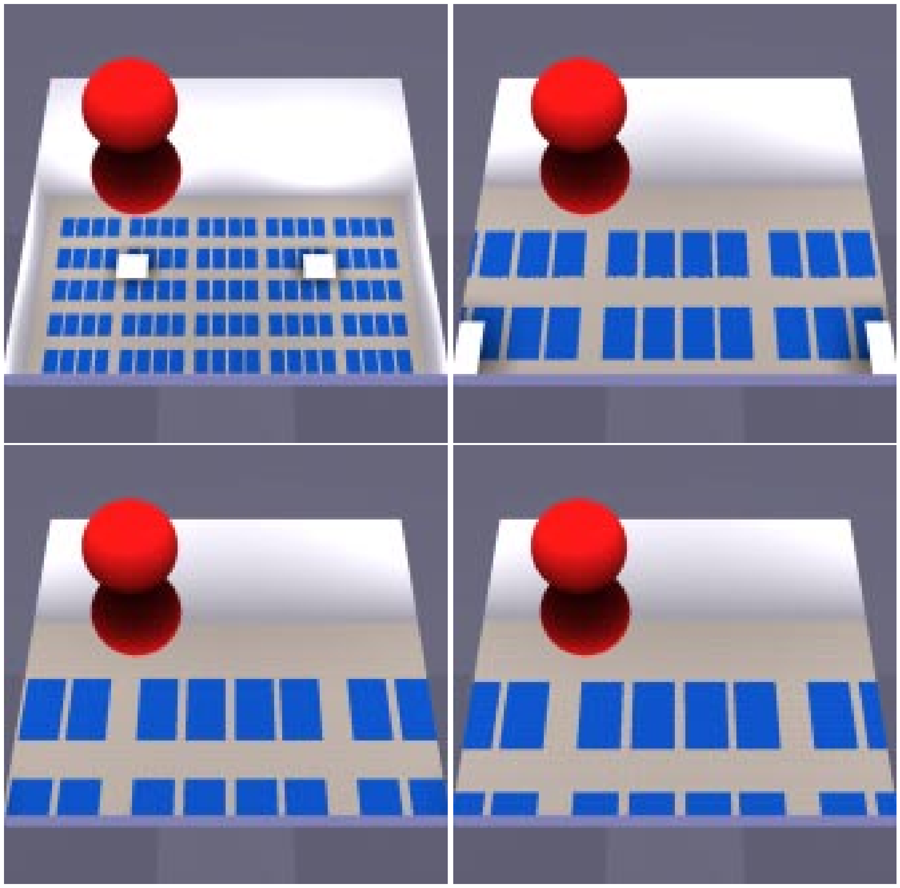

Fig. 7. Test scene rendered with increasing map size. The table top is a purely specular reflecting surface

in on a specular reflecting table, a resolution of at least 512 x 512 is needed to generate acceptable images. For more distant view points a lower resolution can be used. The same holds for more diffuse objects.

The amount of time needed to generate the environment maps quadruples when doubling the resolution in both u and v directions. This is in accordance with expectations, as four times as many rays are cast. Actual rendering times do not vary as a function of the map resolution, because the number of secondary rays that are traced remains constant.

Varying the distance between table and environment map, it turns out that the dis-

[2] The resolution is the number of map entries in both u and v directions. The distance ratio is $1 : 12.5$.

[3] The ratios given in these columns are the size of the table divided by the distance of the centre point of the environment map. The resolution of the map is 512 x 512.

Table 1. Rendering times in min:sec for test scene using different environment map resolutions and different distances between centre point and map

	Resolution[2]				Distance ratio[3]			
	64	128	256	512	1 : 1.25	1 : 5	1 : 10	1 : 14
Radiosity	9 : 45	9 : 45	9 : 45	9 : 45	9 : 45	9 : 45	9 : 45	9 : 45
Env. gen.	0 : 16	1 : 05	4 : 22	17 : 23	31 : 28	23 : 00	17 : 59	16 : 42
Rendering	6 : 51	6 : 54	6 : 53	6 : 55	9 : 35	9 : 33	9 : 44	9 : 49
Total	16 : 52	16 : 44	21 : 00	33 : 53	50 : 38	42 : 12	37 : 28	36 : 16

location of objects outside the environment map becomes within bounds when the distance ratio is around $1 : 10$. If the map is placed more closely to the table, the dislocation of the reflection increases.

During rendering the effect of the size of the maps on the rendering times is less significant. Here, smaller environment maps mean slightly shorter rendering times. This result is opposite to the relation between size and map generation time, because now a smaller map allows secondary rays to be traced along a shorter distance.

