

International Journal of Shape Modeling
© World Scientific Publishing Company

REFLECTOR DESIGN FROM RADIANCE DISTRIBUTIONS

GUSTAVO PATOW, XAVIER PUEYO

*IiA, Universitat de Girona
E-17071, Girona, Spain
{dagush,xavier}@ima.udg.es*

ALVAR VINACUA

*Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya
E-08035, Barcelona, Spain
alvar@lsi.upc.es*

Received (Day Month Year)

Revised (Day Month Year)

Accepted (Day Month Year)

Communicated by (xxxxxxxxxx)

This paper proposes a technique for the design of reflector shapes from prescribed optical properties (far field radiance distribution) and geometrical constraints, which is of high importance in the field of Lighting Engineering, more specifically for Luminaire Design. The reflector shape to be found is just a part of a set of pieces of what is known in lighting engineering as an optical set, and is composed of a lamp (light source), a reflector, a holding case and a glass that protects the system from dust and other environmental phenomena. Thus, we aim at the design and development of a system capable of generating a reflector shape in a way such that the optical set emits a given, user defined, far field radiance distribution. This problem can be put in the mathematical context of inverse problems, which refer to all the problems where, contrary to what happens with traditional direct problems, several aspects of the scene are unknown. Then, the algorithm is allowed to work backwards to establish the missing parameters. In order to do so, light propagation inside and outside the optical set must be computed and the resulting radiance distribution compared to the desired one. Finally, constraints on the shape imposed by industry needs must be taken into account, bounding the set of possible shape definitions. The general approach taken is based on a minimization procedure on the space of possible reflector shapes. The algorithm moves towards minimizing the distance, in the l^2 metric, between the resulting illumination far from the reflector and a prescribed, ideal optical radiance distribution specified at the far field by the user.

Keywords: Reflector Design; Inverse Problems; Inverse Rendering

1991 Mathematics Subject Classification: 68U05, 68U07, 65D17, 78A97, 78A46

1. Introduction

Finding the surface shape of a reflector from a given set of required optical properties and physical constraints is of high importance in the field of Lighting En-

gineering (see ¹⁵, ²⁴ and ⁶ for an introduction to the subject). In this field, most of the effort is spent in building a reflector and testing it, perhaps by simulating its optical properties as a whole, but more often by building a physical prototype of the reflector where measurements are carried out, discarding it if it does not match the specifications, building another, testing it again, and so on. This process is extremely expensive and time-consuming.

Some tools have been developed to face the problem of reflector shape design for lighting engineering, but, in general, they follow the build-test-restart principle used by traditional engineering ¹⁰ ¹² ¹¹, which is an extremely costly and painful process, even with the fact that traditional designers already have a reasonably good idea of what the final reflector shape will be.

In this paper we present a fully automated solution for this problem, widely known to be a difficult one, specially in the theoretical and numerical aspects. Thus, although at this moment our solution is slow (it takes in the order of days), we aim at showing the feasibility of such computations, in the sense that our algorithm is already able to be used in the industry, lowering current production costs in terms of time and money. As future work, we point out several ways of improving the performance.

The paper is organized as follows: In Section 2 we present the previous work done in the field so far, and compare it against our solution. In Section 3 we give the problem formulation, and in Section 4 we make an overview of the proposed solution. In Section 5 we discuss the choice of the representation for the outgoing radiance (the radiance that goes out of the system). In Section 6 we present surface representation alternatives, Section 7 explains the light simulation step, and in Section 8, we present and discuss various possible optimization strategies ⁸. In Section 9 the implementation details are described, we present our results and conclusions in Section 10 and finally Section 11 presents future work.

2. Related Previous Work

The central problem of the paper can be put in the context of inverse problems. In illumination simulation, inverse problems are those where we know the effect of the illumination and we have to compute some of the parameters that produce this effect. These include problems such as:

- Light source positions and their orientations ³³ ³⁵, where the objective is to find the locations of known light sources in order to achieve a desired illumination.
- Luminaire emittance ⁴¹ ²³ ²⁰ ²⁹ ⁴⁰, where the emittances of already placed light sources are computed to obtain a desired illumination on some surfaces that define the scene.
- Surface characteristics of some relevant surface elements ³⁴, that is, finding the parameters of Bidirectional Reflection Distribution Functions on prescribed surfaces of the scene ²⁹ ⁴⁰ ⁹ ¹⁷ to produce the desired effects in the

environment.

- The shape or position and orientation of the reflectors in the scene ^{1 4 5 13 14}.

One common characteristic of this kind of problems is that, in general, we know in advance the desired effect of the illumination at some regions of the scene (their final radiance distribution). Then, the algorithm has to work backwards to establish the missing parameters. For a recent survey on inverse problems in rendering, refer to ³².

Fortunately, it has been shown ⁴ that the general problem of finding the reflector shape, given the illumination distribution in a general scene, can be strongly simplified. This simplification proceeds in a first step by reformulating the problem to the one that results from the inverse propagation of requirements, backwards from the specified surfaces to an enclosure surrounding the reflector. As a result an outgoing spatial radiance distribution for the sources is obtained. In this way, the problem of finding the reflector shape from the light distribution in the surfaces of a scene can be simplified to find the shape from the required outgoing light distribution from the optical set (reflector, bulb and diffusor).

In this paper we present a solution to the problem of finding the shape of a reflector given the outgoing radiance distribution that should emanate from the resulting optical set, without diffusor, as seen at the far field region, i.e.: large distances from the optical set. Our solution differs from previous approaches in:

- the type of surface used to define the reflector shape (a regular grid of heights instead of a bicubic b-spline ^{16 28}) that gives more flexibility in the range of achievable surfaces (although introducing C^0 continuity on the edges joining triangles).
- the generality of the light propagation simulation step which, in our case, is based on the well known Monte Carlo Light Tracing algorithm that can handle all sorts of Bidirectional Reflection Distribution Functions.
- handling interreflections in an efficient and natural way. Traditional approaches work without taking interreflections or general BRDFs into account ^{25 28 16}.
- the global strategy used for obtaining the desired reflector surface.

3. Problem Formulation

The reflector shape to be found is just a piece of a set called in lighting engineering an *optical set*, which consists of a light bulb, the reflector and the diffusor (Figure 1). The reflector has a border, contained in a plane, that limits its shape, as seen in Figure 1. In general, a reflector must fit inside a holding case, so its shape cannot be lower at any point than the plane defined by the border nor higher than a certain threshold defined by the case. We can say that the case defines a bounding box for the reflector.

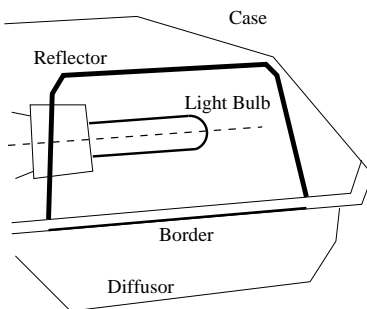


Fig. 1. An Optical Set.

