Localizing the Final Gathering for Dynamic Scenes using the Photon Map

Takehiro Tawara, Karol Myszkowski, Hans-Peter Seidel

Max-Planck-Institut für Informatik, Computer Graphics Group
Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany
Email: {tawara, karol, hpseidel}@mpi-sb.mpg.de

Abstract

Rendering of high quality animations with global illumination effects is very costly using traditional techniques designed for static scenes. In this paper we present an extension of the photon mapping algorithm to handle dynamic environments. First, for each animation segment the static irradiance cache is computed only once for the scene with all dynamic objects removed. Then, for each frame, the dynamic objects are inserted and the irradiance cache is updated locally in the scene regions whose lighting is strongly affected by the objects. In the remaining scene regions the photon map is used to correct the irradiance values in the static cache. As a result the overall animation rendering efficiency is significantly improved and the temporal aliasing is reduced.

1 Introduction

Global illumination is an important visual cue, which greatly improves the appearance of computer animations. However, the vast majority of state-of-the-art global illumination algorithms require repeating the computation from scratch for even minor changes in the scene. This leads to redundant computations which could be mostly avoided by taking into account the temporal coherence of global illumination in the sequence of animation frames. Another important problem is the temporal aliasing which is more difficult to combat efficiently if temporal processing of global illumination is not performed. Many small errors in lighting distribution cannot be perceived by the human observer when they are coherent in the temporal domain while may cause unpleasant flickering and shimmering effects when such a coherence is lost.

The algorithm of choice for the final rendering of high quality images is the so-called *final gathering* [14, 9, 19, 2]. Usually the direct lighting is computed for a surface region represented by a given pixel, and the indirect lighting is obtained through the integration of incoming radiances, which is very costly. Those costs can be reduced by using the *irradiance cache* data structures [20, 21] to store irradiance samples sparsely in the object space. The cached values are used to interpolate the indirect lighting for each pixel and are computed lazily. The coarse distribution of lighting, which is used for the irradiance integration can be computed in the preprocessing stage using a deterministic or stochastic radiosity [18], photon maps [7], and so on. The irradiance cache technique efficiently removes shading artifacts which are very difficult to avoid if the indirect lighting is directly reconstructed based on the radiosity mesh or the photon maps. However, the price to be paid for this high quality lighting reconstruction are long computation times, which are mostly caused by the irradiance integration that is repeated for many sample points in the scene.

In this work we extend the concept of irradiance cache for dynamic environments to improve the rendering performance and reduce the temporal aliasing. Also, we extend the photon mapping algorithm make the global illumination computation more efficient for such dynamic environments.

In the following section, we briefly overview existing solutions for the global illumination computation for dynamic environments and then we discuss the final gathering for animations, which mostly remains an unexplored problem.

2 Previous Work

Existing global illumination solutions for dynamic environments can roughly be categorized as interactive and off-line. The interactive techniques are designed to trade the image quality for the response speed in order to preserve a constant frame rate

[13, 5, 17, 6] (for a complete survey refer to [3]). Their main objective is to provide a fast response for frame-to-frame changes in the environment, but not to a sequence of such changes. The temporal coherence of lighting can be exploited much better when longer image sequences are considered which requires the knowledge of changes in the environment in advance. Those conditions are met for the off-line global illumination algorithms that are used in the final production of high quality animations.

Many existing off-line solutions resulted from extending hierarchical radiosity algorithms into the time domain, however, they involve huge storage costs which makes them impractical for complex scenes [4, 10]. In Global Monte Carlo Radiosity [1] the temporal coherence of costly visibility computations is efficiently and conservatively exploited, but then the radiosity solution is performed independently for each frame and all radiosity solutions are stored simultaneously in the memory. In the range-image framework [12], direct and indirect lighting are independently sampled in time for selected keyframes. The temporal frequency of sampling is adaptive to the changes in scene illumination. Interpolation is performed between the obtained solutions to derive lighting for inbetween frames, but some important lighting events between keyframes can be overlooked. To avoid this problem, sparse sampling of the scene illumination can be performed for all animation frames [11]. However, the mesh-based lighting reconstruction has many limitations in terms of rendering fine details in the lighting distribution such as caustics and indirect shadows.

