Volume xx (200y), Number z, pp. 1–15

Compression and Importance Sampling of Near-Field Light Sources

Albert Mas, Ignacio Martín and Gustavo Patow

Institut d'Informatica i Aplicacions, Girona, Spain

Abstract

This paper presents a method for compressing measured datasets of the near-field emission of physical light sources (represented by raysets). We create a mesh on the bounding surface of the light source that stores illumination information. The mesh is augmented with information about directional distribution and energy density. We have developed a new approach to smoothly generate random samples on the illumination distribution represented by the mesh, and to efficiently handle importance sampling of points and directions. We will show that our representation can compress a 10 million particle rayset into a mesh of a few hundred triangles. We also show that the error of this representation is low, even for very close objects.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

1. Introduction

One of the most important factors for accuracy and realism in global illumination is lighting complexity. However, most of the times non measurement-based light sources are used. This is reasonable in applications where physical accuracy is not important, but it is critical in situations where we want the lighting simulations to be as close as possible to the real illumination.

Traditionally, a far-field approximation has been used in industry to model real light sources. A far-field representation models a luminaire as an anisotropic point light source, and assumes that objects to be illuminated are located far away. There are some established standards to represent far-field light sources, the most important being IESNA and EULUMDAT [ANS02, bCL99].

However, far-field representations do not produce accurate results when objects are close to the light source. As an example, a far-field representation of a bulb can not be used to compute the light distribution produced by a reflector, since the distance between the bulb and the reflector surface is usually very small. This is crucial for inverse reflector design applications, like in [EN91, PPV04, PP05].

In recent years there has been an important effort to improve real light source capture and representation methods using near-field models. In this case, luminaries are modeled as extended light sources, and there is no assumption on the distance of objects to be illuminated [Ash93, SS96, GGHS03a]. However, the raw data produced by these capture methods can produce several gigabytes of images. Captured data can be compressed to a light field or interpolated to generate a rayset [AR98]. A rayset is a set of particles (point, direction) with equal energy and without spectral distribution, emitted at the light source and captured in a virtual bounding surface (Figure 2). The capture process uses a gonio-photometer that could be mounted on two rotating arms that allow the device to capture the light coming from a source from all possible directions (Figure 1), but other setups are also possible.

Our method focuses compressing measured data of the near-field emission of physical light sources in a way that allows incorporating these sources within global illumination algorithms, such as photon mapping [Jen96]. Raysets are a convenient representation for these type of algorithms since they provide a set of particles that can be used directly for shooting. However, raysets are usually too big (10M particles) to be efficiently sampled.

The main contribution of this work is a novel approach for compressing dense raysets of near-field light source measurements into a set of point light sources, with no significant

Figure 1: *Gonio-photometer RiGO [Rad] used to capture the rayset from light source*

loss of information. We also present an importance sampling method to efficiently sample the emittance of the light source from the compressed data. A rayset is limited to its original elements, and therefore we are not be able to generate new ray postions or directions. The compression technique has the benefit that such new ray positions and directions can be generated

The rest of the paper is organized as follows. We discuss previous work in Section 2. We present an overview of our method in Section 3, present the compression method in Section 4, and in Section 5 we explain how to perform importance sampling on the light source. Then we show the results in Section 6. Conclusions are in Section 7.

2. Previous Work

Light source modeling from measured data can be classified into two categories. The first one is the far-field representation that is established as an industry standard (IESNA, EULUMDAT), and assumes large distances from the sources to the lighting environment, so spatial information in the emission of the light can be neglected. The light source is considered as a point light source and its emittance is modeled as a non-uniform directional distribution. These measurements are obtained using a gonio-photometer, and it is assumed that the target distances are several times the maximum dimension of the source. Empirically, this distance is no closer than five times the maximum width of the luminaire [Ash93].

The second category is the near-field representation, which stores the complete directional and positional emittance information for the source. In recent years there has been a development in devices that are able to capture the near-field light distribution from a light source [AR98, GGHS03a, GGHS03b]. These devices use a digital camera and a robot arm to take a set of photographs from the light source at different angles (Figure 1).

Near-field measurements can be represented as light fields [LH96, GGSC96]. The first attempt to capture the illumination of a light source as a light field was using canned light sources [HKSS98]. Using a representation similar to [Ash95], the idea was to compute a light field for a synthetic light source and then use it in a ray tracing environment. The main drawback of the method is that they don't show a way to importance sample the light field to use it in a particle shooting algorithm. Also, the method presented here outperforms those *regularly sampled* representations at very short distances, as the new method is specifically designed to preserve spatial and directional information following the distribution of the original rayset, something those representations cannot do.

More recently, in [GGHS03a] a method is presented for light source acquisition and rendering. The method allows for much faster rendering, and allows efficient sampling for photon emission. However, their representation needs around 100Mb per light using an efficient memory representation.

Near-field measurements can also be represented as raysets. This representation is just a set of photons that represent the emittance of a light source. The rayset is computed by interpolation from the set of images acquired during the capture process. These representation is the industry standard for optical illumination analysis software [Rad, Lam, Bre, OPT]

3. Overview

Our method deals with rayset models. The goal is to highly compress the data set with a small error even for closely illuminated objects.

A rayset consists of a list of pairs of a point and a direction (see Figure 2). Thus, each particle has a location and an outgoing direction. Each pair in the list can be considered as an exitant particle that comes from the measured light source, all of them carrying the same energy. The particles are located on a virtual bounding surface that usually corresponds to a convex surface like sphere or a cylinder, but different providers use different supporting shapes. To take into account this variety, our algorithm is able to handle any sort of surface as long as it is convex.

In a first step, we compute a partition of the initial rayset into clusters using the particle locations and their directions. For each cluster we compute a representative point, and an average particle density, obtained from the particles included in this cluster. In order to accurately capture the particle density changes, the clustering produces more density in areas

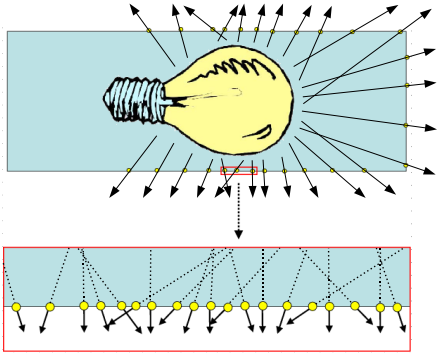

Figure 2: *A rayset is a set of particles (point + direction) stored on a bounding surface. These particles represent the light emission from a light source.*

with a rapid variation of particle density. Areas with constant particle density will have less clusters. Once the clustering is finished, we create an anisotropic point light source for each cluster. The position is the representative point, and the directional distribution is computed using the directions of the particles of the corresponding cluster. We use a simple constant basis function over a subdivision of the sphere of directions into spherical triangles (see Figure 3) in an hierarchical way.

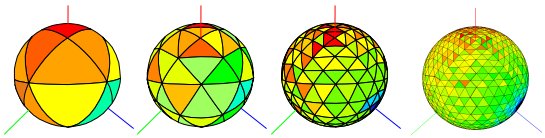

Figure 3: *A regular subdivision of the sphere using spherical triangles. From left to right, the images correspond to levels 1,2,3 and 4 of the subdivision.*

At the end of this process we have a set of direction-dependent point light sources (see Figure 4) located on the virtual bounding surface of the light source. Then, we triangulate this obtaining a mesh representation of the bounding surface (see Figure 5). This triangulation is used for importance sampling. Each time we sample the light source, we first select a triangle and then we select a point inside the triangle. Then we sample one of the three anisotropic point light sources corresponding to the vertices of the triangle for sampling an outgoing direction. This way, we ensure that we sample all the domain of possible outgoing directions.

4. Compressing the Near-Field

Rayset compression has two steps. The first one groups the particles using a clustering technique. The second one creates directional distributions over each cluster from the original particle directional data. The result is a set of point light sources that represents, in a simpler way, the original light source.