Many luminaires incorporate semi-specular reflectors and refractive optics that require physical BRDF measurement and advanced ray-tracing support for repetitively structured optics such as prismatic diffusers, but our objective in the present work is aimed towards the simplest configuration for an optical set, consisting of a light bulb and a reflector surface, which is a common configuration for illumination settings at streets, tunnels and general open spaces, besides other cases.

In view of the simplification mentioned in the previous Section to the problem of finding the shape of a reflector from the specification of lighting on the surfaces of a generic scene, we can focus our efforts on the following problem:

Given the outgoing radiance distribution of a light bulb and a reflector border, and given a desired optical set-outgoing radiance distribution, find the corresponding shape for the reflector. Do this up to a user-defined tolerance.

The following constraints are imposed on the surface shape to be built:

- (1) The shape must satisfy certain constructive constraints that amount to requiring that the shape of the reflector be the graph of a function with respect to the plane of the reflector's border.
- (2) The resulting shape must exactly fit the given border.
- (3) The shape cannot be lower than the border plane ($z = 0$), or higher than a certain maximum height (it must fit inside the case).

4. Overview of the Proposed Method

Our solution approach starts by reformulating the problem in the following manner: in an iterative procedure, minimize the distance between the outgoing radiance distribution of the current reflector, $OutRad(\eta)$, and the desired (user specified) outgoing radiance distribution, $OutRad_{desired}$ (see Figure 2), where η is a vector of dimension n (i.e.: $\eta \in \mathbb{R}^n$) and defines the shape of the reflector (Figure 3). Each representation interprets it as needed: e.g. splines see it as a bidimensional array of control points, while height fields use them as an array of surface points.

We define a function $f(\eta) :: \mathbb{R}^n \rightarrow \mathfrak{R}$ of the form

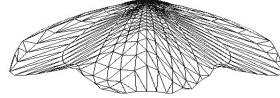


Fig. 2. An example of a real Outgoing Radiance Distribution.

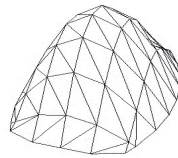


Fig. 3. An example of a reflector in our representation.

$$f(\eta) = \text{dist}(\text{OutRad}(\eta), \text{OutRad}_{\text{desired}})$$

The function dist can be naturally defined in terms of a norm in the space of outgoing radiance distributions, as $\text{dist}(X, Y) \equiv \|X - Y\|$, where, $\|\cdot\|$ is some properly defined norm. Our objective will be to minimize the function f .

Basically, the algorithm works “around” the chosen starting reflector by generating a family of new ones by iteratively moving each original vertex. Those new reflectors are evaluated and the ones with errors close to the one with the best error so far are averaged to obtain a final one. Actually, “close” in this context refers to all reflectors whose evaluation gave a value with a variance that superposes with the error and the variance of the best one. Once this new reflector is generated, and if the user-defined tolerance has not been achieved yet, the algorithm proceeds by refining the surface by adding new vertices, and restarts the brute-force optimization loop mentioned above.

To evaluate the $\text{OutRad}(\eta)$ function, we consider the optical set formed by the reflector η and the light bulb. Each time the evaluation of $\text{OutRad}(\eta)$ is required, two steps are undertaken: first, generate a surface with the shape specified by η . Then, simulate light transport in this reflector shape.

5. Choice of Outgoing Radiance Representation

In general, the information provided by bulb manufacturers or optical set manufacturers is based on the supposition that the size of the bulb is negligible with respect

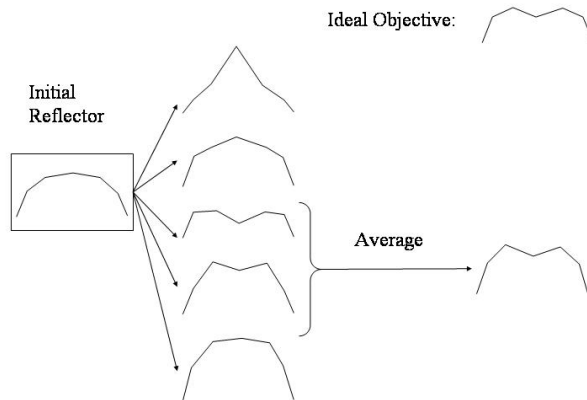


Fig. 4. Overall schema of the optimization procedure. A family of reflectors is generated, and the better ones with superposed variance bars are averaged to obtain the one used at the next iteration.

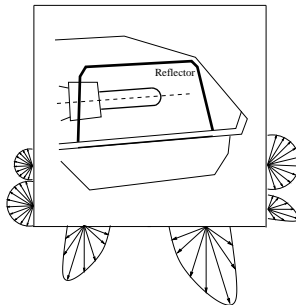


Fig. 5. The theoretical domain of outgoing radiance representations.

to the reflector (which is *not* always the case), and that the final measures on the resulting optical set are taken at a distance that is large compared to the physical dimensions of the set. This allows to make an approximation identical to the one used by Blinn and Newell for Environment Maps ², where the origin of the rays is neglected and only their direction is used to index the radiance representation.

As already noted, those outgoing radiance distributions must be discretized in order to be manageable by a computer. That discretization should be done both in the spatial and angular (directional) variables. Since the former were considered negligible, it is only necessary to discretize into a finite set of directions, further simplifying our radiance representation.

We choose to use the $C - \gamma$ coordinate system ⁶ as our discrete representation

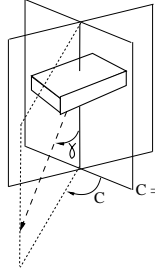


Fig. 6. The $C - \gamma$ coordinate system.

for the outgoing radiance distributions, because it represents a standard in the lighting engineering industry. This coordinate system is depicted in Figure 6. It is worth mentioning that the $C - \gamma$ representation is equivalent to any other spherical coordinate system (see ⁷ Section 10.1.4, General Luminaries). As we can see, the $C - \gamma$ representation is an angular representation given by two angles, c and γ ($c \in C \subset [0, 2\pi]$, $\gamma \in \Gamma \subset [0, \pi]$).

It is easy to see that there is a correspondence between the $C - \gamma$ representation and the space $\mathbb{R}^{s \times t}$ of real matrices $s \times t$, with $s = Size(C)$ and $t = Size(\Gamma)$, see Section 9 for details on the implementation. Actually, it is a common practice in the lighting engineering jargon to refer to them as $C - \gamma$ matrices. From now on, and in virtue of the mentioned correspondence, we will use interchangeably both matrix notation and the more formal (and correct) $C\Gamma$ notation. This allows us to say that $OutRad(\eta)$ can be modelled by the discretized version $OutR(\eta)$, which receives the parameter η that defines a reflector surface, and outputs a $C - \gamma$ matrix for outgoing radiance distribution, see Figure 7. In this way, each entry of the $OutR(\eta)$ matrix is the intensity going in the ij -th direction with a subtended solid angle ω^{ij} .

