Multi-pass Density Estimation for Infrared Rendering

Antoine Boudet[†♯], Mathias Paulin[♯], Paul Pitot[†] and David Pratmarty[†]
[♯]IRIT-UPS-CNRS, 118 Route de Narbonne, 31062 Toulouse, France
[†]OKTAL SE, 2 Rue Boudeville, 31000 Toulouse, France

Abstract

The density estimation methods are known to be among the most promising for providing realistic images from 3D scenes. However, in most of these methods, the direct illumination is computed using a raytracing pass which samples each light source. This involves a limitation on the number of light sources that can be handled. This limitation can be removed using a direct estimation of radiance using the particle map. Nevertheless, in this case we would need to store a huge number of particles in order to have a sufficient density in the particle map, which involves a limitation on memory consumption. In this paper we propose a new technique called "multi-pass density estimation" which is independent of the scene complexity and which maintains the memory consumption constant independently of the number of particles used for density estimation, and hence, removes the limitation on their number. This method has been developped in the context of infrared rendering where any surface or participating medium is emissive and where the absorption and scattering phenomena are strongly spectral. Therefore we had to adapt a band model to the density estimation method in order to handle spectral complexity. To reduce memory usage, we make a partition of the big particle map in parts of a given size. Each particle map provides a density estimation. We finally take the average value of these estimations. We also demonstrate the efficiency of our method in the visible spectrum, which is in fact a simplified application of our technique.

1. Introduction

A lot of work has been done on realistic rendering. For the majority of existing techniques, it is difficult to handle scenes with a huge number of extended light sources. As our goal is to do infrared rendering, we do have to handle all surfaces of the scene as extended light sources. In most of the existing density estimation techniques, a raytracing pass is used for direct lighting computations and density estimation is only used for indirect illumination computation. In the case of infrared rendering, the raytracing pass cannot be handled because it is prohibitive to sample efficiently all the sources (all the surfaces) of the scene. Thus we chose to compute the whole illumination using density estimation. But it appeared that we needed a huge number of particles to handle direct illumination correctly and this number can't be stored in nowadays computer memory.

Our research to solve infrared simulation problems leds us to a multi-pass density estimation method which allows us to render 3D scenes containing a lot of light sources without increasing the memory consumption. Actually, rendering visible scenes is a simplified application of our infrared approach which can handle all surfaces and participating media as light sources. The memory consumption of the method is independent of the scene complexity and of the number of particles used to render the final image.

Firstly, we present the problems which has to be solved for infrared rendering. Secondly, an overview of global illumination techniques explains our choice to improve the density estimation method for infrared rendering. Then we expose our spectral method, the memory limitation that arises and the approach we found to overcome this problem. Finally, we test our method and discuss the results.

2. Infrared constraints

Nowadays, infrared sensors enable a lot of thermal phenomena to be analyzed such as detecting hot points in engines, detecting isolation problems in a house, night observation, etc. We want to simulate these phenomena to make the engineers' work easier. For example, if we can simulate an engine before it has been built, we can predict a lot of design failure and so save a lot of time and money. The three major differences between infrared and visible spectrum are described below.

First of all, for wavelength in the visible spectrum, only very hot materials are emissive (light sources), whereas for infrared spectrum, every object can be a light source because emissivity is more sensitive to temperature for these wavelengths. The emitted radiance of a material is modelized by the Planck's Black Body Law. This law gives the emitted radiance in function of the wavelength and the temperature. Figure 1 presents the Planck's law. For instance, a table at 293 Kelvin emits at infrared wavelength and not at visible ones.

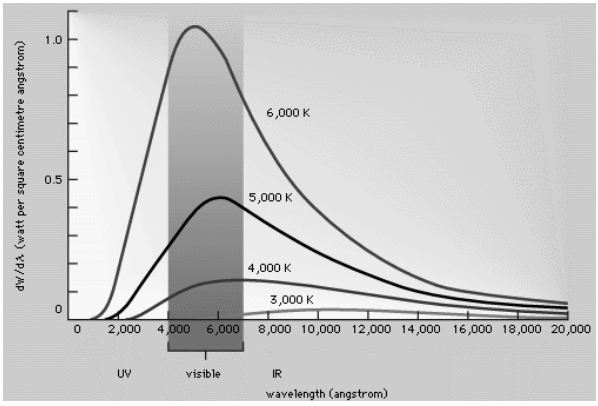


Figure 1. Planck's Black Body Law. Each curve gives the intensity emitted by a black body in function of wavelenght for a given temperature.

