Low Memory Spectral Photon Mapping

Antoine Boudet[1,2], Mathias Paulin[1], Paul Pitot[2], and David Pratmarty[2]

[1]IRIT-UPS-CNRS, University Paul Sabatier, Toulouse, France
[2]OKTAL SE, Toulouse, France

Abstract

Photon Mapping [Jen01] *is a really interesting algorithm for simulating global illumination. Most of time, this method is used to render images in visible spectrum. In this paper, we show how to apply the method to a wider spectrum, in particular, for Infra-Red (IR) simulation. The major limitation of Photon Mapping method is the memory consumption. All surfaces are emissive in IR spectrum. Thus we have to use a huge number of photons to reduce noise. Furthermore, extension to spectral method also increases memory needs. We present a multi-pass photon map method which provides very accurate results using few memory resources which allows us to use as many photons as we need.*

Keywords:

Photon Mapping, Infra-Red, spectrum, low memory method

1. Introduction

Nowadays, image synthesis is used in many domains from video games to architecture. Another major field is Infra-Red simulation. IR permits to analyze a lots of thermal phenomena such as detecting hot points in engines, detecting problems of isolation in a house, viewing people in night. Most of time, image synthesis produces images in visible spectrum (RGB). We choose to adapt a global illumination technique to spectral IR rendering. First we will describe the Photon Mapping method [Jen01] which proposes a great number of advantages. Then we will show how it can be adapted easily to spectral rendering. Finally we propose a multi-pass Photon Mapping method to reduce memory cost without losing simulation accuracy.

Journal of WSCG, Vol.12, No.1-3., ISSN 1213-6972
WSCG'2004, February 2-6, 2004,Plzen, Czech Republic.
Copyright UNION Agency - Science Press

2. Background

Photon Mapping [Jen01] is an extension to Monte Carlo ray-tracing techniques [BLS94, VG95]. The goal of this global illumination method is to speed up computation times while providing the same advantages as Monte Carlo techniques. First, we will explain density estimation which is basis of the Photon Mapping method, then we will describe a Photon Mapping algorithm.

2.1. Probability Density Function

2.1.1. Definition

For a continuous function, the probability density function (PDF) is the probability that the variate has the value x. Since for continuous distributions the probability at a single point is zero, this is often expressed in terms of an integral between two points.

$$\int_a^b f(x)dx = Pr[a \leq X \leq b]$$

For a discrete distribution, the PDF is the probability that the variate takes the value x.

$$f(x) = Pr[X = x]$$

2.1.2. Importance sampling

PDF are mostly used in Monte Carlo integration to speed up computing time. They are used to choose sample points with a particular distribution. Convergence time decreases when the correspondence between the distribution and the function to integrate increases. In Photon Mapping, PDF can be used each time we have to choose an emission or reflection direction or to choose if a photon should be absorbed or reflected.

2.2. Photon Mapping concept

Photon Mapping consists in a two-pass method. First we propagate light flux from light sources into the scene and store illumination information in a specific data structure independent from the geometric model. Then we use a classic ray-tracing method to render the image using density estimation of the photon map.

Photon emission : Photons are created on the light sources located in the scene. Any type of light source can be handled. The power of all light sources is divided among all the emitted photons. A photon is a data structure which contains a fraction of the light sources power, a position and a propagation direction.

Photon Propagation : Photons emitted from the light sources are propagated into the geometric model with standard ray-tracing methods. Each time a photon has an intersection with a surface, the Russian roulette method is used to know if the photon is absorbed or reflected. Jensen proposed to store photons in different photon maps according to their path. It is useful to optimize particular effects such as caustics. For better comprehension, we assume that we use only one photon map and store photons each time they hit a surface whatever the nature of the materials. If the photon is reflected, the photon is modified to take into account the BRDF (Bidirectional Reflection Distribution Function) of the material. We can use importance sampling using this BRDF as a PDF to choose the reflection direction.

The photon map : Photons are stored at intersections in a balanced kd-tree [Ben75]. This data structure is separated from the geometric model. Thus no meshing is needed. We can handle complex scenes easily. The use of a kd-tree for storing photons allows a quick search of a certain amount of photons in the photon map.

2.3. Kernel density estimation

2.3.1. Principles

The kernel density estimation constructs an estimate $\hat{f}$ of an unknown density function f from a set of observed data points X_1, ..., X_n:

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right)$$

with h the bandwidth or smoothing parameter and K a kernel function [Sil86].