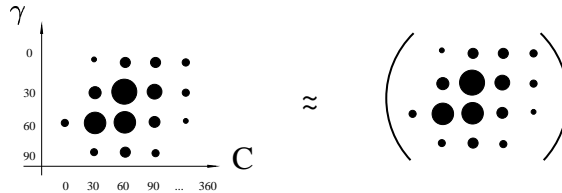


Fig. 7. Diagram showing a $C - \gamma$ matrix. Observe the degeneracy for $\gamma = 0$, which we solved by correcting the values after each manipulation.

It is important to mention that, for practical reasons in the implementation of the light propagation algorithms described below, we have chosen to extend this representation from the discrete set \mathcal{S}' to the whole sphere of directions \mathcal{S} by per-

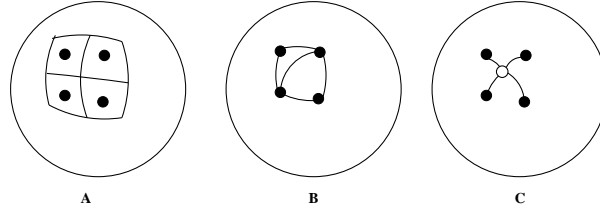


Fig. 8. Extensions from the discrete $C-\gamma$ domain to a real function on the unit sphere. A) constant, B) bi-linear, C) non-linear

forming a bilinear interpolation in the $C-\gamma$ space, but this extension is performed only in the mentioned step, without affecting the rest of the computations. It can be argued that this would lead to inaccuracies due to the continuous nature of the extension versus the discrete approximations performed so far, but it was found in practice that this effect is negligible in comparison with the other approximations and numerical factors, leading to differences two or more orders of magnitude below the other magnitudes involved. Other possible non-linear extensions were tested (See Figure 8, where three possible schemes are presented: constant, bi-linear and non-linear extensions), such as those proposed by Neubauer et al.²⁸. We found that, although all the benefits commented in those works are true, the non-linear nature of the extensions produced artifacts in the resulting distributions that turned them unusable for our purposes.

As file format we used a simple text file describing the number of C (s) and γ (t) angles used, then a listing of those angles (a list of $s+t$ entries) and, finally, the listing of the $s \times t$ values for the respective intensities in the corresponding directions (evaluated by the associated c and γ angles).

5.1. Choice of Distance

The distance metric we have used for our outgoing radiance representation is the well known l^2 norm. Since we are using the $C-\gamma$ representation for outgoing radiances, we therefore take the error to be given by

$$Error_2(\eta) = \left(\sum_{ij} \omega^{ij} |\mathbf{OutR}(\eta)^{ij} - \mathbf{OutR}_{desired}^{ij}|^2 \right)^{1/2} \quad (1)$$

i.e. summing the module squared of the difference between matrix entries, where the sum over indices ij must be understood over the two angular indices in the $C-\gamma$ representation, $i \in [0, s]$, $j \in [0, t]$ and ω^{ij} is the solid angle subtended by the ij -th entry of the $C-\gamma$ matrix.

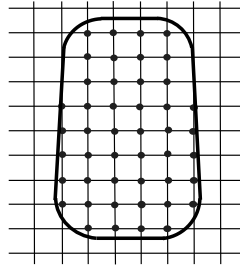


Fig. 9. Lattice Border and Optimizable points.

6. Shape Representation

The choice for the representation of the shape is a very difficult one, since *any* specific shape representation scheme has limitations to model free-form surfaces. All possibilities involve, one way or the other, some sort of restriction on the shapes that could be built. For example, there are real reflectors that present only C^0 continuity along a section of the reflector along its main axis¹⁵, while others present perfect continuity, at least to third degree.

We choose to implement the regular grid representation scheme because:

- An accurate template or tool to build a physical reflector from the resulting data can easily be made⁶.
- The overall convergence and stability of the algorithm can be easily tested.
- The routines for border matching are very simple: Only the grid points *inside* the border are available for optimization (the ones marked with dots in Figure 9). When assembling the surface, the polygons that join the grid points that define the surface in the interior of the border with the border itself are created, making a polygonal approximation to the originally given border, see Figure 10. In order to do this, the border is intersected with the grid lines and the intersection points (marked with circles in Figure 10) are used to build the polygons. Those polygons are created following a pattern like the one shown in the Figure.

On the other hand, regular grid representations present the following disadvantages

- C^0 continuity at *all* the triangle lines in Figure 10, that are all over the surface.
- To be flexible to adapt to the illumination requirements imposed by the $OutRad_{desired}$ radiance distribution, the surface built this way needs many vertices, which increases computing times.
- A fine grid is needed to achieve the smoothness needed to achieve the manufacturing standards of the industry.

Our current implementation naturally verifies constraints of type 1 and 2 enu-

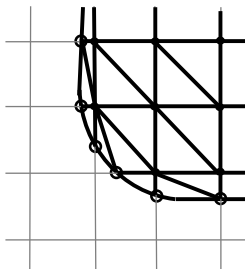


Fig. 10. Close up of tessellated lattice border.

merated in Section 3. Constraints of type 3 are treated by adding them as penalizing terms to the objective function, thus transforming the constrained problem to an unconstrained one. The penalty imposed to the objective function is the square of the constraint violation. In this way inequality constraints are handled through a penalty function that “turns on” when the constraint is not satisfied. For example, the inequality constraint given by $(GridVertex_i < UpperBound)$ results in a penalty term $f_{C_i} = A_i(UpperBound - GridVertex_i)^2$ when $GridVertex_i$ is greater than $UpperBound$ and zero otherwise²³, with A_i being a convenient weighting factor defined as a system constant with a very large number. Thus, in the final optimization process a new error metric is used, taking into account the defined constraints for each vertex:

$$Error_{used}(\eta) = Error_2(\eta) + \sum_j \delta(f_{C_j}, GridVertex_i)$$

where $\delta(x, vertex)$ is a function that returns x when $vertex > UpperBound$ and zero otherwise. By using this $Error_{used}$ instead of the original $Error_2$ we are able to solve a constrained problem almost in the same way we would for an unconstrained one.