Secondly, infrared spectral value can change quickly with the wavelength due to absorption phenomena.

Finally, participating media cannot be neglected in infrared rendering. Actually, absorption and scattering phenomena are strongly spectral for infrared wavelengths. Furthermore, gases in a scene are most of the time at ambient temperature, thus these gases are emissive.

Consequently, we need a method that can handle all surfaces and participating media as light sources and that can deal with the spectral complexity of infrared phenomena.

3. Background

A lot of visible rendering methods already exist. Let us see if one of them could help us in our aim. We thought about global illumination rendering techniques because we need physical and accurate results.

3.1. Radiosity

Among all global illumination methods, radiosity is the first. It offers good computation times for simple scenes (i.e. lambertian surfaces and simple geometry). Nevertheless the cost of this method becomes prohibitive for complex scenes. Simple information about radiosity can be found in [20]. A lot of work has been done in order to reduce the complexity of radiosity algorithm such as hierarchical radiosity [2] and clustering [7]. However, the complexity problem is not completly solved using these methods. As we want to render very complex scenes (materials and geometry) with a great physical accuracy, we thought about Monte Carlo ray tracing techniques.

3.2. Monte Carlo Ray Tracing techniques

These methods use point sampling to estimate the illumination in a model. A point sample consists in tracing a ray through the scene and computing the radiance in the direction of this ray. In Monte Carlo Path Tracing [10], rays are sent out from the virtual eye point through a pixel, whereas in Monte Carlo Light Tracing [3] rays are traced from the light sources. Therefore the random walks are independent of the pixels. Instead of considering each pixel at a time, all pixels can be treated at the same time. Many improvements of these methods have been proposed such as stratified and importance sampling. So far, for a complete overview of these improvements, see [18, 11, 4, 22]. These techniques have several advantages. Any kind of geometry with any kind of material can be handled without tessellation. Ray tracing is the best way for specular reflections. The accuracy is controlled at the pixel level using the variance and these methods are unbiased. Unfortunately these methods have a slow convergence because all possible paths have to be traced for rendering. Bidirectional Path Tracing [12, 23] combines path and light tracing. Paths are traced starting from both the eye and the light sources. The idea is to exploit the fact that certain paths are most easily sampled from the eye whereas other paths can be sampled better by starting at the light. This method takes into account all possible paths, and caustic paths in particular. But the problem of this technique is the noise. Actually, the cost to reduce the noise is very expensive, particularly when dealing with mirror reflections for which Path Tracing is a better solution.

3.3. Particle Tracing and Density Estimation methods

The computation of global illumination using density estimation provides the same advantages as Monte Carlo ray tracing techniques. Furthermore, the technique is consistent, which means that it converges to the correct solution as more points/particles are used. This method is performed in two passes. First, the flux is propagated from the light sources in the scene using particle tracing and stored into the particle map. Then a density estimation [21, 15, 19, 24] is used to estimate the radiance from the flux information stored in the particle map. Estimation points are found using a simple ray tracing from the eye into the model. Jensen [9] proposed using density estimation only for computing indirect illumination. Direct illumination is computed using a Monte Carlo ray-tracing pass. For each intersection point, a ray is traced towards each source. However, the number of rays to be traced becomes prohibitive for a large number of light sources. On the other hand, if we compute both direct and indirect illumination using a density estimation, the memory consumption becomes prohibitive because a high density of particles has to be present in the map in order to reduce noise, and hence, a large mass of data has to be stored. Thus memory is limiting the accuracy of direct estimation.

Consequently, density estimation appears to be the most interesting technique for physical rendering. But some problems remain. The next section presents our density estimation approach for spectral infrared rendering and shows that it increases memory consumption. Then section 5 presents our solution to free us from the memory limitation.

4. Spectral density estimation

4.1. Spectral model

The origin of our work was to improve the spectral ray tracing software SPECRAY provided by OKTAL SE. Therefore, we had to use its spectral band model to represent the spectrum. It consists in cutting the wavelength domain in little segments called bands. For each band, the spectrum value is defined for each bound of the band. The value for the band can be constant or linearly varying between the bound values. Then we have an approximation of the spectrum (piecewise constant or linear). Any kind of infrared spectrum can be handled by this model. Even absorption rays can be represented using very narrow bands.

Much work has already been done on spectral models [8, 17, 1] but optimizing the representation of the spectrum is not the point of this article. Furthermore, these models are only used for visible simulation, and hence, take into account the human vision system and the human perception which is pointless for infrared rendering. An efficient spectral model is the k-distribution model. For more information see [6]. But, in any case, the memory problems would have been the same even if we had chosen to use one more efficient spectral model.