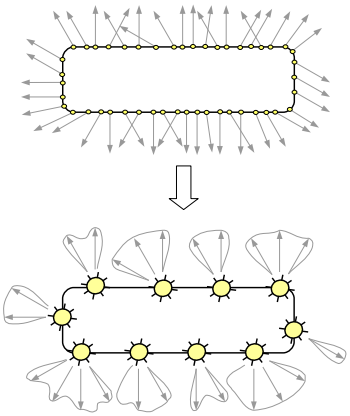

Figure 4: *The process of transforming a rayset into a set of anisotropic point light sources.*

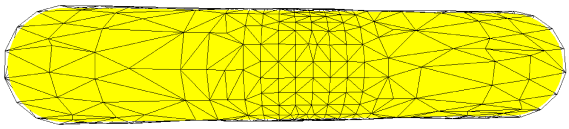

Figure 5: *Mesh produced from a 10M rayset corresponding to a OSRAM PowerBall bulb.*

4.1. Clustering creation

The first step of our method is to group the original set of particles into a set of clusters. At the end, each cluster will have associated a subset of particles, and a representative point. The goal is to have a higher cluster density where the emittance is more variable, so we can correctly capture the high frequencies on the emission distribution, both positional and directional.

The algorithm starts with a very dense initial clustering that is computed by creating an octree, see Figure 6. The criterion for voxel subdivision in this first step is to have a maximum number of particles in each voxel. This helps avoiding a too fine initial discretization, which would lead to an unacceptable computational cost. Our experiments show that octrees with a number of leafs between 20000 and 30000 are enough. Once the octree has been created, each leaf voxel of the octree corresponds to a cluster. Then, an iterative process removes unnecessary clusters until no cluster has to be removed. The particles of the removed clusters are redis-

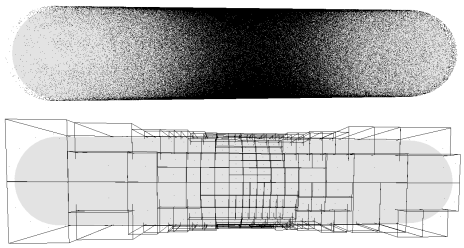

Figure 6: *Set of clusters (below) that represents the original particle spatial distribution (above).*

tributed to the remaining clusters. Each of these iterations has the following steps:

1. 3D triangulation of cluster representatives. This produces a mesh approximation of the luminaire virtual bounding surface.
2. All clusters are traversed and marked for removal in case that they are unnecessary. A cluster can be removed if it does not help to capture detail. We will show the specific criteria used below.
3. Cluster removal and particle redistribution. Particles of the removed clusters are redistributed to the nearest neighboring cluster that has not been removed.

After each iteration, the new cluster set is re-triangularizated before a new iteration starts. Following, we explain in more detail the first two steps.

4.1.1. Triangulation

In first step we need to triangulate the cluster representatives that lie on the virtual bounding surface of the light source. A cluster representative is the particle location belonging to the given cluster $\mathcal{C}_i$ that is closer to the average location of the subset of particles associated to the cluster. This average location of a cluster is computed as:

$$\mathbf{C}_i = \frac{1}{N_i}\sum_{j}^{N_i}\mathbf{P}_j$$

where $\mathbf{P}_j$ is the position of particle j, and N_i is the number of particles associated to cluster i. Then we choose the representative $\mathbf{R}_i$ as:

$$\mathbf{R}_i = \{\mathbf{P}_n : \forall m \in 1..N_i, m \neq n, \mathbf{P}_m \in \mathcal{C}_i \quad \|\mathbf{P}_m - \mathbf{C}_i\| > \|\mathbf{P}_n - \mathbf{C}_i\|\}$$

There are a lot of methods that calculate triangulated surfaces from point clouds, such as [SR01] or [AB99]. Unfortunately, most of these methods do not guarantee that all input points are used as mesh vertices, considering some of them irrelevant for mesh construction. On the one hand, as we are dealing with clusters of carrying-energy photons, we

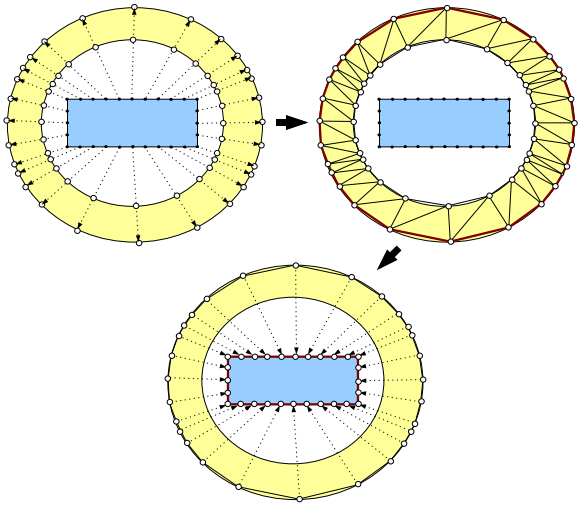

Figure 7: *Simplified 2D sectional illustration of light source bounding surface triangulation.*

can not neglect any point. So, we need a method that considers all input points, because each point represents a cluster. On the other hand, those methods have a high computational cost for a simple surface like the ones used by most rayset providers.

Our method starts from the tetrahedralization of the point cluster representation and the posterior elimination of tetrahedra sides that do not belong to the virtual bounding surface. The tetrahedralization is done by the Delaunay algorithm [She97]. The correct triangles are chosen in three steps. First, all input points are duplicated and projected over two concentric bounding spheres with different radii, the original ones over the inner sphere, and the duplicated ones on the exterior one. Then, all points are tetrahedralized, and the tetrahedra faces that have three vertices formed by points in the initial set are chosen. Next, the selected triangles are reprojected over the original volume surface, so the triangle edges are projected over the surface. In Figure 7 a simplified 2D sectional illustration of this method can be observed.

This algorithm is only valid for star-shaped surfaces, but the nature of the input models used in industry already guarantees that. Note that the center of the star must be also the center of the bounding spheres. This algorithm works for simple convex shapes, such as spheres or cylinders. For more complex convex shapes we could use the ball pivoting algorithm [BMR*99] with a large sphere.

4.1.2. Cluster removal

This is the second step of the iterative process mentioned before. Once the triangulation is finished, all clusters are traversed and tested for removal. The idea is that, if the density of a cluster can be approximated by linear interpolation of

t_a (deg)	t_d (%)	t_e (%)	Clusters	Loops	Time (sec)	Size MB
90	50	20	347	29	476	1.2
70	45	20	841	38	428	2.6
60	40	20	1680	48	496	5.8
30	35	15	4696	57	531	16.6
25	25	15	8776	51	642	31.6

Table 1: *Number of clusters, number of loops in iterative clustering process, precomputation time and result memory usage for different thresholds (angle difference, density and edge filtering threshold), for the Osram PowerBall.*

the neighboring vertices, then the cluster can be removed since the particle density is correctly represented without it. Then, the particles are redistributed among its neighbors, and the set is re-triangulated (which is much more efficient in this case than keeping information for an incremental update).

The algorithm consists of the following steps. For each cluster $\mathcal{C}_i$, a normal vector $\mathbf{N}_i$ and a plane $\mathcal{S}_i$ is approximated. This approximation is computed performing a nearest neighbors search centered on the cluster representative $\mathbf{R}_i$. This search of nearest clusters is accelerated using a kd-tree built with the original particle data. Next, all adjacent cluster representatives $\mathbf{R}_j$ in the triangulated mesh are projected onto $\mathcal{S}_i$ (see top of Figure 8). For each projected cluster representative $\mathbf{Q}_j$ and for $\mathbf{R}_i$, and considering only the 2D projected coordinates, the respective density values d_j and d_i are added as a third coordinate. Finally, a new regression plane $\mathcal{S}_r$ is calculated for the new points (see bottom of Figure 8). If the projection distance t between d_i and $\mathcal{S}_r$ is larger than a user-defined threshold (called density threshold t_d), the cluster cannot be removed from the mesh. The threshold is a percentage of the maximum projection distance of neighboring clusters. Otherwise, if this threshold test is passed, the distance from the cluster to its neighbors is verified, in a way such that the cluster will not be removed if the distance is larger than a given threshold, called edge filtering threshold t_e. The edge filtering threshold is a percentage of the length of the longest edge of the bounding box, which was observed to be a good distance measure. Finally, the cluster's mean emittance direction is compared with the ones from its neighbors, and the cluster is removed if the angle they form is smaller than a third threshold, the angle threshold t_a. Observe that this last verification avoids collapsing clusters on edges with sharp angles. Note also that this is only a first approximation that works sufficiently well on our experiments.