7. The Light Simulation Step

This step has as a main objective the computation of the outgoing CT radiance distribution from a given reflector shape. For this, several possibilities were studied, arriving to the conclusion that Monte Carlo Light Tracing is the best option: this algorithm is, by far, the most efficient and straightforward to code, since it propagates the radiance from the light sources to the rest of the optical set²². In this method, each fired ray contributes to the final result and there are no lost rays. Light rays are sampled at the light source according to its energy distribution per solid angle, shooting more rays where more light is emitted. Thus, we get an optimal usage of the computational effort of tracing rays, since each ray propagates from the light source and each of them contributes significantly to the final result, which in turn results in several orders of magnitude less rays (and less computing

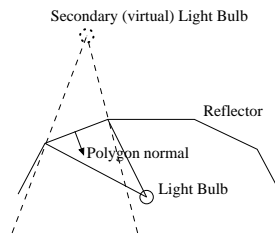


Fig. 11. Picture of our approximate but fast light propagation test algorithm.

time) to achieve the same accuracy and guarantee a variance small enough to make the results reliable. It is important to take into account that the *Light Tracing* algorithm was developed with extended, non-isotropic light sources in mind, uniformly sampling locations over the light bulb surface. Thus, although all our experiments were done with point light sources, our algorithms should perform equally well on lamp photometric data provided by near-field goniophotometers (see ³⁹).

Unfortunately, such a method requires too many samples to limit the variance, so we also implemented for our tests a second light simulation algorithm that sacrifices accuracy through some approximations to gain speed.

This specifically-tailored algorithm is based on the following hypothesis:

- Perfect Specular surfaces. No attenuation computed in the BRDF.
- No attenuations due to propagation.
- No visibility computations are performed.
- As a consequence, no interreflections are computed.
- $C - \gamma$'s computed at infinity, so the origin of rays is not taken into account.
- Polygonal approximation for the reflector surface.
- Isotropic point light source located at the origin

With all these assumptions and simplifications, a fast, specific light propagation algorithm can be developed, which is fast enough for us to test our optimization strategies (see Section 8) without losing the main properties of the function to evaluate, see figure 11.

Our algorithms were tested with both methods, showing the same convergence properties in both cases.

8. Optimization strategy chosen

In this section we will describe the optimization strategy developed. The highly nonlinear nature of the function to optimize must be taken into account, as well as the fact that it is plagued with local minima hiding the global minimum we are searching for.

There are various possible optimization strategies available, some enforcing the local properties of the search (with the risk of converging to a local minimum), and

others performing a global search. Among the first kind of optimizers, we tried the nonlinear Conjugate Gradients method^{36,42} and Powell’s method³⁶; we tested the Simulated Annealing method³⁸ for the second kind. They were chosen because of their well-known convergence properties and their stability for tough problems¹⁸.

8.1. *Classic strategies behavior*

By close examination of the function shape, we observed some very important features:

- There is, at least in lower dimension problems (up to four vertices to optimize), a clear global minimum surrounded by many local minima.
- An extremely high coupling among the different degrees of freedom (i.e.: the vertices of the reflector, see above) is present.

The first of these two facts, and the properties of Monte Carlo computations (Monte Carlo evaluation of the radiance distribution is noisy in its nature, lowering its variance as the number of used samples increases. See Section 7), preclude the use of local descent methods like Powell’s or Conjugate Gradients³⁶. This is specially true for the last one, where the noisy Monte Carlo method makes the computation of derivatives either meaningless or, if an appropriate strategy for computing derivatives is used³⁷, too costly to be affordable. This is so by the impossibility of accurately computing small variations of the reflector parameters, which obviously excludes the computation of derivatives.

And, because of the high coupling among the different degrees of freedom by the highly non-linear shape of the function to optimize, the optimization cannot be done by taking subsets of the vertices. It is an “all or nothing” situation.

8.2. *Foundations*

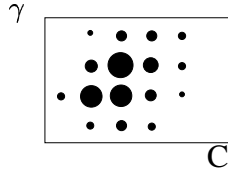


Fig. 12. Diagram showing a vertex influence region, Δ_i . Bigger dots mean greater difference values.

It will be useful to introduce a new $C - \gamma$ matrix Δ_i whose jk -th element is defined by:

$$\Delta_i^{jk} = |\mathbf{OutR}^{jk}(\eta) - \mathbf{OutR}^{jk}(\eta^i)| \quad (2)$$

where η is the vector defining the reflector shape and η^i the modified vector whose components are $(\eta^i)_j = (\eta)_j + \delta_j^i \epsilon$, with δ_j^i the Kronecker delta, ϵ is a small distance

(in our case, it was set to 5 millimeters, see Section 9). It is important to mention that Δ_i represents the i -th vertex differential influence on the output $C - \Gamma$ (see Figure 12), which shows us the regions on the $C - \Gamma$ matrix affected by the i -th vertex given its current position.

We can sort the different vertices according to the degree of “wrongness” they show. This deviation from the desired position can be measured by comparing the desired radiance distribution with the current one, and studying the overlap between this and the different vertex influence regions. So, we can define the D $C - \gamma$ distribution whose jk -th element is defined by:

$$D^{jk} = |\text{OutR}^{jk}(\eta) - \text{OutR}_{desired}^{jk}| \quad (3)$$

which shows the difference between the current $C - \gamma$ distribution and the desired one, and enables us to classify the vertices accordingly to their “status”, which is a computation of the superposition of the Δ_i and the D $C - \gamma$ distributions weighted by the subtended angles. Its elements are defined by (see Figure 13):

$$Status_i = \sum_{jk} \omega^{jk} \Delta_i^{jk} D^{jk} \quad (4)$$

with Δ_i the differential region of influence of the i -th vertex on the resulting $C - \gamma$ distribution, as defined in Equation 2. It is important to observe that $Status_i \geq 0$, as the Δ_i and D are matrices of positive elements.

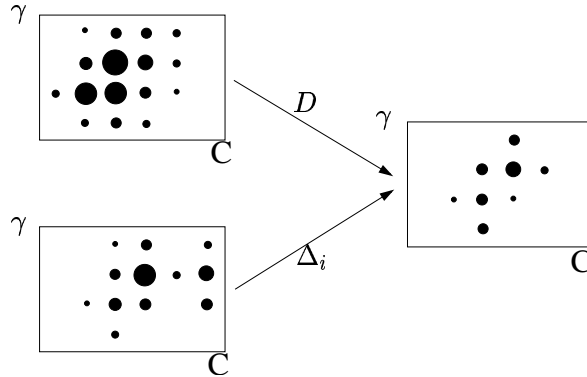


Fig. 13. Drawing of the superposition between D and Δ_i , as defined in equation 2 and following.

It is very important to mention that, because of the far-field approximation used so far, any measure, and specially the $Error_{used}$ defined in Equation 1, would be more sensitive to changes in reflector shape than in particular vertex deviations. So, if the overall reflector shape is right, the error would be low disregarding how far each particular vertex could be from the desired position in the desired reflector. Thus, $Status_i$ would provide an ordering for the different vertices that gives precedence to reflector shape over final position of the vertices, so it should be taken as



Fig. 14. Reflector measurements: when the middle one is taken as reference, the left one gives lower error than the right one, which has all its vertices more or less in their right places.

a relative measure of deviation, more than an absolute indicator of each vertex's positional error. That is, it is useful to indicate regions of wrong shape, and not individual vertex position deviations (see Figure 14).