2.3.2. From photon density to radiance

According to the photon propagation algorithm, the probability $p(x)$ that a photon hits an object at a given point x is:

$$p(x) = \frac{1}{\Phi n} L_i(x)$$

with n the total number of photons hits, ϕ the mean value of photons power, and $L_i(x)$ the incident radiance at this point. Using the definition of kernel density estimation, we obtain an estimator of the incident radiance $L_i(x)$:

$$\hat{L}_i(x) = \frac{\phi}{h}\sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right)$$

2.4. Rendering : the radiance estimate

Once the photon map is built, we use its information to evaluate the rendering equation. Rays are traced in the scene from the eye. The radiance is computed at intersection of rays with the scene according to the rendering equation given by Kajiya [KH84]:

$$L_r(x,\vec{\omega}_r) = L_e(x,\vec{\omega}_r) + \int_{\Omega} f_r(\vec{\omega}_i,\vec{\omega}_r) L_i(x,\vec{\omega}_i)(\vec{n}_x \cdot \vec{\omega}_i) d\omega_i$$

Where L_e is the surface emitted radiance, L_i is the radiance coming from direction ω_i, f_r is the BRDF et Ω is the hemisphere of incident directions. The integral term is computed using a density estimation of the photons around the intersection point.

Jensen proposed to use the photon map only to compute indirect illumination and to use standard ray-tracing techniques to compute direct illumination. We will see in section 3 that we can evaluate illumination directly with the photon map using our method without degrading the accuracy of the method.

3. Spectral Photon Mapping

Our goal is to do IR rendering. We want to simulate all IR effects such as emission from all surfaces, scattering and caustics. We choose to develop a spectral renderer for the three next reasons:

- In IR, emissivity can change a lot with wavelength. Surface emission depends on Black Body Law, which give radiance function of temperature and wavelength.
- In gas, absorption and scattering phenomena are strongly spectral.
- Observations of IR phenomena are done with sensors. These sensors have a spectral response.

Photon Mapping is a very good technique to simulate scattering. This method is reliable for visible spectrum. We thought to adapt this method to spectral IR rendering. A basic method could be to divide the whole spectrum useful for the sensor in bands and associate one photon per band. But this method is very expensive, because we have to compute path for each photon, keeping in mind that the number of photons is proportional to the number of wavelength band of the sensor.

3.1. Spectral photons

We choose to divide the spectrum in bands of different widths to handle any spectral distribution. This repartition is done accounting spectral emission, spectral material reflection, gas absorption and gas scattering properties. In our method we define spectral photons that have a weight defined for each band of the sensor. This weight is stored in a vector. They still have a position and a direction as in classical method.

Photons emitted from light sources transport a part of the total spectral power of all light sources. Thus, all bands are propagated at the same time when the photon is traced into the geometric model, reducing paths computing time. In fact, the time used to compute paths is quite the same as in classical method.

3.2. Russian roulette

Because materials has spectral properties (a reflection coefficient for each band), we have to extend the Russian roulette method to take into account spectral reflections. The idea is the same as in the classical method [Jen01], except that we use average coefficient instead of the unique coefficient used for mono band rendering. For a given material, we can compute average diffuse and specular reflectance$\overline{\rho_d}$ and $\overline{\rho_s}$:

$$\overline{\rho_d} = \frac{1}{\Delta L}\int_{\Delta L} \rho_{d_l}\, dl$$

$$\overline{\rho_s} = \frac{1}{\Delta L}\int_{\Delta L} \rho_{s_l}\, dl$$

where ΔL is the sum of bands widths.

We use these values for Russian roulette. Given a random number ξ between 0 and 1 :

- if $\xi \in [0, \overline{\rho_d}]$ then diffuse reflection.
- if $\xi \in]\overline{\rho_d}, \overline{\rho_d} + \overline{\rho_s}]$ then specular reflection.
- if $\xi \in]\overline{\rho_d} + \overline{\rho_s}, 1]$ then absorption.

To account the fact that the reflection should have used a spectral reflectance value, we need to scale the power of the reflected photon. For choosing the reflection case, we used a PDF defined for each band b by :

- For a diffuse reflection : $PDF_d(b) = \overline{\rho_d}$.
- For a specular reflection : $PDF_s(b) = \overline{\rho_s}$.

Thus, photon weight needs to be modified to take into account importance sampling using the precedent PDF. For diffuse reflection and for each band b we have:

$$\Phi_{r,b} = \Phi_{i,b}\frac{\rho_{d,b}}{PDF_d(b)}$$

where $\Phi_{i,b}$ is the weight of the incident photon for band b and $\Phi_{r,b}$ is the weight of the reflected photon.