In Table 1 there are some results of clustering creation method.

4.2. Creation of Point Light Sources

Once the clustering is finished, we create a point light source for each cluster $\mathcal{C}_i$ at its representative point. The approximation used is to accumulate all the particles to $\mathbf{R}_i$. This will result in a set of directions centered at the point. As the number of directions can be quite high, we have to choose a more compact representation that can be efficiently sampled with importance sampling.

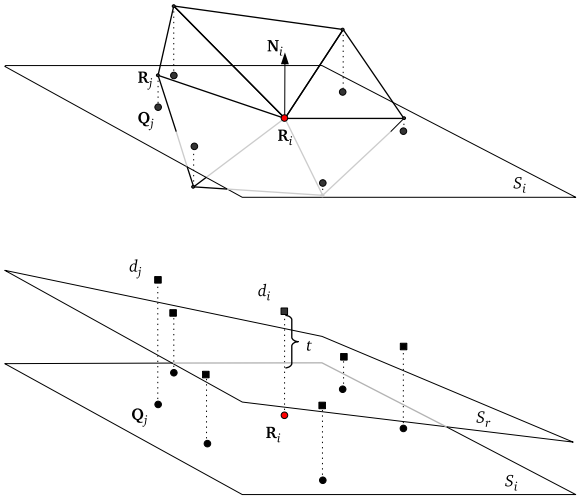

Figure 8: *Top: neighboring cluster representatives $\mathbf{R}_j$ are projected onto plane $\mathcal{S}_i$. Bottom: each projected cluster representative $\mathbf{Q}_j$ is augmented with its density values. A regression plane is computed with all d_j values.*

We have used a piece-wise constant set of basis functions defined over the sphere of directions in an hierarchical way. The support of each basis function is a spherical triangle. The sphere of directions is initially subdivided into 8 spherical triangles, and then each triangle is recursively subdivided using a quad-tree [Arv95]. We force this subdivision for a fixed initial number of levels so we get a uniform subdivision of the sphere of directions (see Figure 3). Then, a new subdivision is done adaptively to perform a tightest directional representation. For the raysets we have used to test our technique, we have found that from 3 to 5 levels of subdivision are enough to get an accurate representation (see Figure 9).

Once the subdivision is created, we store at each spherical triangle the number of original particles in the cluster with directions that belong to the corresponding solid angle. This number also represents the exitant energy through the corresponding solid angle. To improve compression, we only store the triangles that have non-zero energy. Typically, less than half of the triangles need to be stored as only half of the sphere is pointing towards the source of the light.

5. Importance Sampling

One of the main goals of the rayset compression technique is to be able to perform importance sampling on the light

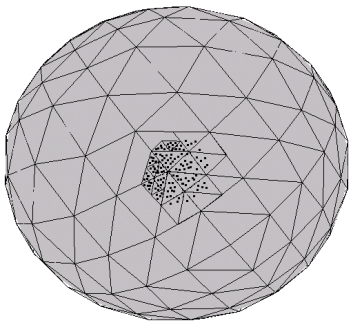

Figure 9: *Adaptive spherical triangle subdivision for directional data of a cluster. Each point over triangles is a ray direction in directional space.*

source. As one of the main advantages of the rayset representation is that it is very easy to use in light tracing algorithms (photon maps, radiosity, etc.), we want to maintain as much as possible the accuracy of our compressed model (within user-provided thresholds), but at a fraction of the original storage cost.

In order to be able to sample the complete domain, we create a triangulation of the bounding surface using the point light sources as the vertices for such triangulation (see Section 4.1.1). Once we have created the triangulation, every time we want to generate the 3D position of a particle, we first have to choose a triangle. We construct a probability density function (pdf) for this, and each triangle is assigned a given value proportional to its energy. We set this energy to the amount of original particles that exit the light surface through the given triangle, without taking into account the densities computed for the respective vertices.

With this pdf we can choose a triangle. Then we have to select a random point on the triangle. The straightforward approach would be to choose a point following a uniform distribution. But this poses another problem: the pdf that results is not continuous over the edges of the mesh, resulting in illumination artifacts. These artifacts are caused because the spatial distribution generated by this approach changes strongly from triangle to triangle (see Figure 10, left).

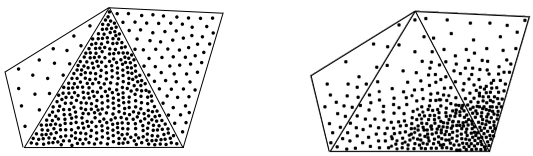

Figure 10: *Uniform sampling over triangles produces artifacts (left). A continuous sampling over the edges of the mesh avoids them (right).*

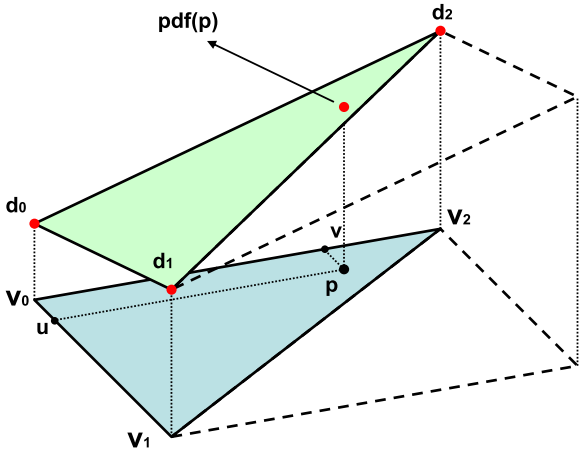

Figure 11: *Plot of the pdf used for sampling a point inside a triangle. We consider the function over the whole quadrilateral, but samples outside the triangle V_0,V_1,V_2 are rejected.*

To solve this problem we propose a non-uniform pdf that is C^0 continuous over the edges of the mesh (see Figure 10, right). The idea is to compute the density of particles at each vertex of the mesh and perform a linear interpolation across each polygon. For a triangle n we define the pdf at a point x in parametric space as (see Section 5.1):

$$p_n = \frac{d_0 + u(d_1 - d_0) + v(d_2 - d_0)}{2A_n \int_0^1 \int_0^{1-v} (d_0 + u(d_1 - d_0) + v(d_2 - d_0))\, du\, dv}$$

where u, v are the coordinates of point p within the triangle, d_0, d_1 and d_2 are the densities of vertices V_0, V_1 and V_2 (see Figure 11), and A_n is the area of the triangle.