The general idea behind the algorithm is to sort the vertices V_i according to their $Status_i$ and then optimize them with a global optimization algorithm. We cannot take a subset of the vertices because of the high overlap (correlation) among them.

8.3. Algorithm description

The algorithm developed can be simply described as:

```

Reflector := create a low-res reflector
while (not converged) and (not userDefinedStop)
  FreeVertexList := all vertices in Reflector
  WrappedRefl := wrap(Reflector, FreeVertexList)
  while (not converged) and
    (FreeVertexList is not empty)
    addVertices(WrappedRefl, FreeVertexList)
    optimize(WrappedRefl)
  if (not converged)
    increaseResolution(Reflector)

```

We start with a low dimensional reflector (generally, a 2×4 reflector, but when taking into account symmetry factors, only 4 vertices remain), and try to optimize its shape. In order to alleviate the quadratic nature of the change in reflector resolution (number of vertices to optimize), and also to allow a more progressive reduction of the error distance, we introduce a wrapping scheme that allows a progressive introduction of vertices to optimize, see the following subsections.

8.3.1. Wrapping the surface

Basically, a surface wrapper is a data structure that wraps around the basic polygonal surface to optimize, exposing only a few parameters to the optimization algorithm. The way each of that set of shown parameters affects the underlying surface depends on the exact wrapper definition. For example, a wrapper can expose a

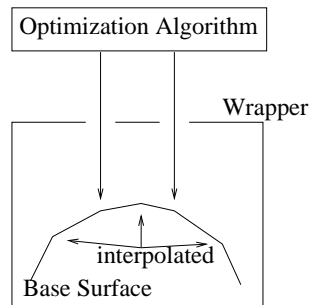


Fig. 15. A wrapper shows a selected set of vertices, interpolating the others.

certain region of the underlying reflector for the optimizer to focus on this area, or it can show a selected set of vertices, interpolating the others in a way transparent to the rest of the system. See Figure 15.

As mentioned above, a wrapping strategy was introduced both in order to avoid a quadratic growth in the number of vertices at each iteration (see next subsection), and to reduce the search space for the global optimization algorithm, because adding too many vertices at one time introduces too many variables to be handled and thus forces the algorithm to perform searches in a broader optimization space. This strategy consists on iteratively adding only a few vertices until convergence is achieved or a new change in resolution is needed, which occurs when there are no vertices left to add.

We can analyze the two components of this wrapping strategy separately:

- *The surface wrapper:* this component consists of a basic wrapper which is initialized with the vertices optimized at the previous *resolution* of the reflector (function “*wrap*” in the code above), and which interpolates the other vertices. In our current implementation we are using the well known Akima polynomial interpolator.
- *The addition of new vertices:* each time the surface needs more flexibility (optimization has not converged with current set of vertices), new vertices are added by sorting the already used vertices according to their $Status_i$ (see equation 4), and choosing the N with worst values. For each of these, its surrounding vertices are selected from the FreeVertexList, and the m with maximum reflector free area coverage are added to the wrapper.

Using this strategy has a small drawback that must be taken into account: it introduces a limit to the minimum achievable $Error_{used}$ since there is a certain difference between the unwrapped surface and the surface after wrapping is imposed. See Section 9 for some results about this.

8.3.2. Global Optimization Algorithm

As mentioned earlier, traditional numerical optimization algorithms fail in solving this problem due to the high coupling among the different degrees of freedom (i.e.: the vertices of the reflector, see above), the noisy nature of the Monte Carlo evaluation of the radiance distribution (see Section 7) and the many local minima present in the shape of the function. Thus, we were forced to switch to a global procedure that performs a brute-force search in a subset of the search space. At the core of the algorithm, we conducted tests of the performance of each member of a family of reflectors obtained by iteratively combining the addition of an increment to each one of the vertices (see Section 9). For this to be practical, the size of this generated family of reflectors has to be kept manageable. Obviously, the desired accuracy given by these increments added to the vertices (trying to ensure that the family contains at least one reflector close to the desired target), and the size of such family are closely related.

This algorithm makes too many unnecessary computations, so a more intelligent version was developed, which sorted the vertices according to their $Status_i$ and put more effort where a larger error was found. This is possible as $Status_i$ can be seen as the vertex differential influence on the difference between the actual and desired output, but it must be remarked that this local measure of the error is only used to give a hint to the optimization algorithm on where to put more effort and by reducing the search space of possible reflectors to test, but the global optimization is still guided by a global criteria, as is the $Error_{used}$ mentioned before. The vertices were grouped in sets and the subset of vertices with worse behavior was more finely sampled. This can be done because $Status_i$ gives hints on the overall shape error in a vertex neighborhood, so we bias the importance of the optimization towards those regions that show the biggest deviations from the desired shape (see Subsection 8.2 and Figure 14).

The final search space was built from the set of η vectors $\{\eta^{base} + \sum_{i=1}^N d^i\}$, with d^i the vector whose components are $(d^i)_j = \delta_j^i k^j StepSize$. The values for k^j are taken from $k^j \in [-k_{MAX}^j, k_{MAX}^j]$, and k_{MAX}^j is the number of steps for each degree of freedom resulting from the $Status_i$ sorting ($k_{MAX}^i \geq k_{MAX}^j \equiv Status_i \geq Status_j$). In this equation, η^{base} is the parameter vector obtained in the previous iteration.

The brute force procedure described above is not enough when taking into account the noisy nature of the Monte Carlo algorithms used in the light simulation step (see Section 7), since a certain variance is associated with each evaluation of the **OutR** function. Thus, performing the brute force search and retaining only the best value is not right, since each evaluation has an error associated that cannot be disregarded. So, a final version of the brute force algorithm was developed which deals with this situation: each time a value is computed, its error is taken into account by comparing its value minus its error with respect to the best so far plus its respective error (remember we are min-

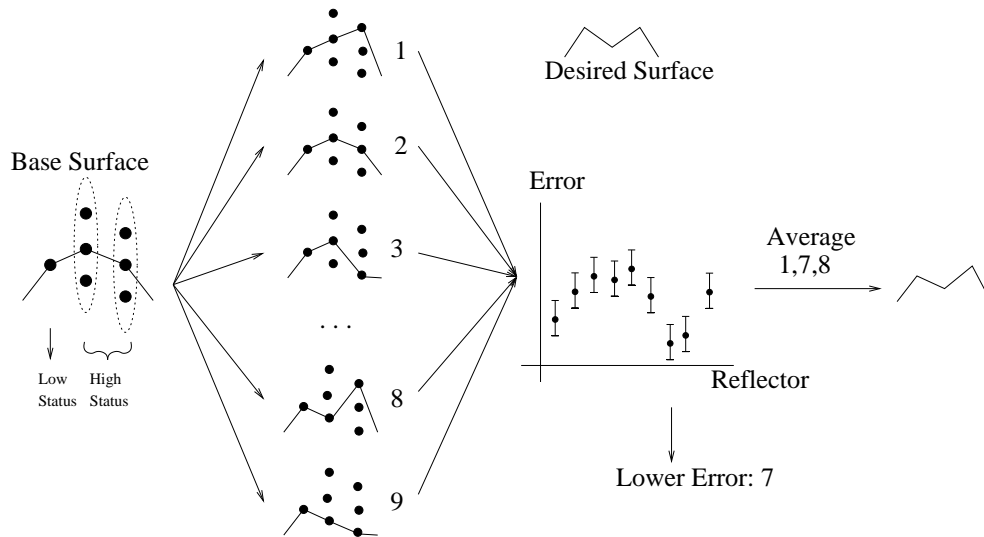


Fig. 16. The Optimization Algorithm: an overview.