3.3. Spectral radiance estimate

Rendering is quite similar to classical Photon Mapping rendering. Actually, we trace rays from the eye into the model. These rays gather spectral radiance at their intersection with the scene in the model. The radiance is given by density estimation of the photon map. All radiance bands are computed simultaneously using vectors. For a given band b, we have :

$$L_{r,b}(x, \vec{\omega}_r) = \int_{\Omega} f_{r,b}(\vec{\omega}_i, \vec{\omega}_r) L_{i,b}(x, \vec{\omega}_i)(\vec{n}_x \cdot \vec{\omega}_i) d\omega_i$$

where $L_{r,b}$ is the reflected radiance for band b at x in direction $\vec{\omega}_r$. Ω is the hemisphere of incoming direction at point x, $f_{r,b}$ is the BRDF at x for band b and $L_{i,b}$ the incoming radiance for band b. The photon map contains informations on the flux. Using the relation between flux and radiance :

$$L_{i,b}(x, \vec{\omega}_i) = \frac{d^2\Phi_{i,b}(x, \vec{\omega}_i)}{(\vec{n}_x \cdot \vec{\omega}_i) d\vec{\omega}_i dA_i}$$

The radiance equation becomes :

$$L_{r,b}(x, \vec{\omega}_r) = \int_{\Omega} f_{r,b}(\vec{\omega}_i, \vec{\omega}_r)\frac{d^2\Phi_{i,b}(x, \vec{\omega}_i)}{dA_i}$$

The incoming flux $\Phi_{i,b}$ is approximated using the photon map by locating the n nearest photons to x. We have:

$$L_{r,b}(x, \vec{\omega}) \approx \sum_{p=1}^{n} f_{r,b}(x, \vec{\omega}_p, \vec{\omega})\frac{\Delta\Phi_{p,b}(x, \vec{\omega}_p)}{\Delta A}$$

where $\Delta\Phi_{p,b}$ is the power of the photon p for band b, and ΔA is the surface where we found the n photons used to estimate the radiance.

4. Low memory multi-pass method

Our spectral method could use large amounts of memory if we use a lot of wavelengths. Furthermore, while in IR all surfaces are emissive, photons emission positions are spread on all surfaces of the model. Thus, photon impacts are spread too on all surfaces of the model, generating a lot of noise. In particular, for specular materials, we need to use an important number of photons in the specular cone to have an accurate density estimation. Raising the number of photons in the photon map needs lots of memory. It is quite impossible to match sufficient accuracy because of memory requirements of the classical photon mapping method. Figure 1 shows a comparison between the rendering of an emissive and specular tube with few photons (left) and enough photons (right).

We need to throw a huge number of photons to reduce noise. But we haven't unlimited memory on our computers.

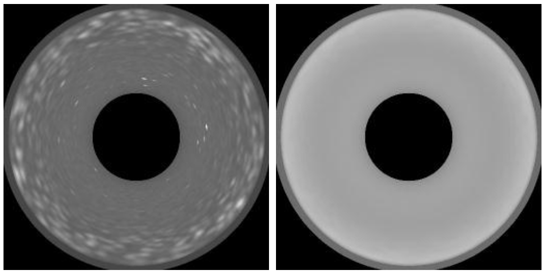


Figure 1: *Rendering of an emissive and specular tube with one million photons (left) and approximatively one hundred million photons in the photon map(right).*

So, we thought of an extension of the method. We propose a multi-pass method which prevents to use lots of memory, and provides a very good accuracy.

4.1. Basic idea

Photon Mapping is based on density estimation. Thus, accuracy of the results raises with the number of samples (photons) used. While we want a great accuracy, we need to store a high number of photons. We have to find a strategy to raise the number of photons without consuming memory. The first idea to do so is to cache the photon map on disk. But then we have to take into account the save and load times from the disk. Actually, for each radiance estimate, we need to load all parts of the photon map, which cost a lot in terms of disk access.

So we choose to use little photon maps which fit the memory to compute temporary radiance instead of caching the photon map. This radiance is stored in an image, which is easier to cache than a big photon map.

4.2. Using little photon maps

Larsen and Christensen [LC03] showed that computing indirect illumination using several little photon maps is faster than computing with only one. A photon map is generated using many random choices. This is done using a pseudo-random numbers generator. Thus we thought about generating several photon maps using several distinct seeds and use them one after the other to reduce noise in the image.