Unfortunately, sampling values from p_n is a complex task due to the integration needed over the triangular domain, which would require a slow numerical integration. In order to simplify computations, we created an instrumental distribution p'_n, which is the extension of the previous pdf to the whole unit square, and get our samples by the rejection sampling method [Bek99], see below. To guarantee the positiveness of the new pdf, we choose the origin of the u, v parameterization (vertex V_0) as the vertex with a lower density, allowing us to be sure that the pdf is positive all over the domain. The pdf also holds the condition that the integral over the domain is 1:

$$p'_n(u,v) = \frac{d_0 + u(d_1 - d_0) + v(d_2 - d_0)}{2A_n \int_0^1 \int_0^1 (d_0 + u(d_1 - d_0) + v(d_2 - d_0))\, du\, dv} \quad (1)$$

If we want to generate random points proportionally distributed to this pdf we have to apply the principles described in [Scr66, Shi90]. Given two uniformly sampled random numbers $r_u \in [0,1]$ and $r_v \in [0,1]$, the sample point is

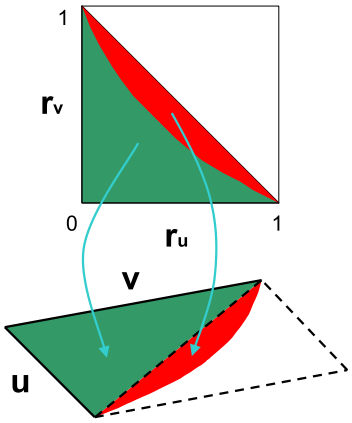

Figure 12: *Not all of the random sample pairs r_u, r_v such that $r_u + r_v \leq 1$ produce u, v pairs such that $u + v \leq 1$. However, all the random pairs in the green area produce u, v pairs that are not rejected.*

$u_i = F_0^{-1}(r_u)$, $v_i = F_1^{-1}(r_v)$, where functions F_0 and F_1 are defined using a function F:

$$F(u,v) = \int_0^v \int_0^u p'_n(u',v')\,du'\,dv'$$

and, from here:

$$F_0(u) = F(u,1)$$

$$F_1(v) = \frac{F(u_i,v)}{F(u_i,1)}$$

Solving these integrals gives a quadratic polynomial that can be easily evaluated. Unfortunately, this mechanism produces samples all over the domain $u_i \in [0,1]$, $v_i \in [0,1]$. That means that we have to apply rejection sampling and reject samples that verify $u_i + v_i > 1$. This can be very inefficient since at least 50% of the samples are rejected. Even worse, in case that d_0 is much lower than d_1 and d_2, the number of rejected samples can be much higher.

However, as vertex V_0 corresponds to the lower density, it is clear that random numbers such that $r_u + r_v > 1$ will always produce invalid samples (see Figure 12). Taking this into account, we always generate uniform random points on the triangle defined by equation $r_u + r_v \leq 1$. Results show that, by using this sampling strategy, the number of rejected samples is very small: in our experiments, less than 10% of the samples were rejected.

Figure 13 shows the original rayset point distribution (no directions shown) and the point set produced by our sampling technique.

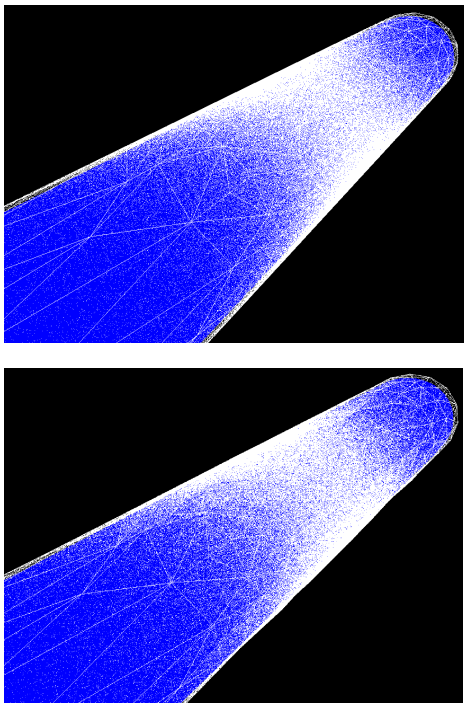

Figure 13: *Comparison of original rayset with importance sampled set. Top image shows the point distribution of the original rayset, and bottom image shows the point distribution generated by our sampling technique.*

5.1. PDF for Position Sampling

In order to have a continuous Probability Density Function defined all over the light surface domain, it is necessary to propose a well defined, continuous global function, and normalize it. We decided to use a piece-wise linear function defined over each triangle of our triangulation: at each vertex $\mathbf{P}_i$, we require the evaluation of the global function to be $f(\mathbf{P}_i) = d_i$, where d_i is the density associated with the corresponding vertex. Continuity is granted over different triangles as triangles with common vertices will use the same d_i values. This way, continuity is C^∞ within the interior of the triangles and C^0 at the edges.

In the u,v parameter space we can formulate an expression for the n-th triangle

$$f_n(u,v) = (d_1 - d_0)u + (d_2 - d_0)v + d_0$$

Now that we have our function defined over all the triangles i, we can compute the global normalization, defined as the integral $I = \int_\Omega f(\mathbf{r})d\mathbf{r}$ where the domain Ω is the triangulated surface we got. This expression is nothing else than the sum over all N triangles, each with domain Ω_n:

$$I = \int_\Omega f(\mathbf{r})d\mathbf{r} = \sum_{n=0}^{N-1} \int_{\Omega_n} f_n(\mathbf{r})d\mathbf{r}$$

by changing variables to the unit square, we get that this is equal to

$$I = \sum_{n=0}^{N} \int_{Tri_n} f_n(\mathbf{r}(u,v)) \left| \frac{\partial(x,y)}{\partial(u,v)} \right| dudv$$

where the integration domain Tri_n is the lower half triangle in the unit square, and $\frac{\partial(x,y)}{\partial(u,v)}$ is the Jacobian of the transformation [Wu99].

It is important to notice that computing the absolute value of the determinant of the Jacobian gives the same expression as $2A_n$, twice the area of the triangle $A_n = 1/2|(\mathbf{r_j} - \mathbf{r_i}) \times (\mathbf{r_k} - \mathbf{r_i})|$, where $\mathbf{r_i}$, $\mathbf{r_j}$ and $\mathbf{r_k}$ are the three vertices of the triangle.

Now, we can proceed with our integration

$$I = 2\sum_{n=0}^{N} A_n \int_0^1 \int_0^{1-v} f_n(u,v) dudv = 2\sum_{n=0}^{N} A_n I_n$$

Then, we can write our final pdf as

$$pdf = \frac{\sum_{n=0}^{N} f_n(r)\delta(\Omega_n, r)}{I}$$

where $\delta(\Omega_n, r)$ is 1 over the triangular domain Ω_n and zero everywhere else. In order to have a clearer sampling strategy, we can multiply and divide each term by $A_n I_n$, what will give

$$pdf = \sum_{n=0}^{N} \frac{A_n I_n}{I} \frac{f_n(r)\delta(\Omega_n, r)}{A_n I_n} = \sum_{n=0}^{N} \frac{A_n I_n}{2\sum A_n I_n} \frac{f_n(r)\delta(\Omega_n, r)}{A_n I_n}$$

which clearly is a linear combination of pdfs [Bek99]. Actually, we can say that it is a linear combination of N primary pdf's p_n:

$$p_n = \frac{f_n(r)\delta(\Omega_n, r)}{2A_n I_n} = \frac{f_n^{'}(r(u,v))}{2A_n I_n}$$

To be able to sample the pdf, a primary p_n is selected first with probability

$$p(n) = \frac{A_n I_n}{\sum A_n I_n}$$

and next a sample is drawn using p_n. so, the final expression for the pdf becomes

$$pdf = \sum_{n=0}^{N} p(n) p_n$$

Although only a single primary pdf is sampled, the result of that sampling is obtained following the combined pdf. Obviously, when sampling p_n the $\delta(\Omega_n, x, y)$ factor can be omitted, as it was built to be 1 all over the domain.

5.2. Sampling Directions

Sampling a direction involves using importance sampling on the three directional distributions stored at the vertices of the triangle.