4.2. Spectral particles

In [9], a particle is a data structure containing a position, a direction, and RGB values of the light flux. Similarly, we use particles containing a position, a direction and a spectral flux represented by an array of values. Each element of the array corresponds to a band of the approximated spectrum. We could have used one particle per band but we would need a great number of bands to accurately represent the spectrum. For example, we need at least 10000 bands for certain sort of gas. In this case, if we have one band per particle, we have to compute a huge number of particle paths, and we have to store a huge number of positions and directions (particles). As the complexity of sorting the particle map and searching particles raises with the number of particles in the map, we choose to reduce the number of particles stored by transporting a whole spectrum per particles. Furthermore, Larsen and Christensen showed, for a different aim, that using multiple small particle maps is faster than using one big particle map [13].

4.3. Spectral reflections

Materials have spectral properties. These properties are described using the same bands as the spectrum of the particle. The particle tracing pass uses the Russian roulette method as in [9] whereas it must be adapted to spectral rendering. We use an 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 $\overline{\rho_s}$:

$$\overline{\rho_d} = \frac{1}{\Delta\Lambda} \int_{\Delta\Lambda} \rho_{d_\lambda} d\lambda$$

$$\overline{\rho_s} = \frac{1}{\Delta\Lambda} \int_{\Delta\Lambda} \rho_{s_\lambda} d\lambda$$

where $\Delta\Lambda$ is the sum of bands widths.

These values are used in the 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 for the fact that the reflection type should have been chosen using a spectral reflectance value, we need to scale the spectral power of the reflected particle according to the chosen case. In fact, the choice of the reflection case uses a probability density function (PDF) [5] defined for each band $\Delta\lambda$ by :

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

Thus, particle weights need to be modified to take into account importance sampling using the precedent PDF. For diffuse reflection and for each band $\Delta\lambda$ we have:

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

where $\Phi_{i,\Delta\lambda}$ is the power of the incident particle for band $\Delta\lambda$ and $\Phi_{r,\Delta\lambda}$ is the power of the reflected particle.

4.4. Spectral radiance estimate

Our rendering method differs a little from classical density estimation methods. Rays are traced from the eye into the model. These rays gather spectral radiance at their intersection with the scene in the model. We cannot use the optimization that consist to trace a random ray to each light source to compute direct illumination because all surfaces of the model are light sources. If we do so, we have to trace rays in all directions, which is prohibitive. Thus, the radiance is directly given by the density estimation of the particle map at the first intersection point of the ray with the model. This implies having a sufficient density of particles in the map. All radiance bands are simultaneously computed using arrays. For a given band $\Delta\lambda$, we have adapted the rendering equation :

$$L_{r,\Delta\lambda}(x,\vec{\omega_r}) = \int_\Omega f_{r,\Delta\lambda}(\vec{\omega_i},\vec{\omega_r}) L_{i,\Delta\lambda}(x,\vec{\omega_i})(\vec{n_x}\cdot\vec{\omega_i})d\omega_i$$

where $L_{r,\Delta\lambda}$ is the reflected radiance for band $\Delta\lambda$ at x in direction $\vec{\omega_r}$. Ω is the hemisphere of incoming direction at point x, $f_{r,\Delta\lambda}$ is the BRDF (Bidirectional Reflection Distribution Function) [16] at x for band $\Delta\lambda$ and $L_{i,\Delta\lambda}$ the incoming radiance for band $\Delta\lambda$. The particle map contains information on the flux. Then using the relation between flux and radiance :

$$L_{i,\Delta\lambda}(x,\vec{\omega_i}) = \frac{d^2\Phi_{i,\Delta\lambda}(x,\vec{\omega_i})}{(\vec{n_x}\cdot\vec{\omega_i})d\vec{\omega_i}dA_i}$$

The radiance equation becomes :

$$L_{r,\Delta\lambda}(x,\vec{\omega_r}) = \int_\Omega f_{r,\Delta\lambda}(\vec{\omega_i},\vec{\omega_r}) \frac{d^2\Phi_{i,\Delta\lambda}(x,\vec{\omega_i})}{dA_i}$$

The incoming flux $\Phi_{i,\Delta\lambda}$ is approximated using the particle map by locating the n nearest particles to x. We have:

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

where $\Delta\Phi_{p,\Delta\lambda}$ is the power of the particle p for band $\Delta\lambda$, and ΔA is the area of the surface where we found the n particles used to estimate the radiance.