The final gathering has been initially introduced in the context of the radiosity algorithm to overcome the problems arising with the mesh-based storage of lighting [14, 9, 19, 2]. Recently, some efficiency versions of the final gathering designed specifically for the hierarchical radiosity with clustering algorithm have been proposed [15, 16]. Final gathering based on the irradiance cache concept [20, 21] is also commonly used in the photon mapping algorithms because a direct rendering of diffuse illumination based on the density estimation of photons leads to poor image quality [7]. All those final gathering approaches can be used for the walkthrough animation, however, they are not suitable for the rendering of dynamic environments.

Recently, Martin et al. [10] proposed a final gathering algorithm in the framework of space-time hierarchical radiosity. Martin et al. classify the hierarchical radiosity links into the *good links* if the error of gathered lighting is within given bounds and the *bad links* otherwise. For the good links the final gather step is not required, and resulting lighting is accumulated in a texture using linear interpolation within a given patch. For each shooter polygon associated with a bad link the graphics hardware is used to estimate the visibility of a receiver patch texels using the projective shadows technique. In the temporal domain the bad links are classified as *static* and *dynamic*. The costly visibility computation is performed once for a given time interval for the static links, and is repeated for each frame for the dynamic links.

The final gathering method proposed by Martin et al. leads to significant rendering speedup (1.7–7 times in the examples given by the authors). However, the method shares typical drawbacks of hierarchical radiosity solutions such as poor handling of non-Lambertian surfaces and significant storage costs required by the link data structures. Those costs are even more significant in the presented solution because the history of links is also stored, e.g., links are not deleted when refined for possible reuse in different time intervals.

The problem of efficient handling the final gathering rendering in dynamic environments for photon-based algorithms was not addressed so far. In the following section we present our technique which is an attempt to fill this gap.

3 Algorithm

In our approach we use a two pass photon mapping algorithm [7]. In the first pass, photons are traced from light sources and stored in the photon map. In the rendering pass, the lighting computation is performed separately for direct illumination and glossy/specular reflection, diffuse indirect illumination, and caustics. The two first lighting components are computed for each frame from scratch using ray tracing. The diffuse indirect illumination is computed using the irradiance cache [20, 21]. Illumination values stored in the cache are computed through integration of the scene illumination, which is reconstructed from the photon map using the nearest neighbor density estimation technique. Such an integration is not performed to render direct