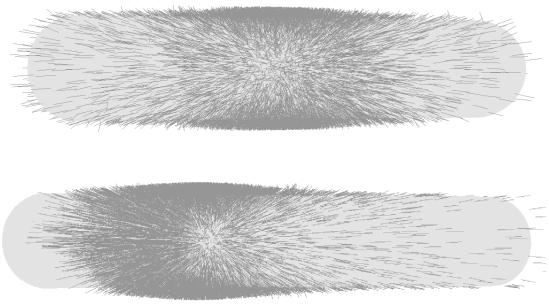

Figure 14: *Real-measured raysets. At top, the OSRAM PowerBall model. At bottom, the Tungsten Halogen model.*

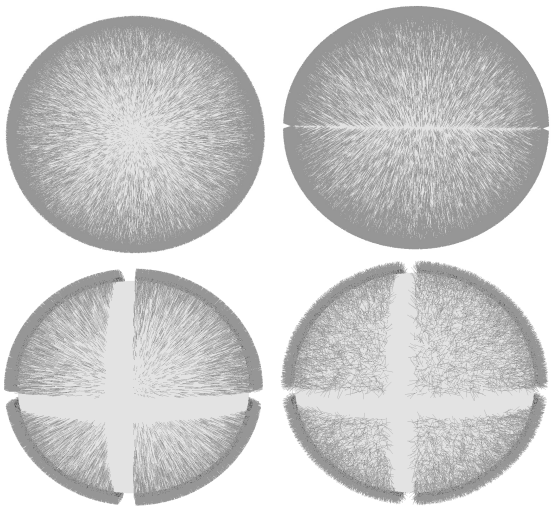

Figure 15: *Synthetic tested raysets: Phong (top left corner, Phong exponent = 500), Phong directional pattern (top right corner, Phong exponent = 25), radial with pattern (bottom left corner) and cosine with pattern (bottom right corner, exponent = 1) distributions.*

Once we have selected a point x, we compute its barycentric coordinates u, v and $1-u-v$. As their sum is 1, we can use these values to construct a small CDF for the three vertices. This allows us to select one vertex and its corresponding point light source. Then, we can perform importance sampling on the point light source by creating a CDF from all the spherical triangles. With the CDF we can select a spherical triangle and then sample it uniformly with respect to the solid angle.

5.3. Direct Illumination

On the one As explained above, the light source is represented by a bounding geometry, which can be sampled to

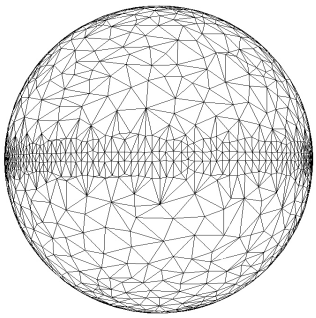

Figure 16: *The resulting triangulation of the Phong directional distribution with a directional patterns shown on the right top part of Figure 15.*

find the illumination at a given point. But, for lights with a high directional variation, this approach can be quite inefficient. Also, the highly directional distributions that can be generated with our method preclude the use of this technique in conjunction with Lightcuts [WFA*05], which is a technique valid for isotropic point lights or directional lights with a cosine distribution which cannot model those distributions.

So, we decided to separate the Global Map of the traditional Photon Map algorithm into two, as done in [GGHS03a]. On the one side, we create a Direct Map that stores direct impacts from the light sources, and on the other, a new Global Map that stores impacts after one diffuse reflection. So, two different possibilities arise to compute the direct illumination: to reconstruct the illumination directly from the Direct Map, or use the Direct Map for generating importance information to guide a standard Monte Carlo lighting integration. The option of directly using the Direct Map promises to be efficient for highly directional sources, which can be made even better by storing a large quantity of photons in the Direct Map, while keeping a low quantity for the Global Map. Of course, this involves the limitation of the integration to the areas where there are direct photons, or the construction of an importance sampling table from the nearest particles in the Direct Map [GGHS03a].

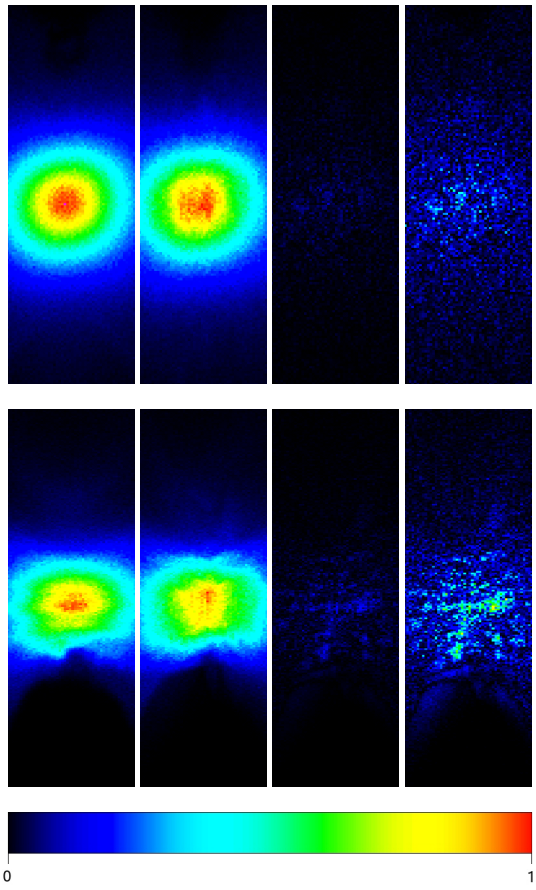

Figure 17: *Images of 10 million particles gathered on a plane situated at 1mm of the bounding surface. First row corresponds to the OSRAM PowerBall bulb with a compressed data of 1680 clusters (see Table 2). Second row corresponds to the Tungsten Halogen bulb with a compressed data of 452 clusters. In columns, from left to right, the images correspond to original rayset, sampled compressed data, difference image (error), and scaled difference image respectively (both at x5). Under the false color images you can find the scale used, normalized over the entire set of positions/directions.*

6. Results and Discussion

We have tested our method with two raysets corresponding to real measurements. Both of them have 10 million particles. The first one corresponds to a OSRAM PowerBall bulb (courtesy of Lambda Research), and the second one is a Tungsten Halogen bulb (Radiant Imaging demo) (see Figure 14).

Also, we have tested four synthetic raysets (see Figure 15), sampling 10 million particles for each one in a uniform way. The first one has a Phong distribution over a sphere. We use the same samplig method than [LW94] to construct the Phong distribution.. The second one has Phong distribution, but with a directional pattern distribution. The other two are radial and a cosine ray direction distribution over the sphere, but with a positional pattern onto the sphere of origins. These synthetic raysets are used to check the performance of our method in different conditions, such as high frequencies in ray positions or directions. It is specially interesting the case of the Phong directional pattern distribution, which leads to a triangularization which is shown in Figure 16, showing that

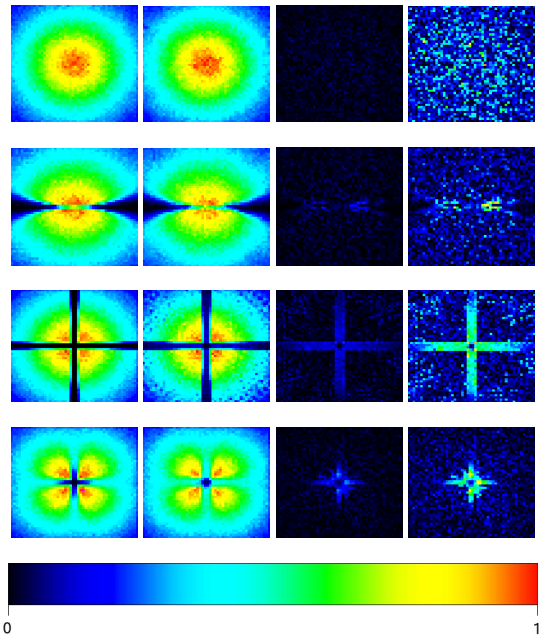

Figure 18: *Images of 10 million particles gathered in a planes situated at 1mm of the bounding surface. First row corresponds to the Phong synthetic rayset, using a compressed data of 1597 clusters (see Table 2). Second row corresponds to the Phong Pattern synthetic rayset, using a compressed data with 1146 clusters. Third row corresponds to the radial pattern synthetic rayset, using a compressed data of 4454 clusters. And the fourth row corresponds to the Cosinus Pattern synthetic rayset, using a compressed data of 2244 clusters. In columns, from left to right, the images correspond to original rayset, sampled compressed data, difference image (error), and scaled difference image respectively (phong model at x8, and the others at x3). Under the false color images you can find the scale used, normalized over the entire set of positions/directions.*

the angle threshold criteria for the triangularization effectively preserves the discontinuity in the distribution

Figures 17 and 18 show false color images for particle emission experiments. The images represent the energy arriving at a plane that is located 1 mm from the bounding surface of the rayset. There is a side by side comparison between the 10*e*6 original particles and a 10*e*6 particle emission using importance sampling. Also, difference images are displayed for each one. Observe that, with the above explained method, no photons are generated *inside* the bounding surface or *pointing inwards* from it, so any surface intersecting its interior will not receive any hit. Actually, this cannot be a problem since this is the space physically occupied by the light bulb itself.