imizing towards zero). That is, if $Error_{used}(\eta) - Error_{AtCurrentAccuracy} \leq Error_{used}(best) + Error_{AtBest'sAccuracy}$, the value is kept in a list that represents all values which may improve with respect to the best so far (considering the respective error bars). To obtain the error bars, the algorithm uses a pre-computed table of standard deviations indexed by the number of rays used in the light simulation. Finally, at the end of a Brute Force pass, the list retains a certain number of candidates whose error bars overlap with the best's one. To choose one, the list is progressively filtered at higher accuracy by increasing the number of fired rays at the light propagation computations, until only one value remains in the list or until we have reached our maximum tolerable accuracy. In the later case, we can consider the values as samples of the same ideal reflector, and average them to get the final value. See Figure 16.

The code is depicted in the following algorithm:

```

{ Brute Force Pass }
ErrorBar := Error Bar at Current Accuracy
For each reflector R in the restricted SearchSpace
  evaluate R
  if (Error_used(R) - ErrorBar <=
      Error_used(BestAtList) + ErrorBarForBest)
    add (R, ErrorBar) to List
{ Filtering Pass }
while not at maximum accuracy
  increase accuracy

```

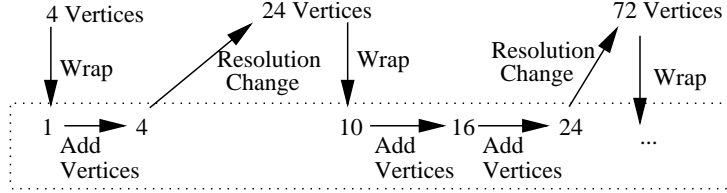


Fig. 17. Number of vertices at each iteration and resolution changes (axially symmetric reflectors). The dotted rectangle shows the actual reflector resolutions used for optimizations.

```

filter List at new accuracy
return (Average reflectors in List)

```

Each time the algorithm runs out of new vertices to add to the wrapper and the achieved result does not satisfy the convergence criteria, a change in the reflector resolution is needed. This is simply performed by doubling the grid resolution in both axes simultaneously, thus getting an overall quadratic behavior in the number of new vertices added at the outer loop in the algorithm described above. This brings about the need of a wrapping scheme to alleviate the effects of this growth, as depicted in figure 17.

9. Implementation Details

The system was built in C++ on top of SIR, the rendering kernel developed at the Universitat de Girona ³⁰.

With respect to the discretization used for *OutRad* (see Section 5), our current implementation uses $s = 34$ and $t = 25$ (lower hemisphere only) for *OutRad_{desired}* and $s = 34$, $t = 50$ for *OutRad_{lamp}* (the whole sphere of directions).

The value for ϵ in Equation 2 (the size each vertex i is moved when computing its respective Δ_i) was chosen to be of 5 millimeters, since choosing a value too small introduced variations that were of magnitude comparable to the Monte Carlo noise, and values too large produced too large variations that caused the values of $Status_i$ to be too similar for us to use them as a classification criteria.

10. Results and Conclusions

Our algorithm was tested against a set of test problems built with simulated data, as well as real $C - \gamma$ distributions used in industrial projects. For the former cases an isotropic light source was used, while a real one was used for the latter. The shapes of the test objective reflectors can be found in the left column at Figure 18. As can be seen in Table 1, it performed well for those problems, both with our fast but inaccurate light propagation method, and with the much more precise and general Light Tracing algorithm. Examination of this table shows that the iterations finished with reflectors that had between 10% and 14% of the initial error, after

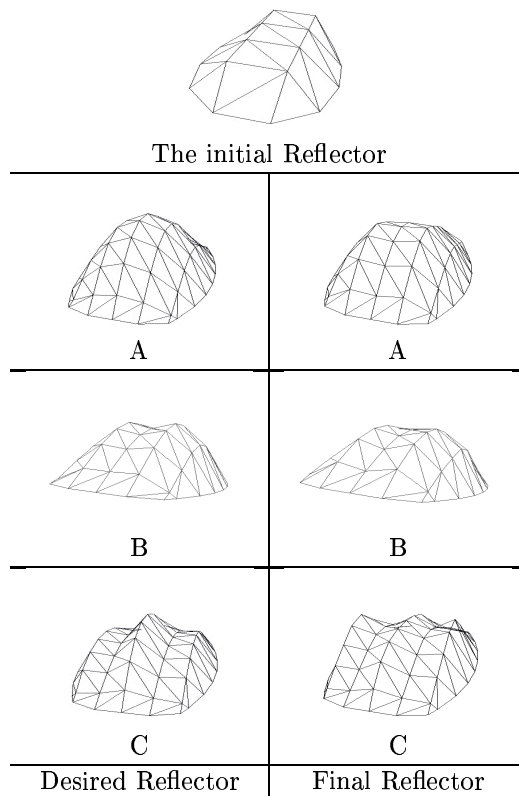


Fig. 18. Our desired objective reflectors (left column) and the ones obtained as final result of our optimization procedure (right column). The first row shows the starting reflector shape used in our algorithm. This starting reflector produces the light distribution shown in figure 19, while the distributions for the different test cases can be found in Figure 20

only two iterations of the inner loop, in the second iteration of the outer one, as described in Section 8.

One can see the initial light distribution and the final light distribution for these test cases in figures 19 and 20.

We also performed a test with varying BRDFs, ranging from a pure specular to an almost diffuse surface, see Table 2. The used BRDF is the Phong Model widely used in the Computer Graphics Rendering community, and the variations were performed by changing the diffuse and specular weights in the model. The first thing to notice is the rise in the variance in the function evaluations as the BRDF goes more and more diffuse, something that was expected. The consequence is that the more diffuse the surface, the more rays are needed to achieve a small increase in accuracy, which leads to forbiddingly high computing times for a very

diffuse surface. Also, for very diffuse reflectors, it makes the results of our algorithm untrustworthy, since the variance approaches the magnitude of the function value. In this context, we can see that it makes no sense to use *any* optimization algorithm for highly diffuse surfaces, as can be seen in the last row in Table 2.