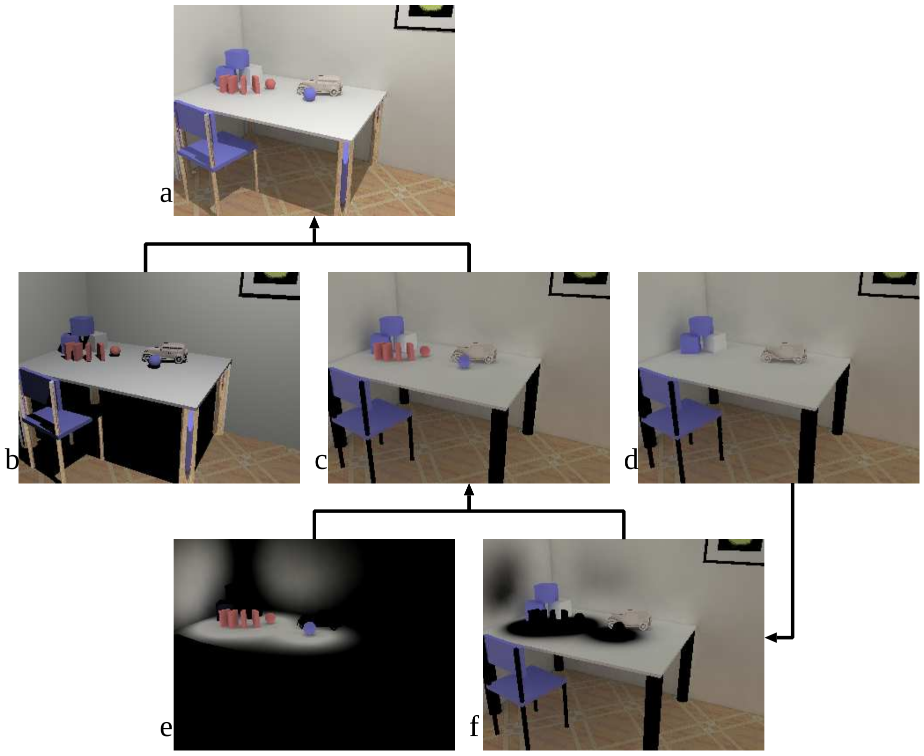

Figure 1: Processing flow in the computation of global illumination for an animation frame: a) the final frame, b) direct lighting, c) indirect lighting, d) static indirect lighting computed using the static irradiance cache, e) dynamic indirect lighting computed using the dynamic irradiance cache, and f) the dynamic indirect lighting computed through the photon density estimation and summed with the static indirect lighting which is shown in d).

caustics, which are directly reconstructed through density estimation of the caustic photon map.

In this work we focus on exploiting the temporal coherence of photons to speedup the costly irradiance cache computation and to improve the quality of indirect lighting reconstruction. We introduce the notion of *static irradiance cache*, which is computed once for an animation segment. For the static irradiance cache computation we remove all dynamic objects (i.e., objects changing their position, shape or light reflectance properties as a function of time) from the scene and we trace the so-called *static photons*.

The illumination component reconstructed from the static irradiance cache is perfectly coherent in the temporal domain and results in the flicker-free animations. However, the dynamic illumination component caused by dynamic objects must be also considered. For this purpose the *dynamic photons* which interact with dynamic objects are computed for each frame and are stored in a separate photon map. The map may store photons with negative energy [8], which are needed to compensate for occlusions of the static parts of the scene by dynamic objects. For example, the negative photons are stored in the regions of indirect shadows cast by dynamic objects. We describe the dynamic photons approach in more detail in Section 3.1.

Figures 1 illustrate the illumination reconstruction using our technique. Figure 1a shows the final animation frame whose lighting was composed from the direct illumination and specular reflection (Figure 1b) and the diffuse indirect illumination (Figure 1c). The dynamic component of the indirect lighting is reconstructed at two levels of accuracy depending on the influence of dynamic objects on local scene illumination. In the regions with the higher influence (we discuss the problem of detecting such regions in Section 3.3) the dynamic irradiance cache is recomputed, which leads to better accuracy of reconstructed dynamic lighting (refer to Figure 1e). In the remaining scene regions as shown in Figure 1f the illumination stored in the static irradiance cache (shown in Figure 1d) is corrected by adding its dynamic component reconstructed from the dynamic photon map using density estimation. A direct visualization of the dynamic illumination component is not shown because its values are rather small for the most parts of the scene and are negative in the regions occluded by dynamic objects. We blend lighting reconstructed using those two different methods (Figures 1e and 1f) to assure smooth transition of the resulting lighting.

While the static photons processing is performed in the same way as tracing ordinary photons for the static scenes [7], the dynamic photons are treated differently which we describe in the following section. In Section 3.2 we discuss our extensions of the irradiance cache to handle dynamic and static illumination components. We describe our animation rendering algorithm in Section 3.3.

3.1 Tracing Dynamic Photons

We introduce the dynamic photon map which is an extension to the photon maps to handle dynamic environments. It is leveraged to estimate the indirect illumination contributed only from dynamic objects (i.e., objects changing the position, orientation, shape, and material properties). The dynamic photon map stores dynamic photons which intersect with dynamic objects at least once.

Figure 2 illustrates the paths of two dynamic photons in the room with two dynamic objects, which are shown as an ellipsoid and a circle. The dynamic photons are traced from light sources toward the scene as in the traditional photon tracing approach [7]. When a photon hits on a dynamic object, the ray is marked as a dynamic ray, and then it is reflected or transmitted with the positive energy, or simply absorbed. In the first intersection with the dynamic object, a negative ray is spawned at the intersection point. This ray pierces the dynamic object and it is traced further as an usual ray with the only difference that it carries negative energy (in Figure 2 all negative rays are depicted using dashed lines). When a dynamic photon hits on a diffuse surface, it is stored in the dynamic photon map.