The test scene as used in the preceding test has been used for the comparison with the afore mentioned algorithms as well. The table top, however, is now 20% specular reflecting and 80% diffuse. In Figure 8 (see also image in colour section), the results of the algorithms are shown. Both the environment mapping and the path tracing picture exhibit colour bleeding between the ball and the table and the ball casts a more accurate and darker shadow upon the table than is the case with the source selection algorithm. The ball also receives more reflected light from the table. These effects can be attributed to the diffuse sampling performed by the environment mapping and path tracing algorithms. As the shading is completely independent of the local radiosity mesh, artifacts due to insufficient meshing are thus avoided with this algorithms.

The number of rays for each algorithm are given in Table 2 and the rendering times are given in Table 3. For this simple test scene, the differences are not very prominent. For more complex environments the rendering times for the environment mapping algorithm will not increase much, while the rendering times for path tracing will grow significantly. The environment map generation times are expected to grow slowly with increasing scene complexity, while the source selection algorithm's time complexity is relatively independent of the scene to be rendered.

4 Discussion

Environment mapping is a technique which can be incorporated in a ray tracing based radiosity algorithm. With the exception of elongated objects, environment mapping can be used for single objects or groups of objects. Three important parameters that influence the accuracy of the images are the resolution of the maps, the distance between

[4] Number of rays per pixel or per map element.
[5] Number of rays generated per intersection.

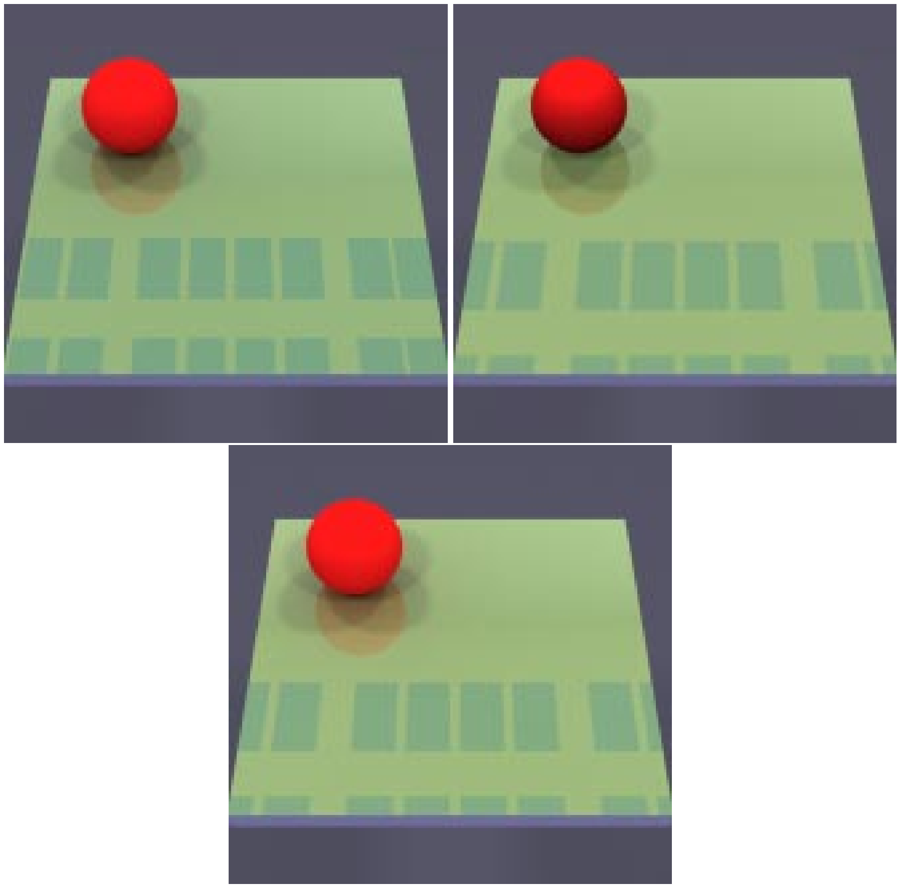

Fig. 8. Qualitative comparison of hybrid rendering algorithms. In the upper left the three stage environment mapping algorithm, the result of the source selection algorithm is to the right and the path tracing picture is below

Table 2. Number of rays shot

Type of ray	Env. mapping		Source sel.	Path Tr.
	Map gen.	Rend.		
Primary[4]	1	1	1	1
Shadow[5]	16	16	16	16
Diffuse[5]	0	100	0	100
Specular[5]	1	1	1	1

Table 3. Timings of the the rendering algorithms in hour:min:sec

Pass	Env. mapping	Source sel.	Path Tr.
Radiosity	$9:45$	$9:45$	$9:45$
Env. map generation	$1:50:52$	n.a.	n.a.
Rendering	$7:17:43$	$5:39:08$	$11:17:36$
Total	$9:08:20$	$5:48:53$	$11:27:21$

the centre point and the environment maps and the location of objects outside the environment maps. The resolution needs to be highest for perfect mirrors, while a lower resolution is sufficient for glossy and diffuse surfaces.