In fact, we construct one little photon map per pass using a seed for the random numbers generator. Then, this photon map is used to render a temporary image. This image presents a strong noise because it was generated using a low precision photon map. We choose to compute little photon maps because they are faster to compute than huge photon maps and because the search of photons is faster in little photon maps. Using a balanced kd-tree to represent the photon map guarantees that search time for M photons in a N photons photon map is $O(M\log_2(N))$. The smaller is N, the smaller is search time. Furthermore, little photon maps always fit in memory.

4.3. Final image reconstruction

We assume that an image corresponds to a set of sample points which are the same in each pass. Actually, a pixel of the image defines a unique ray from the eye, and thus defines a unique intersection point in the scene. So, we don't need to recompute the points where radiance is computed from one pass to another. We saw that the radiance in this point is given by:

$$L_{r,b}(x,\vec{\omega}) \approx \sum_{p=1}^{n_r} f_{r,b}(x,\vec{\omega}_p,\vec{\omega}) \frac{\Delta\Phi_{p,b}(x,\vec{\omega}_p)}{\Delta A}$$

where n_r is the number of photons used for density estimation.

Assuming that light power for band b, Φ_b, is divided equally among all the photons, we have:

$$\Delta\Phi_{p,b}(x,\vec{\omega}_p) = \frac{\Phi_b}{N_t}$$

where N_t is the total number of photons in the Photon Map.

Combining with radiance expression gives:

$$L_{r,b}(x,\vec{\omega}) \approx \sum_{p=1}^{n_r} f_{r,b}(x,\vec{\omega}_p,\vec{\omega}) \frac{\Phi_b}{N_t \Delta A}$$

Assuming that an integer m exists such as :

$$N_t = mN$$

and assuming that :

$$n_r = mn$$

Then we have :

$$L_{r,b}(x,\vec{\omega}) \approx \sum_{p=1}^{n\dot{m}} f_{r,b}(x,\vec{\omega}_p,\vec{\omega}) \frac{\Phi_b}{Nm\Delta A}$$

We can cut the sum in m terms :

$$\begin{aligned} L_{r,b}(x,\vec{\omega}) \approx \Bigg[& \sum_{p=1}^{n} f_{r,b}(x,\vec{\omega}_p,\vec{\omega}) \frac{\Phi_b}{Nm\Delta A} \\ & + \sum_{p=n}^{2n} f_{r,b}(x,\vec{\omega}_p,\vec{\omega}) \frac{\Phi_b}{Nm\Delta A} \\ & + \dots \\ & + \sum_{p=n(m-1)}^{nm} f_{r,b}(x,\vec{\omega}_p,\vec{\omega}) \frac{\Phi_b}{Nm\Delta A} \Bigg] \end{aligned}$$

We do a variable change for each sum and we assume that for each sum, photons $\vec{\omega}_{p_i}$ are in photon map i and that photon maps are distinct. Then we have:

$$\begin{aligned} L_{r,b}(x,\vec{\omega}) \approx \Bigg[& \sum_{p_1=1}^{n} f_{r,b}(x,\vec{\omega}_{p_1},\vec{\omega})\frac{\Phi_b}{Nm\Delta A} \\ & + \sum_{p_2=1}^{n} f_{r,b}(x,\vec{\omega}_{p_2},\vec{\omega})\frac{\Phi_b}{Nm\Delta A} \\ & + \dots \\ & + \sum_{p_m=1}^{n} f_{r,b}(x,\vec{\omega}_{p_m},\vec{\omega})\frac{\Phi_b}{Nm\Delta A} \Bigg] \end{aligned}$$

We can write this as a sum:

$$L_{r,b}(x,\vec{\omega}) \approx \sum_{i=1}^{m} \sum_{p_i=1}^{n} f_{r,b}(x,\vec{\omega}_{p_i},\vec{\omega})\frac{\Phi_b}{Nm\Delta A}$$

We factorize:

$$L_{r,b}(x,\vec{\omega}) \approx \frac{1}{m} \sum_{i=1}^{m} \sum_{p_i=1}^{n} f_{r,b}(x,\vec{\omega}_{p_i},\vec{\omega})\frac{\Phi_b}{N\Delta A}$$

The term $\sum_{p_i=1}^{n} f_{r,b}(x,\vec{\omega}_{p_i},\vec{\omega})\frac{\Phi_b}{N\Delta A}$ is a density estimation searching n photons in a photon map with N photons.