Special care is required in handling the direct photon paths from light sources to the scene. Since we separately compute the direct illumination, a dynamic photon directly hitting an object is not stored in the dynamic photon map to avoid doubling the computation of the direct illumination (refer to **points a**, **b** and **g** in Figure 2). Also, for the negative rays their intersection with dynamic objects must be ignored because their purpose is to subtract the energy from the static irradiance cache, which is computed for the scene without dynamic objects. For example, note that the ellipsoid is simply ignored on the way of the negative ray which travels between a pair of **points h** and **i**. Finally, note that the negative ray is not spawned at **point e** when the dynamic ray hits on the another dynamic object. This can be interpreted as the redirection of energy which in the case of static scene traveled along the path between **points b**, **c**, and **d** into the new path between **points a**, **e**, and **f** in the complete scene with dynamic objects.

The key point of construction of the dynamic photon map is to store only those photons which indirectly intersect with a dynamic object at least once.

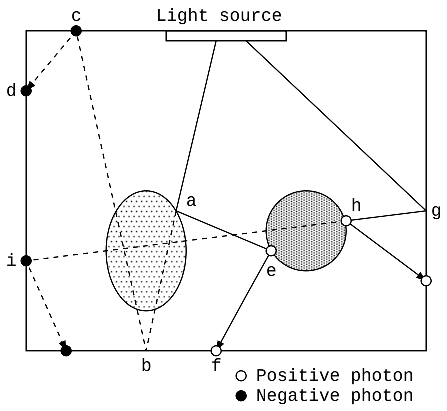

Figure 2: Tracing the dynamic photons: Example photon paths.

3.2 Static and Dynamic Irradiance Caches

We also extend the irradiance cache to handle dynamic environments. We use two (static and dynamic) irradiance caches for the efficient rendering of indirect illumination in dynamic environments. The static irradiance cache is computed for the scene with static objects only, i.e., all dynamic objects are removed. This cache can be computed only once for an animation sequence when lighting conditions do not change significantly. Otherwise, the animation must be split into shorter segments with coherent illumination. For example, a new static irradiance cache must be computed when a light is turned on in the dark room.

When the camera path is known in advance, which is usually the case for the off-line animation rendering, all values in the irradiance cache can be pre-computed. It is also possible to lazily reconstruct the static irradiance cache in the frame-by-frame manner. However, in such a case all photons for the static scene must be kept in the memory.

On the other hand, the dynamic irradiance cache is always lazily recomputed for each frame from scratch for those scene regions whose illumination changes significantly (e.g., for dynamic objects themselves and their neighborhood). The irradiance in the dynamic irradiance cache is computed using the global photon map of the current frame for the complete scene with static and dynamic objects. This approach is similar to the traditional irradiance cache technique with the only difference that in our approach the computation is not performed for the whole scene but rather for its selected regions. The problem of identifying such regions is discussed in detail in the following section.

Note that the sample locations in the dynamic irradiance cache are different for each frame, and they depend on the dynamic changes of illumination. By contrast, the sample locations in the static irradiance cache are the same for each considered animation segment.

Figure 3 shows the locations of sample points in the static and dynamic irradiance caches for the scene depicted in Figure 4.