	Initial $Error_{used}$	FnD: Final $Error_{used}$	LTR: Final $Error_{used}$	Iter.
A	1.8 ± 0.1	0.278 ± 0.004	0.17 ± 0.03	2/2
B	2.2 ± 0.1	0.287 ± 0.004	0.19 ± 0.03	2/2
C	2.0 ± 0.1	0.262 ± 0.03	0.16 ± 0.03	2/2

Table 1. Comparative Results for several experiments. Here, FnD refers to our fast but rough light propagation algorithm, while LTR is the full Light Tracing computation for this step. The “Iterations” column shows the number of outer-loop/inner-loop iterations needed until convergence was acceptable by user’s criteria.

Diff. to Spec. Coeff.	Initial $Error_{used}$	Final $Error_{used}$	ratio
0	0.25 ± 0.1	0.09 ± 0.09	2.8
0.05	0.3 ± 0.1	0.14 ± 0.04	2.1
0.44	0.6 ± 0.4	0.4 ± 0.4	1.5
1	0.8 ± 0.6	0.6 ± 0.6	1.3

Table 2. Comparative Results for different BRDFs, ranging from a pure specular one (first row) to a diffuse one (last row). The Diffuse to Specular coefficient is the ratio between those coefficients in a regular Phong BRDF formula.

	$Error_{used}(WrappedSurf)$ using the unwrapped surface as reference
A	0.21 ± 0.01
B	0.23 ± 0.01
C	0.13 ± 0.01

Table 3. Errors introduced when wrapping a surface, measured with respect to the same surface prior to the wrapping process (FnD).

It is important to mention that the process of wrapping a surface introduces a limit for the maximum achievable precision, since the wrapped surface cannot be equal to the original, however similar. In Table 3 it is possible to see the above mentioned limits for our test configurations in the case of our rough light propagation method, and this values should be taken into account when setting an objective for

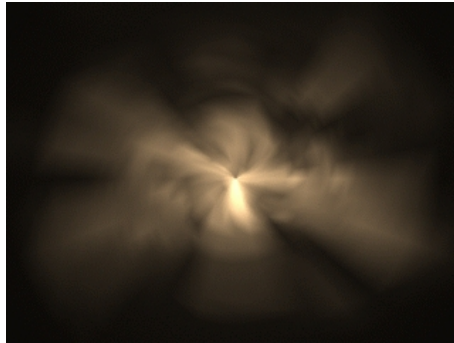


Fig. 19. Initial Light Distribution on a diffuse flat surface.

optimization. For this reason we can conclude that, at the level of approximation used, the final values in Table 1 cannot be expected to be lower than the values shown in Table 3 and are, in this light, very acceptable.

With respect to the previous work done in the field, and to the best of our knowledge, this is the first algorithm general enough to deal with generic BRDFs, interreflections inside the optical set and a piecewise polygonal approximation to the surface, thus performing any comparison with our results is difficult, if not impossible, and probably meaningless.

From our experiments we can conclude that the presented algorithm is stable and robust for pure specular and highly glossy reflector surfaces, while its convergence diminishes as the BRDF goes diffuse. This happens because diffuse BRDFs tend to blur the information out, making it impossible for the optimization algorithms to distinguish the dominant features they are trying to determine. This phenomenon can easily be predicted by studying the behavior of the variances for a given BRDF with respect to the number of fired rays.

From the above, we can conclude that our objective of presenting a first feasible algorithm able to treat the full inverse reflector design problem (general BRDFs, Interreflections, general light bulbs, etc), and taking into account industry restrictions, has been achieved.

The main drawback of our current solution is its speed (execution times are in the order of days), but ways of improving its performance are clear (see next Section). Nevertheless, our solution in its current form is already useful for light designers since the traditional methods take several weeks to provide a solution.

11. Future Work

An extremely promising way of restricting the search space, and thus reducing execution time, is to involve the designer into the search process, using his knowledge to provide guides for the search algorithm.

is propagated from an isotropic point light source into the environment via a curved reflector by means of the use of a Layered Depth Cube (LDC)²⁶ associated with the reflector. This LDC would be used to store the radiance coming from the reflector into the scene and treated, in a second step, as a new, non-isotropic secondary source. The resulting illumination would be the sum of the direct contribution plus the contribution of all secondary sources. Other promising possibilities include various forward radiosity approaches^{7 19}, which include the works by Christensen et al.³ and Slussallek et al.⁴³ which present different representations and algorithms for radiance calculations.

Alternatives to our Brute Force minimization method include, among others, the well known Genetic Algorithms (GA) approach, that, along with Simulated Annealing, are the only two widespread algorithms for global optimization of general functions.

Several possibilities are offered for the shape definition, among which the splines²¹ and a hierarchy of regular grid representations seem to be the most promising alternatives. Other options include a wavelet representation, multi-resolution surfaces and irregular grids. It would be interesting to explore the advantages of those methods.

Another line that should be studied is the possibility of using adaptativity in the $C - \gamma$ resolution, using a coarser resolution at the beginning, low-res reflectors, and a finer one for later iterations. Also, adaptativity in the usage of the rendering algorithm can be looked at: first, using our approximate but fast light propagation algorithm, leaving for a later stage a 2nd pass with the more costly Light Path Tracing-based algorithm.

Acknowledgments

We would like to thank the anonymous reviewers for their useful comments. This work was supported by grants TIC2001-2226-C02-02 and TIC2001-2392-c03-01 of MCyT (Spain) and 2001SGR0296 of DURSI of Generalitat de Catalunya.

References

1. Ashdown, I. 1994. Non-Imaging Optics Design Using Genetic Algorithms *Journal of the Illuminating Engineering Society*, 23(1):12-21 (Winter).
2. Blinn, J. F. and Newell, M. E., Texture and Reflection in Computer Generated Images *CACM*, 19(10):542-547, Oct. 1976
3. Christensen P. H., Lischinski D., Stollnitz E. J. and Salesin D. H., Clustering for Glossy Global Illumination. *ACM Transactions on Graphics*, 16(1):3-33, 1997.
4. Cardoso Costa, A.; Augusto Sousa, A. and Nunes Ferreira, F. Optimisation and Lighting Design WSCG '99 (Seventh International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media), pp. 29-36, 1999.
5. Cardoso Costa, A.; Augusto Sousa, A. and Nunes Ferreira, F. Lighting Design: A Goal Based Approach Using Optimization. *Rendering Techniques '99 (Proceedings of the 10th Eurographics Workshop on Rendering)*, pp. 317-328, 1999.
6. Coaton, J. R. and Marsden, A.M *Lamps and Lighting* Ed. Arnold, London, 1997.