We have tested different compression levels for each one

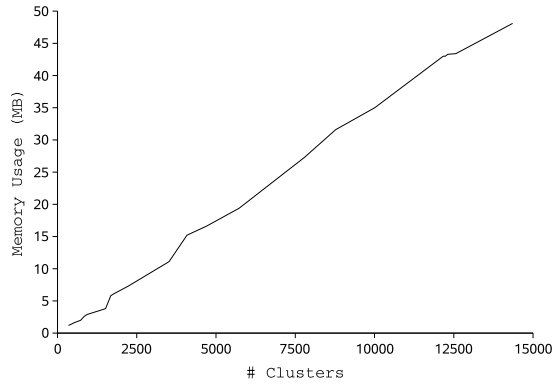

Figure 19: *Relationship between number of clusters and memory usage for OSRAM PowerBall.*

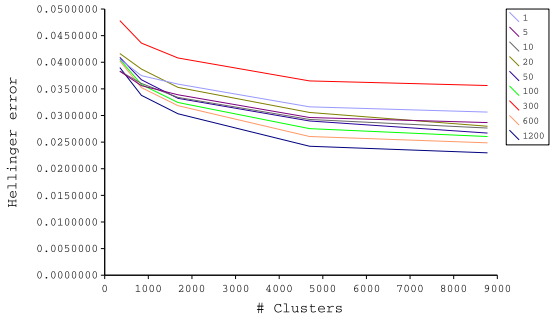

Figure 20: *OSRAM PowerBall Hellinger errors for different measurement distances in function of number of clusters .*

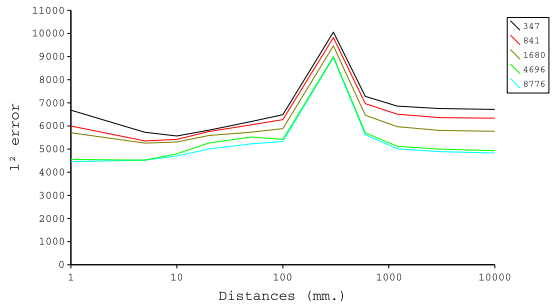

Figure 21: *OSRAM PowerBall l^2 errors for different number of clusters in function of measurement distances .*

of our raysets. The memory sizes have been reduced drastically, as it can be seen in Figure 19, since the rayset representation of all of these models have a memory consumption of about 270MB. In Figure 20 and Figure 21 there are some results of the OSRAM PowerBall rayset using different compression levels and measuring the error at different

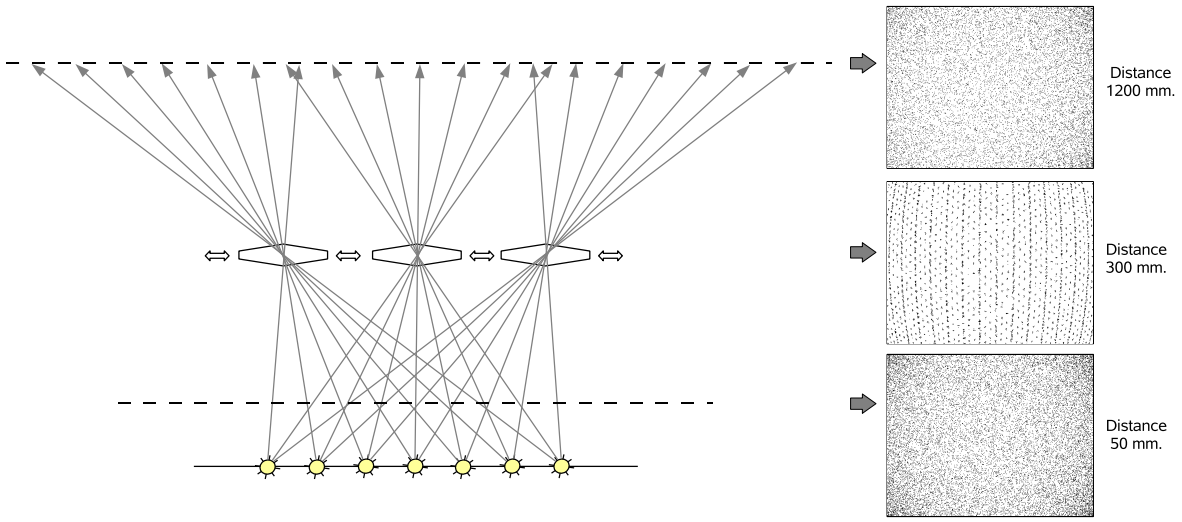

Figure 22: *Left: acquisition system scheme. Right: ray gathering over bounding spheres at different distances. The observed pattern at distance of 300 mm corresponds to photosensor distance, and each shot accumulation is each photosensor placement.*

distances. Two error metrics have been used: l^2

$$D_{l^2}(a,b) = \sqrt{\sum_i^N (a_i - b_i)^2}$$

and Hellinger [RFS03],

$$D_{Hellinger}(a,b) = \sqrt{\frac{\sum_i^N (\sqrt{\frac{a_i}{N}} - \sqrt{\frac{b_i}{N}})^2}{2}}$$

with similar behavior on results.

In Figure 20 it can be observed how the error decreases as the number of clusters increases, in the same way for each tested distance.

In Figure 21 three zones of interest are shown. The first one is the error obtained at near distances, as 1 mm. In this case the importance sampling positional error is the main contributor to the overall error. The second one is the error obtained at large distances. Here, the directional sampling error was found to be the main source of error. The third case is the peak observed at distance 300 mm. To explain it, we have traced the particles of original rayset on a set of bounding spheres of different radii. The results (Figure 22) show that, at distance 300 mm, it can be observed a pattern over sphere. This is because the pattern of the acquisition mechanism that has been used to obtain the rayset can be found in this region. If the gonio-photometer used in acquisition system uses photosensors placed over a virtual bounding sphere, then the gathering distance (bounding sphere radii) is 300 mm. So, each accumulation point in the pattern corresponds to each photosensor position in the acquisition process.