4.5. Problems due to spectral density estimation for infrared rendering

4.5.1. Specular noise After making a few tests, some problems appear. Figure 2 is the infrared rendering of a tube with a specular and emissive material using one million particles. The tube is the only light source in the scene. The image is noisy because there are not enough particles to estimate radiance for specular materials. Actually, when the radiance is estimated, the power of the particles is multiplied by the BRDF value. As a consequence, a particle with its direction in the specular cone is more important than a particle with its direction in the diffuse part of the BRDF. The probability to have particles in the specular cone depends on its width and on the particle density around the sample point. Since we cannot change the material properties, our only choice is to raise particle density. Thus we have to trace and store more particles. Figure 3 illustrates the specular problem.

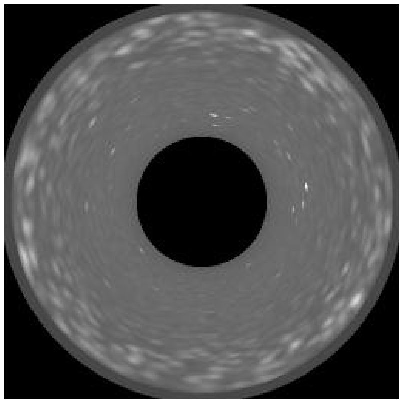
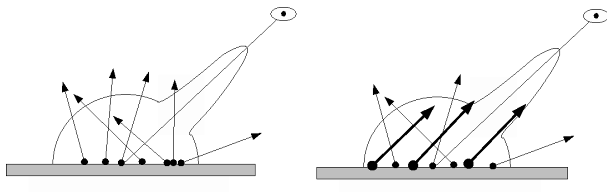


Figure 2. Rendering of an emissive and specular tube using one million particles.



(a) (b)

Figure 3. Estimation problem for specular materials. (a): no particle has its direction in the specular cone. (b): particles in bold are in the specular cone.

4.5.2. Memory cost Therefore, storing more particles has a cost. Let us count the size of a spectral particle. First, the position and the direction are stored using 6 *double*. Then each band power is stored using 1 *double*. Thus if we have only one band, the particle size is 7 *double* or 54 bytes. This means that we need 54 megabytes to store 1 million particles. But these results are only for one band. If the number of bands is 100 then we need 848 bytes per particle, then for 1 million particles we need 848 megabytes. This is prohibitive for the majority of the computers. Hence, we have to find a solution to raise the number of particles for density estimation without using so much memory. Of course, if we use *float* instead of *double*, we can divide the memory consumption by two. We can use Ward's shared-exponent RGB-format as in [9], but as we need to throw more than 20 million of particles, the memory problem remains.

5. Multi-pass method

5.1. Objectives

As we said in the previous section, our objective is to find a way to raise the number of particles stored in the particle map. But if the number of particles and/or the number of bands is too large, we do not have enough memory to store the entire particle map. Thus we have to cache the particle map on the disk. And we want to keep up similar performances. So our objective is to optimize particle map cache management.

5.2. Existing technique

Ward et al. proposed Irradiance caching [25] in 1988. They only use the density estimation to compute indirect illumination. Thus, the idea is to take advantage of the slow changes in the indirect illumination. They can precompute irradiance for a certain number of sample points. The number of points depends of the illumination regularity. Actually, they use an estimate of the illuminance gradient to know if the sample points density is sufficient or if they have to add some points. Then the irradiance is interpolated during the rendering. This method is very efficient for a majority of scenes and could probably be adapted to infrared rendering. However, there is three type of infrared rendering using short (from $1e^{-6}$ to $2e^{-6}$ meters), medium (from $3e^{-6}$ to $5e^{-6}$ meters) and long (from $8e^{-6}$ to $12e^{-6}$ meters) wavelenghs. Irradiance caching is suited for medium and long wavelengths because the direct illumination changes slowly due to the emission of all materials. Nevertheless, in short wavelength rendering, the direct illumination can be irregular as in visible spectrum. For this kind of rendering, we cannot use this method because we compute the whole illumination using direct density estimation of the particle map and direct illumination can change quickly then the hypothesis of slow illumination changes isn't verified. As direct illumination can change quickly, the number of sample points to use can be very big depending on the scene.

Shirley and Walter [19, 24] proposed a view independent method by caching the illuminance using a mesh. A particle tracing phase then a density estimation are used to compute the illumination over the scene mesh. Then a decimation phase is used to produce an illumination mesh where the illumination can be considered linear over each face of the mesh. Then the illuminated mesh can be displayed using standard OpenGL output. As the illuminated mesh depend on the scene complexity and the illumination regularity, this method is dependent on both of them.