The distance between maps and the centre point depends on the size of the objects for which the maps are generated. If the ratio between these two is chosen too small, then the reflected surroundings appear dislocated. With respect to the quality of the images, both environment mapping and path tracing allow more accurate shadows due to the diffuse sampling that is performed during rendering. In addition, these methods are capable of handling more complicated brdf's than source selection, which splits a surface's reflection properties into a specular and a diffuse component of which the latter is taken from the radiosity mesh.

A disadvantage may be that this rendering technique requires more memory, as in addition to the scene description, a number of possibly large environment maps need to be stored. However, if a similar qualitative result were to be obtained without environment mapping, a much finer radiosity mesh would be needed, which largely cancels out the memory disadvantage of environment mapping.

Due to the more local data references made during rendering, environment mapping is better suited for parallel implementation than the source selection and path tracing algorithms. Each processor in a MIMD computer could be assigned a number of objects with their associated environment maps. This results in an object space subdivision where all objects within an environment cube, are physically stored with the same processor. Rendering can then be accomplished with minimal communication requirements, as only primary rays need to be distributed and results must be transferred to the frame buffer. Secondary rays may also induce communication, but all diffuse and specular secondary rays are handled locally. For the generation of the environment maps, communication between processors will still be required. However, the number of rays needed in this stage is relatively small compared with the amount of sampling which would otherwise be needed during rendering.

As only a sequential implementation of the environment mapping algorithm exists, our future plans include implementing this algorithm on a transputer system. Performance and scalability issues will then be examined. The ability to use more accurate brdf's has not been fully exploited yet in the implementation discussed in the preceding paragraph. Therefore, inclusion of more realistic brdf's remains work to be done. Finally, we would like to extend this algorithm so that large flat objects can be handled correctly as well.

References

Blinn, J. F., Newell, M. E.: Texture and reflection in computer generated images, Communications of the ACM **19**(10), 542–547, (1976).

Chen, S. E., Rushmeier, H. E., Miller, G., Turner, D.: A progressive multi-pass method for global illumination, Computer Graphics **25**(4), 165–174, (1991).

Glassner, A. S.: An Introduction to Ray Tracing, Academic Press, San Diego (1989).

Greene, N.: Environment mapping and other applications of world projections, IEEE Computer Graphics and Applications (1986) 21–29.

Hall, H.: Hybrid Techniques for Rapid Image Synthesis, course notes, SIGGRAPH '86: Image Rendering Tricks, (1986).

Kok, A. J. F., Jansen, F. W.: Source selection for the direct lighting computation in global illumination, *in* Proceedings Eurographics Workshop on Rendering, Barcelona, Spain (1991).

Kok, A. J. F.: Grouping of patches in progressive radiosity, *in* M. Cohen, C. Puech , F. Sillion, eds, Fourth Eurographics Workshop on Rendering, Paris, France, 221–231 (1993).

Kok, A. J. F., Jansen, F. W., Woodward, C.: Efficient, Complete Radiosity Ray Tracing Using a Shadow-coherence Method, The Visual Computer, **10**(1993) 19–33.

Miller, G. S., Hoffman, C. R.: Illumination and Reflection Maps: Simulated Objects in Simulated and Real Environments, SIGGRAPH '84: Advanced Computer Graphics Animation Seminar Notes, (1984)

Rushmeier, H. E.: Realistic Image Synthesis for Scenes with Radiatively Participating Media, PhD thesis (1988).

Rushmeier, H. E., Patterson, C., Veerasamy, A.: Geometric Simplification for Indirect Illumination Calculations, Graphics Interface '93, 227–236 (1993).

Shirley, P.: A Ray Tracing Method for Illumination Calculation in Diffuse Specular Scenes, Graphics Interface '90, 205–212 (1990).

Shirley, P.: Physically Based Lighting Calculations for Computer Graphics, PhD thesis, Urbana-Champaign (1991).

Sillion, F., Puech, C.: A general two-pass method integrating specular and diffuse reflection, ACM Computer graphics (1989), SIGGRAPH '89.

Wallace, J. R., Cohen, M. F., Greenberg, D. P.: A two-pass solution to the rendering equation: A synthesis of ray tracing and radiosity methods, ACM Computer Graphics **21**(4), 311–320, (1987), SIGGRAPH '87.

Williams, L.: Pyramidal parametrics, ACM Computer Graphics **17**(3), 1–11, (1983).

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