The equation shows that computing the radiance with m little photon maps of N photons using n photons for density estimation then computing a mean value is the same as computing radiance with one big photon map of N_t using n_r photons for density estimation.

Using little photon maps avoids to cache the photon map on the disk, and so avoids disk access time. We choose to compute this photon maps one after the other in a multi-pass method. In fact, each new pass corresponds to a raise of the number of photons used for density estimation.

While we use a multi-pass method, we can see the evolution of the image and stop the simulation when the accuracy is sufficient. This could be automated by measuring the difference between the last two pass results.

Many Photon Mapping algorithms compute only indirect illumination using photon maps. Actually, we need a lot of photons to compute direct illumination by density estimation without noise. While we can use as many photons as we need with our method, we choose to compute all kind of illumination using density estimation.

5. Results

5.1. System

We used the software SPECRAY from OKTAL Synthetic Environment on an Athlon 2.4 GHz with 512 Mo running Linux to obtain all results.

5.2. Tests

An emissive specular ring lies on a totally diffuse planar surface. A point light is added that generates caustics in the ring. This scene isn't closed, so we need to throw more photons than we want to store because we lost a lot of photons in the atmosphere. That's why we need to throw approximatively 30,000,000 photons to store 3,000,000 photons. We have faster computing time in closed scenes because we need to throw less photons.

Figure 2 presents time results for rendering the scene shown in figure 3 and 4. Figure 3 shows two images, one computed with a photon map of 3,000,000 photons using 600 photons for density estimation, the other with 30 passes computing photon maps of 100,000 photons using 50 photons for density estimation.

Figure 4 shows IR images rendered at different step of computing. We used photon maps of three million photons and 500 photons for density estimation to render this images because it's the maximum our system can handle.

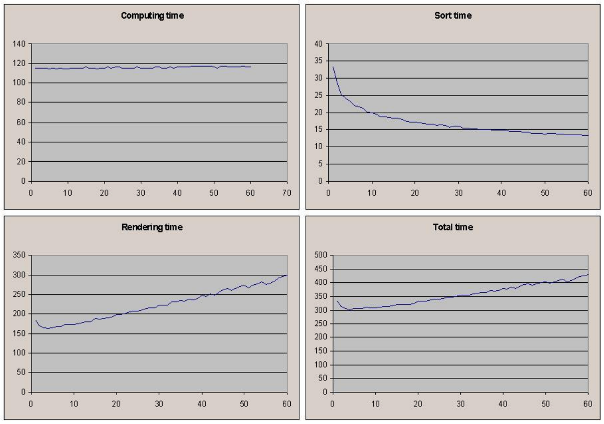
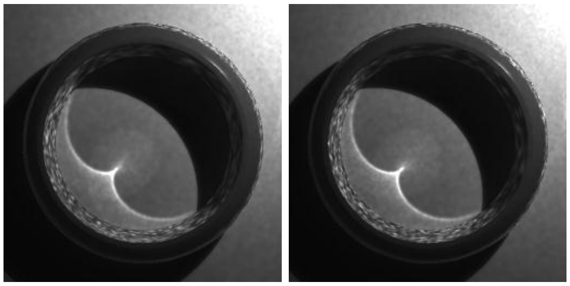


Figure 2: *Computing, sorting, rendering and total times (s) for 3,000,000 photons using different numbers of passes (1 to 60). For m passes, each photon map contains* $3,000,000/m$ *photons, and we search* $600/m$ *photons for density estimation. The original size of images is 256x256 pixels.*

5.3. Discussion

In section 4.3, we showed that computing radiance using a photon map of N_t photons using n_r photon for density estimation gives the same results as computing the mean value of m evaluations using a photon map of N photons using n photons for density estimation assuming that $N_t = mN$ and $n_r = mn$. Figure 2 shows that computing times for one big photon map or m little photon maps are equal. Sorting time decreases with the number of photons in the photon map. Rendering time for one photon map decreases with the number of photons in the photon map and with the number of



	Left	Right
Pixel mean value	1.39736	1.39929
Pixel standard deviation	1.19078	1.19402