For both of these methods, cases exist where these methods are dependent on the scene complexity and the illumination regularity . For example, a scene with a huge number of small and complex objects (typically an engine) may have a irregular illumination which is a problem for these methods because they have to raise the number of sample for irradiance caching and to raise the complexity of the illumination mesh for [19, 24]. As we wanted a method able to render very complex scenes with irregular illumination without increasing the complexity of the algorithm, we had to find another solution.

5.3. Basic idea

Our method has quite a simple principle. As we cannot store enough particles in memory to match the required accuracy, we thought about making several density estimations that use particle maps that fit into memory. These density estimations can be computed one after the other. Nevertheless, this only works if the little particle maps form a partition of the big particle map we should have used. The use of a different seed for random choices when computing each new particle map ensures that we build a partition.

We use a cache to store temporary radiances. This cache is an array of spectral values that have the same size as the final image. Each value of the array corresponds to a sample point. The sample points are the same for each pass as we compute the same image many times. The size of the cache is only dependent on the image size. For example, an image of 1024x768 pixels computed with 100 bands needs a cache of approximately 630 megabytes, which is less than the particle map size (as seen is section 4.5.2 848 megabytes for only 1 million of particles). And we can store in this cache the equivalent illumination to as many photons as we need. Thus writing and loading procedures from the disk are faster than if we had cached the particle map.

Figure 4 presents an example. In fact, the radiance density estimation at a sample point using 600 particles found in a particle map of 2 million particles is equal to the mean value of two density estimations, computed at the same sample point, using 300 particles found in 2 distinct particle maps of 1 million particles. The demonstration is presented in the next section. Figure 5 illustrates the equivalence between one pass and thirty passes.

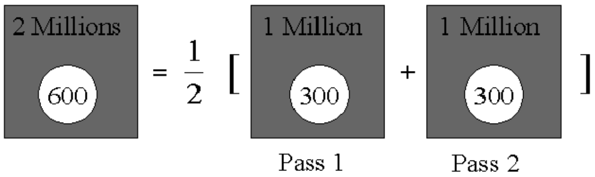
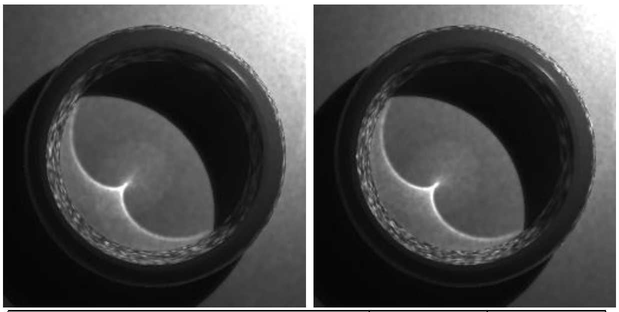


Figure 4. Example of a density estimation decomposition in two passes.



	Left	Right
Number of passes	1	30
Particle map size	3M	100K
Particles for density estimation	600	20
Pixel average value	1.39736	1.39929
Pixel standard deviation	1.19078	1.19402

Figure 5. Illustration of the equivalence between rendering with one (left) and thirty (right) passes.

5.4. Final image reconstruction

A pixel of the image defines a unique ray from the eye, and thus defines a unique intersection point in the scene. We saw in section 4.4 that the radiance in this sample point is given by:

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

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

Assuming that the light power for band $\Delta\lambda$, $\Phi_{\Delta\lambda}$, is divided equally among all the particles, we have:

$$\Delta\Phi_{p,\Delta\lambda}(x,\vec{\omega_p}) = \frac{\Phi_{\Delta\lambda}}{N_t}$$

where N_t is the total number of particles in the particle Map.

Combining with radiance expression gives:

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

Assuming that it exists an integer m such as :

$$N_t = mN$$

and assuming that :

$$n_r = mn$$

Then we have :

$$L_{r,\Delta\lambda}(x,\vec{\omega}) \approx \sum_{p=1}^{nm} f_{r,\Delta\lambda}(x,\vec{\omega_p},\vec{\omega}) \frac{\Phi_{\Delta\lambda}}{Nm\Delta A}$$

We can divide the sum in m terms :

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

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

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

We can write this as a sum:

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

We factorize:

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

The term $\sum_{p_i=1}^{n} f_{r,\Delta\lambda}(x,\vec{\omega_{p_i}},\vec{\omega})\frac{\Phi_{\Delta\lambda}}{N\Delta A}$ is a density estimation searching n particles in a particle map with N particles.