All the other models have also been tested. In Table 2 there is a summary of l^2 error values obtained for each rayset, with different numbers of clusters, and at different distances from the light sources. All models show similar behaviors to the previously explained one. As a reference, we have included for each rayset the errors for a far-field distribution created with the original raysets, but using 2048 spherical triangles, as a distribution with less spherical triangles fails to keep the different pattern details. As we can see, the new compression method outperforms the far-field representation at very short distances, also demonstrating that a far-field representation is unsuited for real light bulbs at short distances, as they cannot be approximated by an anisotropic point light. At large distances compared with the size of the light bulb, both converge to the same values, showing that for those distances it is better to use a far field representation, because of the easy evaluation. However, many applications (e.g. reflector design) require evaluations at short distances, where a far-field is clearly not good enough. One further point should be noted: all measurements in Table 2 have been evaluated by the procedure described in Section 5, so they have a variance associated. The variance depends on the the emitting distribution, the more diffuse, the more variance, as shown in [PPV04]. So, the measurements have a variance, which we have measured to range from ± 31 and ± 64 for the Tungsten Halogen and the OSRAM Powerball respectively, to values of ± 125 for the cosine pattern (which is like a Phong lobe with exponent $k = 1$), of ± 64 for the radial pattern, of ± 41 for the Phong and Phong pattern dis-

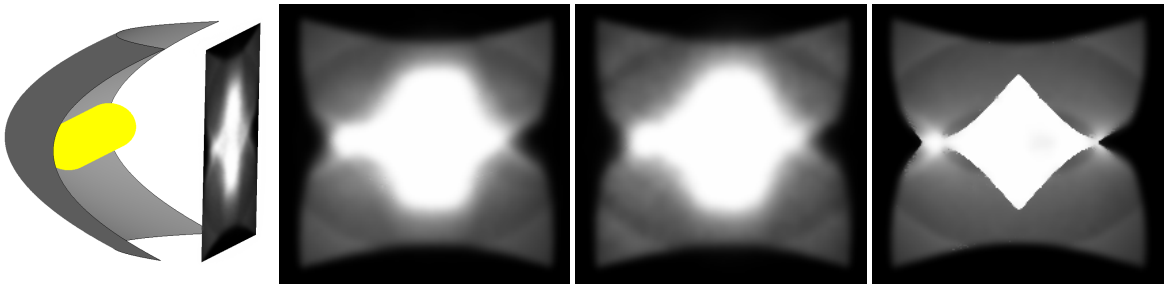

Figure 23: *Lighting from a reflector with the OSRAM Powerball mounted in. At left, the reflector and bulb setup, and the plane used to gather the lighting. Next, from left to right, the lighting using the original rayset, using the compressed rayset (1680 clusters, see Table 2) and using only the bulb farfield.*

tributions (with $k = 500$). This variance is enough to explain some strange behaviors at large distances for some distributions, as the values plus their respective variances overlap, as happens for the cosine patern distribution at 100 and 1200 mm. Also, in Table 2 we have included the resulting sizes of each compressed set, clearly showing the much lower memory usage required by our compression method.

To prove that the representation is accurate enough for cases such as reflector design, we show in Figure 23 a set of renderings of the OSRAM PowerBall bulb model mounted in a reflector, illuminating a plane. We have used three representations of this bulb: the original rayset, the compressed rayset and the farfield. The compressed model has 1680 clusters (see Table 2).

Finally, we have rendered some examples using the Mental Ray Renderer on Maya. To do it, we have developed a plugin that works as interface between our compressed rayset and the Maya rendering system. In Figure 24 you can see a comparison between two Photon Mapping results (without gathering), one using the original rayset, and the other using our compressed rayset, both placed in a near (1 mm.) bounding box around the light source bounding volume. The figures are rendered using only the Direct Map mentioned in Section 5.3. The figure uses the OSRAM Powerball example, which has a bounding cylindrical shape (70 mm. length and 20 mm. diameter dimensions). There are some gaps on the illumination of the sides of the box for original rayset, because very few photons are emitted at cylinder caps. These gaps disappear on compressed rayset results, due to an insufficient sampling, creating a smooth filtering effect.

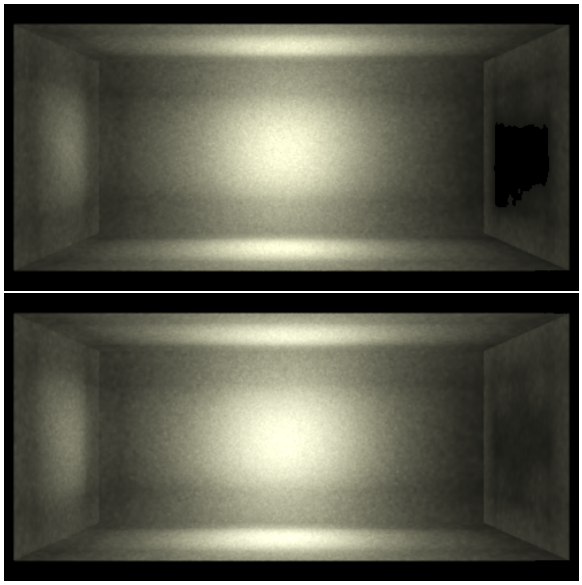

Figure 24: *Photon Map results (without gathering). At top there are the original rayset result. At bottom there is our compressed rayset result.*

6.1. Discussion

The most time and storage consuming part of our method is the directional distribution management. Obtaining the spherical triangle from a given direction means descending through the levels of the subdivision, and this is a costly operation. It must be taken into account that this kind of directional non-analytical representations always have a higher cost than analytic distributions [LRR04, MPBM03].

On the other hand, directional distributions are also very storage consuming. In our technique, we have the advantage that, with a low number of point light sources, we can characterize the illumination distribution of the bulb and, as we only store information of the spherical triangles with non-zero energy, the total memory used is not too high in comparison.

Ray Set	♯Clust.	Size (MB)	Distances (mm.) 1	5	10	50	100	300	1200	3000	10000
Osram	347	1.2	6682.60	5726.10	5566.62	6188.14	6488.13	10057.50	6858.35	6754.26	6714.48
Power-	1680	5.8	5705.55	5258.51	5306.23	5731.51	5884.42	9463.65	5969.38	5807.02	5768.99
Ball	8776	31.6	4455.03	4525.02	4702.06	5224.36	5328.28	8960.63	5011.70	4890.81	4831.09
	FF	0.017	12567.00	9957.12	8288.00	5356.63	4964.11	8850.31	4741.15	4586.17	4530.26
Tungsten	452	0.66	6222.04	6559.96	6866.29	7911.24	8383.64	8888.70	9167.13	9227.08	9251.94
Halogen	1160	2	4664.81	4923.44	5159.80	5672.74	5875.72	6125.81	6293.39	6281.38	6274.02
	2702	2.5	4460.94	4713.48	4948.81	5068.34	5159.71	5265.77	5369.85	5329.95	5345.32
	FF	0.017	6215.19	5764.81	5595.39	5456.35	5576.94	5575.54	5640.79	5638.35	5600.29
Phong	432	0.94	10672.50	12590.80	11269.40	7594.12	7424.74	7288.99	7232.32	7268.94	7277.87
Pattern	658	1.4	9085.03	10265.20	8963.22	6897.97	7224.52	7191.77	7023.90	7056.02	7054.05
$k = 25$	1146	3.2	5844.87	6301.60	5876.75	6179.90	6744.38	6669.15	6430.00	6442.95	6478.13
	FF	0.017	50999.89	48431.90	45218.50	27912.10	18765.60	9441.57	6165.64	5894.46	5863.09
Phong	430	0.76	4609.50	4647.55	4706.65	5426.99	5977.92	6555.81	6811.54	6874.29	6905.22
$k = 500$	1597	2.2	4419.81	4469.87	4550.48	5207.57	5553.68	5872.38	6033.69	6070.29	6091.30
	5349	6	4281.86	4367.94	4474.97	5121.33	5207.98	5032.52	4975.92	5007.21	5028.44
	FF	0.017	4439.08	4413.14	4408.29	4362.60	4328.09	4316.66	4291.46	4288.56	4294.95
Radial	350	0.35	27099.80	27008.30	26889.60	26619.90	26634.10	26762.00	26858.60	26888.90	26904.40
Pattern	1431	0.7	20322.60	20253.20	20166.90	19812.40	19765.60	19892.40	20041.90	20083.60	20105.70
	4454	1.3	10765.80	10794.50	10828.90	10675.10	10432.10	10148.10	10080.50	10078.10	10069.90
	FF	0.017	18449.10	18436.50	18420.40	18376.50	18364.00	18359.10	18354.90	18354.10	18352.00
Cosine	2244	16.9	6687.94	4544.10	4433.10	4430.48	4461.37	4711.09	4749.27	4471.47	4429.14
Pattern	3782	29.9	5300.57	4381.88	4391.92	4384.90	4423.53	4703.22	4883.37	4550.63	4449.72
	8177	71.5	4760.15	4248.68	4287.43	4375.61	4483.85	4625.79	4998.64	4542.61	4391.55
	FF	0.017	22778.30	15483.10	11775.30	5671.10	4850.43	4584.34	4435.97	4340.02	4308.85

Table 2: *Summary table of memory storage needs and l^2 errors for the tested ray sets. Three representative cluster solutions and a far-field (FF) representation have been tested for each ray set at different distances from the light source. The far-field spherical triangle subdivision is similar for each case, so the memory usages differ in a few bytes*

submitted to COMPUTER GRAPHICS *Forum* (5/2008).

