Universidad de Granada
Escuela Técnica Superior de Ingeniería Informática
Departamento de Lenguajes y Sistemas Informáticos

Efficient Monte-Carlo Methods for Global Illumination.

PhD. dissertation by
Carlos Ureña Almagro

Advisors
Juan Carlos Torres Cantero.
Xavier Pueyo Sández.

Granada
June 21, 1998

Acknowledgements.

I wish to thank my advisors, Juan Carlos Torres Cantero and Xavier Pueyo Sandez, for their encouragement and advisement through the production of the results here described, and also for their continuous support to the research on the field of Computer Graphics.

I also wish to thank the people involved in the development of the GIRT system, which I used to test the algorithms described. These are: Jorge Revelles Moreno, Pedro Cano Olivares, Marcelino Cabrera Cuevas, Juan Carlos Torres and Vicente del Sol Lopez. Antonio López Fernández y Miguel Lastra Leidingher did produce useful pieces of code. I also wish to thank all of them the time spent in keeping the computer equipment running correctly.

I wish to thank the *Departamento de Lenguajes y Sistemas Informáticos* at Granada University, to allow me spending a part of my time in Computer Graphics research, and also to use its equipment to produce this texts.

This work has been partially supported by a grant coded as TIC95-0614-C03-02, from the Committee for Science and Technology of the Spanish Government, and also by a Joint Action coded as HU96-37.

Parts of these contents have been previously published [Urena97, Urena97a]. We have also published articles which describe our first rendering system [Urena92, Urena93] and our current system (named as GIRT) [Urena97b]. I would like to thank the members of the Computer Graphics research group at the University of Granada their efforts in the production of those papers, and also to Jose Parets his support in the subject of Object-Oriented programming.

This is dedicated to my parents, Antonio Ureña Muñoz y Mª del Carmen Almagro Pérez, and specially to my wife, Yolanda, who knows so well the effort spent in writing this, and gives it sense.

Contents

List of Figures

List of Tables

List of Symbols.

A	Area measure in S, the scene objects surfaces. See section 1.1. In section 1.5 it is used to mean the square array obtained after discretization of the transport operator $\mathcal{A}$. It is related to the *form factor* matrix and its entries are c_{ij} (see definition 1.91).
$\mathcal{A}$	Linear integral operator which acts over integrable functions in the domain D. It is fully determined by the function k, as can be seen in (1.43).
a	Absorption state. Is an arbitrary entity such that $a \notin D$.
B	Function with domain S and real values, named the *radiosity* function. It obeys the integral equation (1.41) and is equal to radiance when this is independent of outgoing direction.
B'	Function with domain S and real values. It is obtained as the projection of B into an orthonormal basis, as can be seen in (4.1) (see chapter 4).
$\mathcal{B}$	Arbitrary operator acting on functions on the domain D.
$\mathbf{B}$	Arbitrary set of n orthonormal basis functions in the domain D.
B_e	Function with domain S and real values. The value $B_e(x)$ is the part of $B(x)$ due to radiation emission from point x.
B_r	Function with domain S and real values. The value $B_r(x)$ is the part of $B(x)$ due to reflected radiance in x.
b_i	Functions with domain D and real values, elements of the basis set $\mathbf{B}$.
C	The set of all infinite chains whose states are elements of D'.
$\mathbf{C}$	A set of orthonormal basis functions with domain D, such that each of them is constant (see section 2.4).
C_i	Functions with domain D, belonging to the basis function set $\mathbf{C}$ (see equation 2.3).
c_{ij}	Entries of the array A. They are defined as $c_{ij} = \langle \mathcal{A}b_i \mid b_j \rangle$.
$\cos$	Function with domain $O \times O$ and real values. The value $\cos(a,b)$ is the cosine of the interior angle between the two unit length vectors a and b. It is equal to the inner product of a and b.
D	The set of all rays whose origin is in S. Formally, it is defined as $D = S \times O$. It is the domain of the radiance function and other related functions. See chapter 1.

D' Equal to the set D extended in order to include the absorption state, that is $D' = D \cup \{a\}$.

$\mathcal{D}$ The diffuse transport operator. It is the operator which models diffuse reflection, that is, the operator induced by the BRDF f_{rd}.

$\mathbf{D}_i$ Orthonormal basis of the vector space V_i (see section 2.4).

E Function with domain S and real values. The value $E(x)$ is the *irradiance* at x, measured in Watts per square meter (see definition (1.4). In chapter three, it is applied to random variables: for any random variable X, the value $E(x)$ is its expected value or mean.