Figure 3: *IR renderings with one (left) and thirty (right) passes.*

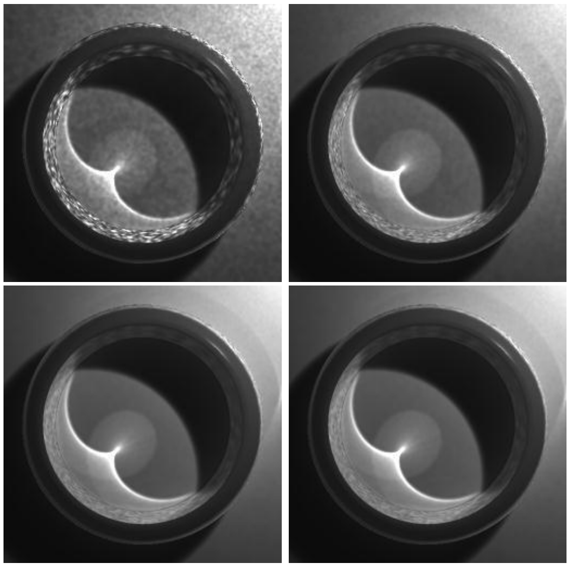


Figure 4: *Several steps IR images. All passes are done using three million photons. From left-up image to right-down image, respectively 1, 5, 20 and 250 passes were used. 500 photons were used for radiance evaluation.*

photons used for density estimation. Nethertheless rendering time for *m* photon map increases when the time saved using little photon maps becomes smaller than the time needed to update the image. Our method rendering is faster than classical method for a judicious number of passes. For example, the best is to use five passes for this scene. This parameter change for others scene.

The two images from figure 3 have the same variance. In fact, the only difference is that the photons used for density estimation are not the same, because we use random sampling for photon map generation.

Figure 4 shows that rendering quality for 3 million photons is far from good. Much noise is visible in the left-up image. Quality increases fastly when we add a few pass. We saw that this quality raises slowly with a great number of pass. This come from the fact that we do a mean image. We have to double the number of passes to divide noise by two.

6. Conclusion

We showed how it is possible to adapt Photon Mapping to a spectral rendering without consuming extra memory while keeping a very good accuracy. Taking into account spectral emission and reflections permitted us to do accurate IR rendering. Our multi-pass method is faster than classical method when photon map fits entirely in memory. We can use an unlimited number of photons. We avoid problems due to photon map caching such as swapping and cache defaults. Thus our method can match very accurate results. This is very interesting for IR rendering, while we can through enough photons to have the noise disappeared.

7. Coming soon...

While we use a Photon Mapping method, we keep all advantages of the method. So far, we have not implemented all functionalities of Photon Mapping such as filtering in angles and use visual importance [KW00, PP98]... Thus we will have to upgrade our spectral Photon Mapping to have all possible advantages.

While we use a multi pass method, we could optimize convergence time by a better construction of the photon map. Actually, the photon map of a pass could be analyzed and give importance information useful to lead the construction of the next pass photon map. Our method is already faster and more accurate than classical method and could be optimized easily.

In classical method, we use a kd-tree to store photons. It provides an interesting compromise between search time and memory consumption. While we are not limited by memory, we could use another data structure which provides better search time than a kd-tree.

References

[Ben75] Jon L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[BLS94] P. Blasi, B. LeSaëc, and C. Schlick. An importance driven monte-carlo solution to the global illumination problem. *5th EG Workshop on Rendering*, pages 173–183, June 1994.

[Jen01] Henrik Wann Jensen. *Realistic Image Synthesis Using Photon Mapping*. A K PETERS LTD edition, 2001.

[KH84] J.T. Kajiya and B.P. Von Herzen. Ray tracing volume densities. *Computer Graphics*, 18(3):165–174, July 1984.

[KW00] Alexander Keller and Ingo Wald. Efficient Importance Sampling Techniques for the Photon Map. In *Vision Modelling and Visualization 2000*, pages 271–279, Saarbruecken, Germany, November 2000.

[LC03] Bent Dalgaard Larsen and Niels Jorgen Christensen. Optimizing photon mapping using multiple photon maps for irradiance estimates. *WSCG POSTER proceedings*, February 2003.

[PP98] Ingmar Peter and Georg Pietrek. Importance driven construction of photon maps. In G. Drettakis and M. Nelson, editors, *Rendering Techniques '98 (Proceedings of the 9th Eurographics Workshop on Rendering in Vienna, Austria)*, pages 269–280. Springer, 1998.

[Sil86] B. W. Silverman. Density estimation for statistics and data analysis. *Chapman and Hall, London and New York*, 1986.

[VG95] E. Veach and L. Guibas. Optimally combining sampling techniques for monte carlo rende-ring. *Computer Graphics Proc. Annual Conf. Series: SIGGRAPH '95*, pages 419–428, 1995.