24 Patow, Pueyo, Vinacua

7. Cohen M., and Wallace J., *Radiosity and Realistic Image Synthesis*. Academic Press Professional, Cambridge, Massachusetts, 1993.
8. Deng, H. L., Gouveia W. and Scales J. A. The CWP Object-Oriented Optimization Library Available from <http://timna.Mines.EDU/cwpcodes/coool/>
9. Debevec, P.; Hawkins, T.; Tchou, C.; Duiker, H-P; Sarokin, W. and M. Sagar Acquiring the Reflectance Field of a Human Face Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 2000), pp. 145–156, August 2000.
10. Deville, P. M. *Modélisation et Simulation des Propriétés Radiatives des Sources Lumineuses*. PhD Thesis, Université Henry Poincaré, Nancy I, France, 1996. (In french)
11. Deville, P.M.; Merzuk, S.; Cazier, D. and Paul, J. C. Spectral Data Modeling for lighting applications. *Computer Graphics Forum*, 13(3):97–106, September 1994, Oslo, Norway.
12. Deville, P.M. and Paul, J. C. Modeling the Spatial Energy distribution of Complex Light Sources for Lighting Engineering. *Rendering Techniques '95* (Proceedings of the Sixth Eurographics Workshop on Rendering), Springer-Verlag, NY, pp. 147–159, June 1995.
13. Doyle, S., D. Corcoran, and J. Connell. Automated Mirror Design Using an Evolutionary Strategy *Optical Engineering* 38(2):323-333, February 1999.
14. Doyle, S., D. Corcoran, and J. Connell Automated Mirror Design for an Extended Light Source Nonimaging Optics: Maximum Efficiency Light Transfer V Proc. SPIE Vol. 3781, pp. 94-102, 1999.
15. Elmer, W. B. *The Optical Design of Reflectors*. Ed. John Wiley and Sons, New York/Chichester/Brisbane/Toronto, 1978.
16. Heinz W. Engl and Andreas Neubauer, *Reflector Design As an Inverse Problem*, Heiliö M. ed, Proceedings of the Fifth European Conference on Mathematics in Industry, pp.13-24, Teubner, Stuttgart, 1991.
17. Boivin, S and Gagalowicz, A. Image-Based Rendering of Diffuse, Specular and Glossy Surfaces from a Single Image. *Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 2001)*, August 2001.
18. Gilbert, J. Ch., Nocedal, J. Global Convergence Properties of Conjugate Gradient Methods for Optimization. *SIAM J. On Optimization*, Vol.2, No. 1, pp.21-42. February 1992.
19. Glassner, A. *Principles of Digital Image Synthesis*. volumes 1 and 2. Morgan Kaufmann Publishers, 1995.
20. Harutunian, V., Morales J. C. and Howell J. R. Radiation Exchange within an Enclosure of Diffuse-Gray Surfaces: The Inverse Problem. appeared in *Inverse Problems in Heat Transfer* ASME/AIAA International Heat Transfer Conference, August 5-9, 1995, Portland, OR
21. L Piegler and W Tiller *The Nurbs Book*. Springer-Verlag Berlin Heidelberg, New York, 1995
22. Lafortune, E. *Mathematical Models and Monte Carlo Algorithms for Physically Based Rendering*, PhD thesis, Department of Computer Science, Katholieke Universiteit Leuven, 1996.
23. Kawai, J. K.; Painter, J.S. and Cohen, M. Radioptimization - Goal Based Rendering. *Computer Graphics, Siggraph'93 Proceedings*, pp. 147-154, 1993.
24. Keitz, H. A. E. *Calculos y Medidas en Luminotecnia*. *Biblioteca Técnica Philips*, Ed. Paraninfo, 1993. In spanish.
25. Sergey A. Kochengin and Vladimir I. Olikier *Determination of Reflector Surfaces from near-field Scattering Data II. Numerical Solution*, *Numer. Math.*, vol 79, num 4, pp.553-568, 1998.

26. Lischinski, D. and Rappoport, A. Image-based rendering for non-diffuse synthetic scenes. *Rendering Techniques 1998*, pp. 301-314.
27. Loscos, C; Drettakis, G and Bounoux, S. Interactive Virtual Relighting and Remodeling of Real Scenes. *Rendering Techniques'99*, pp. 329-340, NY, 1999.
28. Neubauer, A. The Iterative Solution of a Nonlinear Inverse Problem from Industry: Design of Reflectors. In *Curves and Surfaces in Geometric Design*, P. J. Laurent, A. Le Méhauté and L. L. Schumaker editors, A. K. Peters, Boston, pp.335, 1994.
29. Marschner, S. R. Inverse Rendering in Computer Graphics PhD. thesis, Program of Computer Graphics, Cornell University, Ithaca, NY, 1998.
30. Martin, I.; Perez, F. and Pueyo, X. The SIR Rendering Architecture. *Computers and Graphics*, vol. 22, No. 5, pp. 601-609, 1998.
31. Patow, G. A fast approximation to the illumination from curved reflectors. *Siggraph Technical Sketches*, pp.149, Aug 1997
32. Patow, G. and Pueyo, X. A Survey on Reflector Design Problems Submitted for publication. Also available as Universitat de Girona, TR-IIIA 02-07-RR, and from <http://ima.udg.es/~dagush/surv.ps.gz>
33. Poulin, P. and Fournier, A. Lights from highlights and Shadows. *Proceedings of the Symposium of 3D Interactive Graphics*, March 1992.
34. Poulin, P. and Fournier, A. Painting Surface Characteristics. *Rendering Techniques 1995*, Eurographics, pp. 119-129, June 1995.
35. Poulin, P., Ratib, K. and Jacques, M. Sketching Shadows and Highlights to Position Lights. *Proceedings of the Computer International'97 conference*, June 1997.
36. Press W. H.; Teukolsky, S.A.; Vetterling, W.T. and Flannery B. P. Numerical Recipes in C : The Art of Scientific Computing Cambridge University Press, 1992.
37. Press, W. H. and Teukolsky, S.A. Numerical Calculation of Derivatives. *Computers in Physics*, Jan/Feb, pp.68-69, 1991.
38. Press, W. H. and Teukolsky, S.A. Simulated Annealing Optimization Over Continuous Spaces. *Computers in Physics*, Jul/Aug, pp.426-429, 1991.
39. Radiant Imaging Photometric URL: <http://www.radimg.com/>
40. Ramamoorthi, R and Hanrahan, P A Signal Processing Framework for Inverse Rendering Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH 2001), August 2001.
41. Schoenemann, C., Dorsey, J., Smits, B., Arvo, J. and Greenberg, D. Painting with Light. *Computer Graphics*, Siggraph'93 proceedings, pp. 143-146, 1993.
42. Shewchuk, J. R. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. Available from <http://www.cs.cmu.edu/~jrs/>
43. Stamminger, M.; Slusallek, Ph. and Seidel, H.-P. Three Point Clustering for Radiance Computations. *Rendering Techniques '98* (Proc. 9th EUROGRAPHICS Workshop on Rendering '98), Springer-Wien, 1998.