7. Conclusions and Future Work

We have presented a novel approach for compressing near-field light source measurements. From a dense rayset we create a mesh with a relatively low number of triangles that contains illumination information. One of the main contributions is the method to perform importance sampling in this mesh representation. Our method allows for smooth point distributions over the mesh with almost no artifacts, even for very close objects.

It is important to notice that our importance sampling method for smoothly generated random points on the mesh can be used for other applications. Any point cloud distributed over a surface can be represented using our approach. This is one of our future research lines.

The mesh representation is also very efficient in storage terms. High precision raysets can contain up to 10 million particles. The storage needed for this representation is about 270Mb, while our equivalent mesh representation uses up to a few megabytes.

We consider, as future work, the use of non constant piece-wise linear functions to represent the directional distributions, such as wavelets or spherical harmonics. Also, a more discriminant comparisions in clustering process, such as mean directions, can be improved.

Finally, a GPU hardware implementation of the sampling algorithm is another of our future research lines.

8. Acknowledgments

We would like to thank Radiant Imaging for the rayset models, and the anonymous reviewers for their useful comments. This work was done under grant TIN2004-07672-C03 and the Ramón y Cajal program, from the Spanish Government.

References

[AB99] AMENTA N., BERN M.: Surface reconstruction by voronoi filtering. *Discrete Computational Geometry 22*, 4 (1999), 481–504.

[ANS02] ANSI/IESNA: Lm-63-02. ansi approved standard file format for electronic transfer of photometric data and related information, 2002.

[AR98] ASHDOWN I., RYKOWSKI R.: Making near-field photometry practical. In *Journal of the Illuminating Engineering Society* (1998), vol. 27, pp. 67–79.

[Arv95] ARVO J.: Stratified sampling of spherical triangles. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1995), ACM Press, pp. 437–438.

[Ash93] ASHDOWN I.: Near-Field Photometry: A New Approach. *Journal of the Illuminating Engineering Society 22*, 1 (Winter 1993), 163–180.

[Ash95] ASHDOWN I.: Near-Field Photometry: Measuring and Modeling Complex 3-D Light Sources. In *ACM SIGGRAPH '95 Course Notes - Realistic Input for Realistic Images* (1995), pp. 1–15.

[bCL99] BYHEART CONSULTANTS LIMITED: Eulumdat file format specification, 1999. http://www.helios32.com/Eulumdat.htm.

[Bek99] BEKAERT P.: *Hierarchical and Stochastic Algorithms for Radiosity*. PhD thesis, Department of Computer Science, Katholieke Universiteit Leuven, Leuven, Belgium, 1999.

[BMR*99] BERNARDINI F., MITTLEMAN J., RUSHMEIER H., SILVA C., TAUBIN G.: The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics 5*, 4 (October/December 1999), 349–359.

[Bre] BREAULT RESEARCH ORGANIZATION: http://www.breault.com/.

[EN91] ENGL H. W., NEUBAUER A.: Reflector design as an inverse problem. In *Proceedings of the Fifth European Conference on Mathematics in Industry* (Teubner, Stutgart, 1991), M. H., (Ed.), pp. 13–24.

[GGHS03a] GOESELE M., GRANIER X., HEIDRICH W., SEIDEL H.-P.: Accurate light source acquisition and rendering. In *ACM SIGGRAPH 2003* (2003).

[GGHS03b] GRANIER X., GOESELE M., HEIDRICH W., SEIDEL H.-P.: Interactive visualization of complex real-world light sources. In *Proceedings of Pacific Graphics 2003* (Canmore, Alberta, October 2003).

[GGSC96] GORTLER S. J., GRZESZCZUK R., SZELISKI R., COHEN M. F.: The Lumigraph. In *Computer Graphics Proceedings, Annual Conference Series, 1996 (ACM SIGGRAPH '96 Proceedings)* (1996), pp. 43–54.

[HKSS98] HEIDRICH W., KAUTZ J., SLUSALLEK P., SEIDEL H.-P.: Canned light sources. In *Rendering Techniques '98 (Proceedings of Eurographics Rendering Workshop '98)* (1998), Drettakis G., Max N., (Eds.), Springer Wien, pp. 293–300.

[Jen96] JENSEN H. W.: Global Illumination Using Photon Maps. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)* (1996), Springer-Verlag/Wien, pp. 21–30.

[Lam] LAMBDA RESEARCH, INC.: http://www.lambdares.com.

[LH96] LEVOY M., HANRAHAN P.: Light Field Rendering. In *Computer Graphics Proceedings, Annual Conference Series, 1996 (ACM SIGGRAPH '96 Proceedings)* (1996), pp. 31–42.

[LRR04] LAWRENCE J., RUSINKIEWICZ S., RAMAMOORTHI R.: Efficient BRDF importance sampling using a factored representation. In *ACM SIGGRAPH 2004* (August 2004), pp. 496–505.

[LW94] LAFORTUNE E., WILLEMS Y.: *Using the modied phong reflectance model for physically based rendering*. Technical report cw197, Dept. of Computer Science, Katholieke Universiteit Leuven, Leuven, Belgium, 1994.

[MPBM03] MATUSIK W., PFISTER H., BRAND M., MCMILLAN L.: A data-driven reflectance model. In *ACM SIGGRAPH 2003* (August 2003), pp. 759–769.

[OPT] OPTIS, INC.: http://www.optis-world.com/.

[PP05] PATOW G., PUEYO X.: A survey of inverse surface design from light transport behavior specification. *Computer Graphics Forum 24*, 4 (2005), 773–789.

[PPV04] PATOW G., PUEYO X., VINACUA A.: Reflector design from radiance distributions. *International Journal of Shape Modelling 10*, 2 (2004), 211–235.

[Rad] RADIANT IMAGING, INC.: http://www.radiantimaging.com.

[RFS03] RIGAU J., FEIXAS M., SBERT M.: Refinement criteria based on f-divergences. In *EGRW '03: Proceedings of the 14th Eurographics workshop on Rendering* (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 260–269.

[Scr66] SCREIDER Y.: *The Monte Carlo Method*. Pergamon Press, New York, N.Y, 1966.

[She97] SHEWCHUCK J.-R.: *Delaunay Refinement Mesh Generation*. PhD thesis, School of Computer Science, Carnegie Mellon University, 1997.

[Shi90] SHIRLEY P.: *Physically Based Lighting Calculations for Computer Graphics*. Ph.D. thesis, November 1990.

[SR01] S.FLOATER M., REIMERS M.: Meshless parameterization and surface reconstruction. *Computed Aided Geometric Design 18* (2001), 77–92.

[SS96] SIEGEL M. W., STOCK R.: A general near-zone light source model and its application to computer automated reflector design. *SPIE Optical Engineering 35*, 9 (September 1996), 2661–2679.

[WFA*05] WALTER B., FERNANDEZ S., ARBREE A., BALA K., DONIKIAN M., GREENBERG D. P.: Lightcuts: a scalable approach to illumination. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers* (New York, NY, USA, 2005), ACM, pp. 1098–1107.

[Wu99] WU K.: *The Transformation Between Positional Space and Texture Space*. Hpl-1999-103 technical report, HP Labs, 1999.