3.3 Rendering

A practical two-pass rendering algorithm using photon maps is presented in [7]. In this algorithm

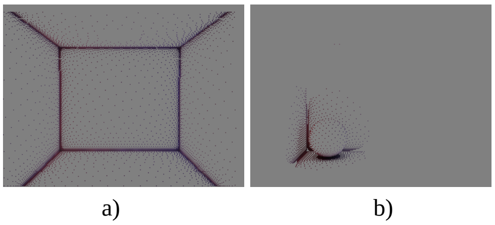

Figure 3: Locations of sample points for the a) static and b) dynamic irradiance caches.

the outgoing radiance L_r is computed as a sum of four terms: direct illumination L_d, soft indirect illumination L_i, caustics L_c, and specular component L_s:

$$L_r = L_d + L_i + L_c + L_s$$

In this work we focus on the computation of indirect illumination L_i, which is usually quite expensive to compute. We further split L_i into two components: L_y which is strongly affected by dynamic objects and L_t which is less affected by them. The choice between L_y and L_t depends on the influence I of dynamic objects on changes of illumination in the scene as follows:

$$L_i = \begin{cases} L_y & I > \tau_u, \\ f(g(I)) * L_y + & \tau_u > I > \tau_l, \\ \quad (1 - f(g(I))) * L_t & \\ L_t & \tau_l > I. \end{cases}$$

where τ_l and τ_u are the lower and upper threshold values for I, and $f(g(I))$ is a blending function between L_y and L_t in the transition scene regions. The blending function must be introduced to avoid discontinuities at lighting distribution due to inaccuracies in L_y and L_t estimations. In particular, the dynamic component of L_t which is computed using density estimation is prone for such inaccuracies. We use a cubic equation as the blending function $f(x)$ and the scaling function $g(x)$ to map the influence values I into the range from 0 to 1:

$$\begin{aligned} f(x) &= -2x^3 + 3x^2 \\ g(x) &= \frac{x - \tau_l}{\tau_u - \tau_l} \end{aligned}$$

The influence I is computed using the following density estimation equation based on the dynamic photon map:

$$I = \frac{1}{\pi r^2} \sum_{p=1}^{N} max(|\Delta\Phi_{p,r}|, |\Delta\Phi_{p,g}|, |\Delta\Phi_{p,b}|)$$

where $\Delta\Phi_p$ is a power of the photon p, and r, g and b denote the red, green and blue components in the power of p. The absolute value of photon energy is used because the dynamic photon map contains photons with both positive and negative energy.

L_y is computed using the dynamic irradiance cache as described in the previous section for the scene regions with $I > \tau_l$ and the dynamic objects themselves. Because such regions are strongly affected by dynamic objects, we use an accurate method for the lighting computation.

To compute L_t, first, all irradiances of the static irradiance cache are updated by adding the differential power estimated by the density estimation of the dynamic photon map. We apply the density estimation only in the positions of the static irradiance cache because we assume that in the scene regions less affected by dynamic objects we do not need to recompute cache positions. Then L_t is computed using interpolation of sample values stored in the updated static irradiance cache.

Figures 4 illustrate the concept of splitting the indirect illumination into two localized in the scene components L_y (Figure 4a) and L_t (Figure 4b). The final image (Figure 4c) is obtained by combining L_y and L_t with the direct illumination. The influence function I is shown in Figure 4c, where brighter regions correspond to higher values of I.

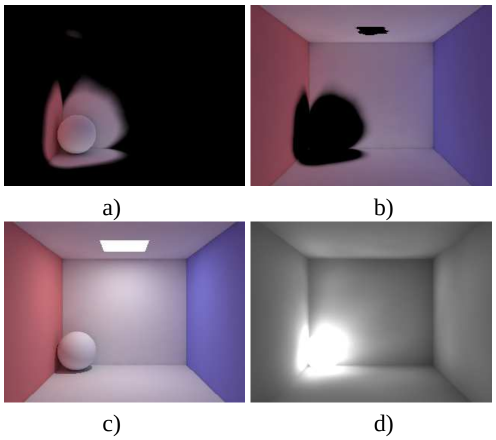

Figure 4: Decomposition of indirect illumination into a) L_y and b) L_t components, and c) the resulting image. The corresponding influence function I is shown in d).

4 Temporal Consideration

It is very important to reduce temporal aliasing in rendered animation frames because the human observer is very sensitive for such artifacts. The stochastic noise, which is inherent for Monte Carlo rendering, may result in intolerable degradation of animation quality if proper temporal antialiasing techniques are not used. The photon mapping approach usually eliminates successfully the high spatial frequency noise in the reconstructed lighting distribution. However, the remaining low frequency noise can be still quite objectionable in the context of animation.