The equation shows that computing the radiance with m little particle maps of N particles using n particles for density estimation hence computing a mean value is the same as computing radiance with one big particle map of $N_t = \frac{N}{m}$ using $n_r = \frac{n}{m}$ particles for density estimation.

5.5. Parallel execution

Our method is ideally suited to take advantage of parallelism. Indeed, passes can be distributed over several calculators. Each calculator asks for a pass to compute. When the computation of a pass is finished, the calculator adds its pass to the final image then asks for a new pass if there are any left.

6. Tests and results

Our method was implemented in the industrialized software SPECRAY from OKTAL Synthetic Environment. We performed the tests on an Athlon 2.4 GHz with 512 Mb running Linux to obtain all results.

6.1. Specular tube

Figure 6 shows that the rendering of the specular and emissive tube is totally different from figure 2 when we use enough particles for density estimation. Indeed, the noise disappears.

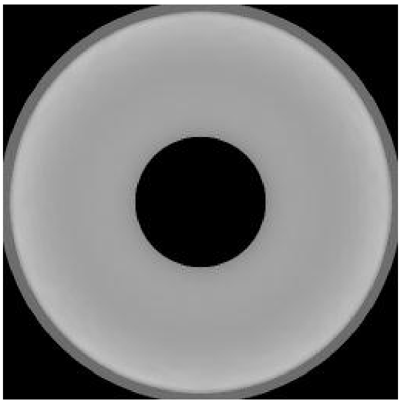


Figure 6. Rendering of an emissive and specular tube with one hundred million particles.

6.2. Infrared Conference room (Soda Hall)

This is a rendering in visible and infrared spectrum of the Conference Room from the Soda Hall. The scene contains approximatively 170000 triangles. In visible spectrum, there are 1300 extended light sources, in infrared all the triangles are emissive. Figure 7 shows the infrared and visible renderings.

The final visible rendering has been done in 200 passes using a 2 million particle map and searching 200 particles for each density estimation. The original size of the image is 1024x768 pixels. The infrared rendering has been done for three common bands :

- Short wavelength infrared (SWIR) from $1e^{-6}$ to $2e^{-6}$ meters,
- Medium wavelength infrared (MWIR) from $3e^{-6}$ to $5e^{-6}$ meters,
- Long wavelength infrared (LWIR) from $8e^{-6}$ to $12e^{-6}$ meters.

The rendering has been done in 100 passes using a 1 million particles map and searching 100 particles for each density estimation. The original size of the image is 640x480 pixels.

6.3. Discussion

We compute several little particle maps. All these particle maps are global in opposition to the localised particle maps dependent on the geometry of [13] which means that each particle map stores particles within the entire model. This ensures that the illumination is spread correctly over the scene for each pass and thus that density estimations are correct. We benefit from the speedup of computing multiple small particle maps shown in [13] without limitation on the mesh complexity because the particle maps are not dependent on the geometry. Unfortunately, when the image size is large (i.e 1024x768 pixels), the time saved while computing and sorting the particle map is lost when rendering due to the number of density estimations needed. Therefore, it is more interesting to compute fewer passes with bigger particle maps when rendering big images. The memory used is constant from one pass to another.

As we do not make a difference between direct and indirect illumination, the cost to reduce the noise in the generated image is more important than in [9] for visible scenes with few lights. However, our method offers better results for visible scenes with lots of light sources than [9] because the cost of tracing rays toward the lights becomes very expensive when the number of light sources becomes impor-