E_n Function with domain D and real values. For any element r in D and a natural number n, the value $E_n(r)$ is the mean of the random variable X_r^n (see section 3.2). The most frequently used are E_1 and E_2. In chapter 4, this symbols denotes a function with domain S and real values. The value $E_n(x)$ is the part of $E_g(x)$ due to radiation coming from patch number n (see definition 4.19).

E'_n Function with domain D and real values. For each element r in D and natural n, the value $E'_n(r)$ is the mean of the random variable Y_r^n /see section 3.3).

E_g Function with domain S and real values. The value $E_g(x)$ is the exact result of *final-gather* over x (see definition 4.3). It is an approximation to the irradiance at x, $E(x)$.

E'_g Function with domain D and real values. The value $E'_g(x)$ is an estimator of $E_g(x)$, and it is obtained after Monte Carlo sampling (see definition 4.6).

F Column vector of the coefficients of f with respect to the orthonormal basis $\mathbf{B}$.

f Function with domain D and real values, it is the solution of the integral equation (1.27). This symbol is frequently used to mean an arbitrary function, and in that cases this is explicitly stated so.

f' Function with domain D and real values, it is the solution to the (discretized) function equation (1.85).

f_i Entries of the vector F. They are the coefficients of the function f' with respect to the base $\mathbf{B}$, so that $f_i = \langle f' \mid b_i \rangle$ (see chapter 1). In chapter 3, this symbol is used to denote the function obtained by applying n times the operator $\mathcal{A}$ to f, that is $f_i = \mathcal{A}^i f$. We are interested in f_1 y f_2.

f_r Function with domain $S \times O \times O$ and real values. It is called the *Bidirectional Reflectance Distribution Function* or simply BRDF (see definition 1.10).

f_{rd} Diffuse BRDF, is the part of f_r due to diffuse reflection (see definition (1.21)).

f_{rp} Diffuse-specular or *Phong* BRDF. It is the part of f_r due to Phong-like reflection (see definition (1.23)).

f_{rs} Perfect specular BRDF. It is the part of f_r due to perfect specular reflection (see definition (1.20)).

G Column vector with the coefficients of g with respect to the base $\mathbf{B}$ (see section 1.5). It is also used to mean a function with domain $S \times S$ and real values, which acts as a geometric term in the definition of the K function (see definition (1.32)).

$\mathcal{G}$ Radiance transport operator which models diffuse-specular reflection, that is, reflections induced by the diffuse-specular BRDF f_{rp}.

g Arbitrary integrable function with domain D and real values.

g_i Elements of the vector G, that is, $g_i = \langle b_i \mid g \rangle$, where b_i are the elements of the basis $\mathbf{B}$ (see section 1.5). In chapter 3, it is used to denote the function obtained after applying i times the operator $\mathcal{A}$ to the function g, that is $g_i = \mathcal{A}^i g$.

H Function with domain $S \times S$ and real values (see definition (1.29)).

h Arbitrary integrable function with domain D and real values.

I Function with domain D and real values, it is in the vector space spawned by the basis function set $\mathbf{W}$. The value $I(\mathbf{r})$ is the part of $L(\mathbf{r})$ which is directly perceived by an observer (see definition 2.2)).

$\mathcal{I}$ Identity operator. For any function f with domain D, it holds $\mathcal{I}f = f$.

K Kernel of the transport operator $\mathcal{T}$. It is a function with domain $D \times D$ and real values, defined in equation (1.35). The value $K(\mathbf{r}, \mathbf{s})$ is the radiance at $\mathbf{r}$ due to a unit of radiance at $\mathbf{s}$.

k Kernel of the transport operator A. Is an arbitrary function with domain $D \times D$ and real values.

l Function with domain D and real values. It is the solution to the integral equation (1.76), which in turn is the dual of (1.27). When l exists, it can be written as $l = (\mathcal{A}^*)^+ h$.

l_i Coefficients of the function I with respect to the basis function set $\mathbf{W}$. They are defined as $l_i = \langle I \mid W_{ei} \rangle$.

L Function with domain D and real values. For each $\mathbf{r}$ in D, the value $L(\mathbf{r})$ is the *radiance* at $\mathbf{r}$ (see definition (1.5)), and it is measured in Watts per square meter and per stereoradian.

L_r Part of L due to reflected radiance, it is called the *reflected radiance* function.

L_e Part of L due to emitted radiance, it is called the *emitted radiance* function.

L_i Function with domain D and real values. For each $\mathbf{r}$ in D, the value $L_i(\mathbf{r})$ is the incoming radiance at x from direction w, where $(x, w) = \mathbf{r}$. It is measured in Watts per square meter and per stereoradian (see chapter1).

M Function with domain S and real values. For each point $x \in S$, the value $M(x)$ is the radiant power per unit area leaving x in all directions, and it is measured in Watts per square meter. It is defined in equation (1.3).