We reduce the aliasing problem by introducing the concept of static irradiance cache, which means that the static part of illumination in the scene is perfectly coherent for the subsequent animation frames. For the dynamic illumination component we use two simple techniques applied for the L_y and L_t computation.

The dynamic irradiance cache used to derive L_y is computed by gathering incoming radiances, i.e., shooting a number of random rays into the scene. The noise in L_y can be substantially reduced by fixing the directions of gathering rays. This simple solution works well for the pseudo-random and stratified sampling.

Our second antialiasing technique deals with shooting dynamic photons. Because illumination changes in L_t are computed through density estimation of the dynamic photon map, ideally those photons should be coherent in the temporal domain as much as possible. One way to achieve this goal is to use quasi random walk for photon tracing. Then, the index of the quasi random sequence should be reset for each frame. Of course, the resulting photon paths are not exactly the same because the presence of dynamic objects in the scene but at least should be quite similar. Another solution is to process photons in the time domain as in [11].

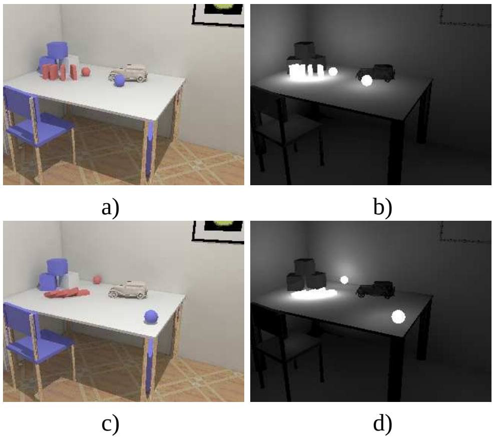

Figure 5: Example animation frames a) and c), and the corresponding influence I b) and d).

5 Results

We experimented with animation rendering for scenes BALL and TABLE (refer to Figures 4 and 5, respectively). The corresponding animations can be accessed on the Web[1].

Since the most time consuming part of the animation rendering is the irradiance cache computation we estimated the number of recomputed irradiance samples per frame (refer to Table 1). The reference solution recomputes the irradiance samples for all visible scene regions of each frame from scratch. On the other hand, our method recomputes them only for the region strongly affected by dynamic objects. Our method usually requires 3–4 times less irradiance samples per frame than the reference solution.

Table 2 compares the rendering time of indirect illumination per frame for our method and the reference solution for scene BALL. The frame resolution is 320×240 pixels. Our method needs extra 1.4 seconds for tracing dynamic photons as well as 1.4 seconds for tracing global photons (shown in the column PT). The computation time for the influence I is rather significant in our approach because we used a large number (300) of nearest photons for its estimation, and a large searching area for every pixel. The timing of L_y is much smaller than in the reference solution. Before computing L_t, we must update the static irradiance cache using density estimation but this updating takes only 1 second. Table 3 compares the timings for scene TABLE. In the experiment performed our method is 1.4 to 3.2 times faster than traditional approach for computation of indirect illumination.

[1] http://www.mpi-sb.mpg.de/~ tawara/research/vmv2002/vmv2002.html

	Reference	Our method
Scene1	6,719	1,543
Scene2	6,081	1,635

Table 1: The number of recomputed irradiance samples per frame.

	PT	L_i (sec.)		
	(sec.)	I	L_y	L_t
Our method	1.4 + 1.4	16	40	5.5
Reference	1.4	-	205	

Table 2: Scene BALL: Timings per frame.

	PT	L_i (sec.)		
	(sec.)	I	L_y	L_t
Our method	6.6 + 33	12	59	54
Reference	6.6	-	224	

Table 3: Scene TABLE: Timings per frame.

6 Conclusions

We presented an efficient technique for high-quality animation rendering. For this purpose we extended the photon mapping approach for dynamic environments. Significant speedup of the computation was achieved by localizing in the scene space costly recomputation of the irradiance cache. Also, the temporal aliasing was reduced by introducing the concept of static irradiance cache which can be reused across many subsequent frames until the scene lighting conditions do not change significantly.