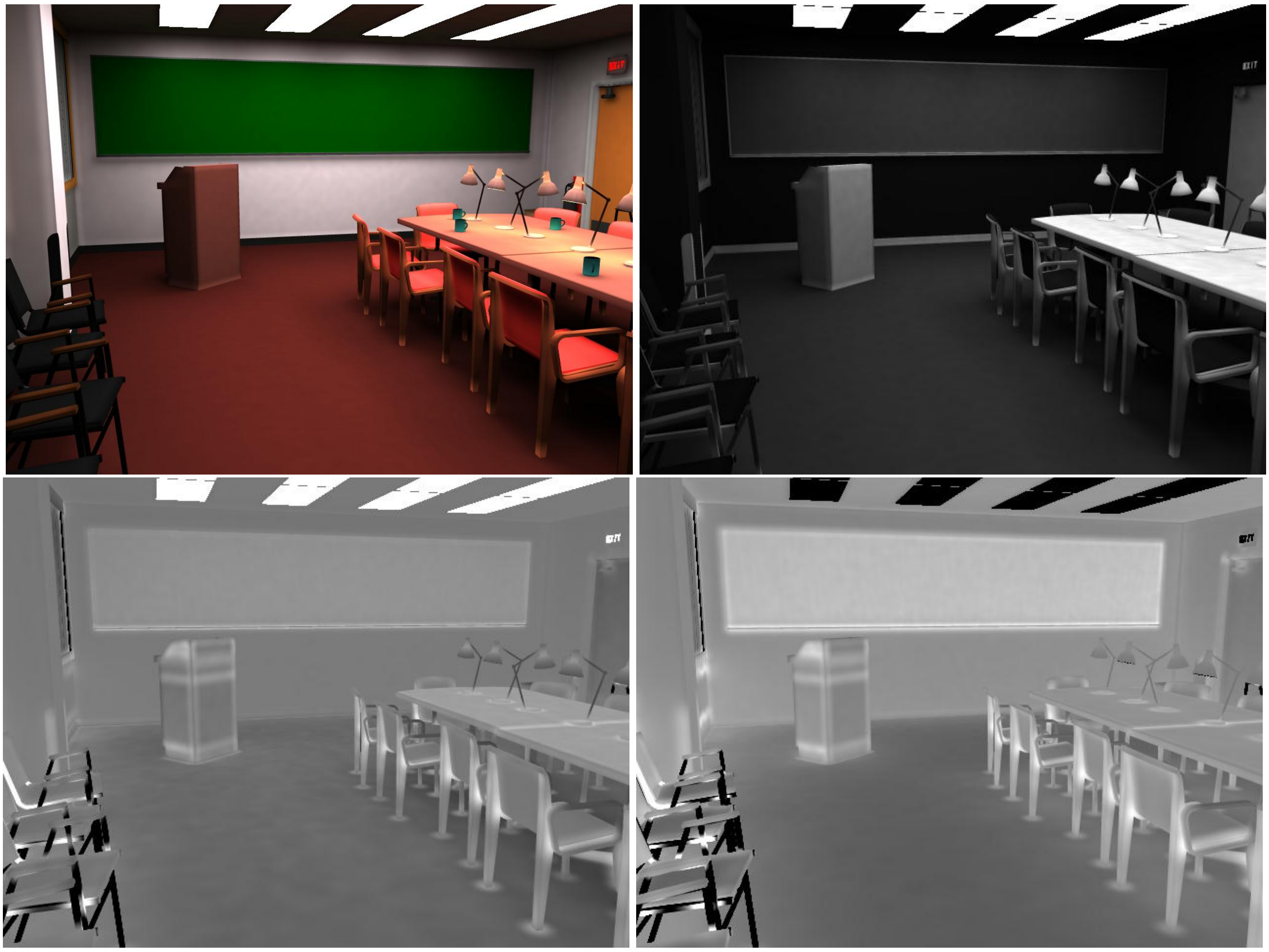


Figure 7. Rendering of the Soda Hall conference room in visible (top left) and infrared spectrum : in SWIR (top right), MWIR (bottom left) and LWIR (bottom right). In order to have a good contrast, the infrared image radiance has been spread out beetween minimum and maximum radiances which are respectively 0-10, 1.2-1.8, 31-37 $w.sr^{-1}.m^{-2}$

tant, whereas our method is independent of the number of lights.

The cache we use to store the illumination makes our method only dependent on the image size, whereas in [25, 19, 24] the cache is dependent on the illumination and scene complexity. Then our method is more suited to very complex scenes, and [25, 19, 24] are more suited to suited to simpler ones.

7. Conclusion and future work

We have proposed a multipass extension of the density estimation methods suited to infrared rendering. Our method is independent of the scene complexity. Since all surfaces are sources for infrared light, our method gives good results for rendering scenes with lots of light sources. However, we have also discussed that our solution is not suited to all types of scenes.

First of all evolutions, we should adapt a new spectral model to our method in order to handle infrared spectrum representation as easily as visible RGB representation.

At this time, inhomogeneous participating media are taken into account using an adaptative Ray-marching technique [9]. But this method is expensive due to participating media complexity. Thus, we have to improve our method to make it faster.

Then, we have to find a method to handle view dependent specular reflections in a better way. Maybe we could estimate directly the density of the particle map plus the evaluation of a set of secoundary rays distributed according to the specular BDRF.