References

[1] G. Besuievsky and X. Pueyo. Animating radiosity environments through the multi-frame lighting method. *The Journal of Visualization and Computer Animation*, 12(2):93–106, 2001.

[2] P.H. Christensen, D. Lischinski, E.J. Stollnitz, and D.H. Salesin. Clustering for glossy global illumination. *ACM Transactions on Graphics*, 16(1):3–33, 1997.

[3] C. Damez, K. Dmitriev, and K. Myszkowski. Global Illumination for Interactive Applications and High Quality Animations. In *State of the Art Reports*. Eurographics, 2002.

[4] C. Damez and F.X. Sillion. Space-Time Hierarchical Radiosity for High-Quality Animations. In *Eurographics Rendering Workshop 1999*, pages 235–246, 1999.

[5] G. Drettakis and F.X. Sillion. Interactive Update of Global Illumination Using a Line-Space Hierarchy. In *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 57–64, 1997.

[6] X. Granier and G. Drettakis. Incremental updates for rapid glossy global illumination. *Computer Graphics Forum*, 20(3):268–277, 2001.

[7] Henrik Wann Jensen. *Realistic Image Synthesis Using Photon Mapping*. A. K. Peters, Natick, MA, 2001.

[8] Henrik Wann Jensen and Niels J. Christensen. Efficiently Rendering Shadows Using the Photon Map. In *Compugraphics*, 1995.

[9] D. Lischinski, F. Tampieri, and D. P. Greenberg. Combining hierarchical radiosity and discontinuity meshing. In *Proc. of Siggraph'93*, pages 199–208, 1993.

[10] I. Martín, X. Pueyo, and D. Tost. Frame-to-frame coherent animation with two-pass radiosity. Technical Report (To appear in *IEEE Transactions on Visualization and Computer Graphics)* 99-08-RR, GGG/IIIiA-UdG, Jun 1999.

[11] K. Myszkowski, T. Tawara, H. Akamine, and H-P. Seidel. Perception-guided global illumination solution for animation rendering. In *SIGGRAPH 2001*, Annual Conference Series, pages 221–230, 2001.

[12] J. Nimeroff, J. Dorsey, and H. Rushmeier. Implementation and Analysis of an Image-Based Global Illumination Framework for Animated Environments. *IEEE Transactions on Visualization and Computer Graphics*, 2(4):283–298, 1996.

[13] X. Pueyo, D. Tost, I. Martin, and B. Garcia. Radiosity for Dynamic Environments. *The Journal of Visualization and Comp. Animation*, 8(4):221–231, 1997.

[14] M.C. Reichert. *A Two-Pass Radiosity Method to Transmitting and Specularly Reflecting Surfaces*. M.Sc. thesis, Cornell University, 1992.

[15] A. Scheel, M. Stamminger, and H.-P. Seidel. Thrifty final gather for radiosity. In *Rendering Techniques 2001: 12th Eurographics Workshop on Rendering*, pages 1–12. Eurographics, June 2001.

[16] A. Scheel, M. Stamminger, and H.-P. Seidel. Grid based final gather for radiosity on complex clustered scenes. Eurographics, 2002.

[17] F. Schoeffel and P. Pomi. Reducing Memory Requirements for Interactive Radiosity Using Movement Prediction. In *Eurographics Rendering Workshop 1999*, pages 225–234, 1999.

[18] F.X. Sillion and C. Puech. *Radiosity and Global Illumination*. Morgan Kaufmann, San Francisco, 1994.

[19] B. Smits. *Efficient Hierarchical Radiosity in Complex Environments*. Ph.D. thesis, Cornell University, 1994.

[20] G. Ward, Rubinstein F., and R. Clear. A Ray Tracing Solution to Diffuse Interreflection. In *Computer Graphics (SIGGRAPH 88 Conference Proceedings)*, 1988.

[21] G. Ward and P. Heckbert. Irradiance Gradients. In *Eurographics Rendering Workshop 1992*, 1992.