Our method suffers from surestimation at edges of the model because we have not yet implemented the existing solutions to that problem [9, 14].

As we use a multipass method, we could optimize convergence time using a better construction of the particle map. Actually, the particle map of a pass could be analyzed and give importance information useful to guide the construction of the next pass particle map.

Finally, we use a kd-tree to store particles like in [9]. This provides an interesting compromise between search time and memory consumption. As we are no longer limited by the memory, we could use another data structure which provides better search time than a kd-tree.

References

[1] K. Devlin, A. Chalmers, A. Wilkie, and W. Purgathofer. Tone reproduction and physically based spectral rendering. In *State of the Art Reports, Eurographics 2002*, pages 101–123, 2002.

[2] G. Drettakis and F. Sillion. Interactive update of global illumination using a line-space hierarchy. In SIGGRAPH '97 (Los Angeles, CA), pages 57–64, Aug. 1997.

[3] P. Dutré and Y. D. Willems. Importance-driven monte carlo light tracing. pages 188–200, 1994.

[4] P. Dutr. *Mathematical Frameworks and Monte Carlo Algorithms for Global Illumination in Computer Graphics*. PhD thesis, University of Leuven, 1998.

[5] P. Dutr. Global illumination compendium. 2003.

[6] B. A. Fomin. A k-distribution technique for radiative transfer simulation in inhomogeneous atmosphere: 1. fkdm, fast k-distribution model for the longwave. *J. Geophys. Res.*, 109, 2004.

[7] J.-M. Hasenfratz, C. Damez, F. Sillion, and G. Drettakis. A practical analysis of clustering strategies for hierarchical radiosity. *Computer Graphics Forum*, 18(3), Sept. 1999. Proceedings of EUROGRAPHICS'99.

[8] J. Iehl and B. Peroche. An adaptive spectral rendering with a perceptual control. In *Computer Graphics Forum*, volume 19, Aug. 2000.

[9] H. W. Jensen. *Realistic Image Synthesis Using Photon Mapping*. A K PETERS LTD edition, 2001.

[10] J. T. Kajiya. The rendering equation. *Computer Graphics, Proc. of ACM SIGGRAPH 86*, 20(4):143–150, 1986.

[11] E. Lafortune. *Mathematical Models and Monte Carlo Algorithms for Physically Based Rendering*. PhD thesis, Katholieke Universiteit Leuven, Feb. 1996.

[12] E. P. Lafortune and Y. D. Willems. Bidirectional path tracing. *Compugraphics '93*, pages 95–104, 1993.

[13] B. D. Larsen and N. J. Christensen. Optimizing photon mapping using multiple photon maps for irradiance estimates. *WSCG POSTER proceedings*, Feb. 2003.

[14] F. Lavignotte and M. Paulin. A New Approach of Density Estimation for Global Illumination. In *WSCG*, Plzen , Czech Republic, Feb. 2002. University of West Bohemia.

[15] K. Myszkowski. Lighting reconstruction using fast and adaptive density estimation techniques. pages 251–262, 1997.

[16] F. E. Nicodemus, J. C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis. *Geometric Considerations and Nomenclature for Reflectance*. Monograph 161, National Bureau of Standards(US), Oct. 1977.

[17] G. Rougeron and B. Peroche. A adaptive representation of spectral data for reflectance computations. *Proc. of the 8th Eurographics Workshop on Rendering*, pages 127–138, 1997.

[18] P. Shirley. *Physically Based Lighting Calculations for Computer Graphics*. PhD thesis, University of Illinois, Nov. 1990.

[19] P. Shirley, B. Wade, P. M. Hubbard, D. Zareski, B. Walter, and D. P. Greenberg. Global illumination via density estimation. pages 219–230, 1995.

[20] F. X. Sillion. A unified hierarchical algorithm for global illumination with scattering volumes and object clusters. *IEEE Transactions on Visualization and Computer Graphics*, 1(3), Sept. 1995.

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

[22] E. Veach. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, December 1997.

[23] E. Veach and L. Guibas. Bidirectional estimators for light transport. *Fifth Eurographics Workshop on Rendering*, pages 147–162, 1994.

[24] B. J. Walter. *Density Estimation Techniques for Global Illumination*. PhD thesis, Program of Computer Graphics, Cornell University, Ithaca, NY, August 1998.

[25] G. Ward, F. Rubinstein, and R. Clear. A ray tracing solution for diffuse interreflection. In *Proceedings of SIGGRAPH 1988*, 1988.