M_e Function with domain S and real values. The value $M_e(x)$ is the part of $M(x)$ due to power emitted from x.

M_r Function with domain S and scalar values. The value $M_r(x)$ is the part of $M(x)$ due to reflected power at x.

m Arbitrary measure function in D, which assigns a scalar value to the subsets of D.

m' Extension of m to the domain D', which assigns unit measure to the absorption state.

m_K Essential supremum of the set of values of the function $\mathcal{T}u$, where u is the function equal to 1 for elements in its domain. The value m_K is equal to the essential supremum of the values $\rho(x, w)$ (see definition (1.65)).

m_k Essential supremum of the set of values of the function $\mathcal{A}u$, where u is the function equal to 1 for elements in its domain (see definition (1.58)).

n_x For any point x in S, the unit length vector n_x is the orthogonal vector to the surface S at x.

O The set of all unit length vectors in the three-dimensional Euclidean space, or the set of all the points in the unit radius sphere centered in the origin.

$\mathcal{P}_{\mathbf{B}}$ For any basis function set $\mathbf{B}$, $\mathcal{P}_{\mathbf{B}}$ is the projection operator which maps any function into its projection onto $\mathbf{B}$.

p Function with domain $D' \times D'$ and real values. The value $p(x, y)$ is the conditional probability for a chain to go to state x when the previous state is y. This symbol has also been used to mean a function with domain $S \times O$ and values in S. The point $p(x, w)$ is the first point in S visible from x in the direction w.

p_0 Function with domain D and real values. The value $p_0(x)$ is the probability for a chain to start at a state x in D.

p_x Function with domain D and real values. It is the probability density function (pdf) used in the final-gather process described in chapter 4. It obeys conditions (4.4) and (4.5).

Q_r For each $r \in D$, it is a probability measure defined in C', and obeys equality (3.88). This probability measure is used to define the random variable Y_r.

Q_h For each h (arbitrary function with domain D and real values), is a probability measure on C', which obeys equality (3.109) and which is used to define the random variable Y_h.

r_i Total radiant power landing in patch number i (see chapter 4).

r_{ij} Part of r_i due to power coming from patch number j (see chapter 4).

$refl_x$ Function with domain O and range O. For each point x in S and unit length vector w, the unit length vector $refl_x(w)$ is the *reflected vector* of w with respect to n_x (see chapter 1).

S The set of all points in the boundary of the scene objects, or the surfaces of those objects.

$\mathcal{S}$ Radiance transport operator which models perfect specular reflection, that is, reflection induced by the perfect specular BRDF f_{rs}.

$\mathbf{S}_i$ Orthonormal basis for the vector space U_i. (see section 2.4).

T Function with domain $S \times S$ and real values (see section 4.2), is defined as $T(x,y) = G(x,y)B'(y)$.

$\mathcal{T}$ Lineal integral operator which models radiance reflection. It is defined by the kernel K in equation (1.37), and also by the equation (1.25.

$\mathcal{U}$ Lineal integral operator which appears in the integral equations characterizing the variance functions V and V' (see definition 3.23).

U_r For each $r \in D$, it is a probability measure defined for all the subsets of C, which is defined by the relation (3.8), and which is used to define the random variable X_r.

U_h For each h (an arbitrary function with domain D and real values), is a probability measure defined on the subsets of C, which is defined by (3.54) and is used to define the random variable X_h.

V Function with domain $S \times S$ and real values. The value $V(x,y)$ is 1 when the points x and y are mutually visible, zero otherwise (usually called the *visibility function*). In chapter 3 this symbol also denotes a function with domain D and real values. Here, the value $V(r)$ is the variance of the random variable X_r.

V_e Function with domain D and real values. Is the source term in the integral equation defining the variance function (see definition (3.41)).

V' Function with domain D and real values. The value $V'(r)$ is the variance of the random variable Y_r.

w_{xy} A unit length vector. For each two different points x and y in S, w_{xy} is the unit length vector from x to y, that is, $w_{xy} = (y-x)/|y-x|$.

W Function with domain D and real values. The value $W(\mathbf{r})$ is the fraction of $L(\mathbf{r})$ that hits an observer directly or after an arbitrary number of reflections. It can be obtained as the sum of every W_i.

$\mathbf{W}$ A set of orthonormal basis functions in D, which models an observer. Its elements are called W_{ei} (see chapter 2).

W_{ei} Function with domain D and real values, belonging to the set $\mathbf{W}$. The function W_{ei} is called the *emitted potential* related to the i-th element of the observer. The value $W_{ei}(\mathbf{r})$ is the fraction of $L(\mathbf{r})$ which directly hits the i-th element of the observer (see chapter 2).

W_i Function with domain D and real values. The function W_i is called the *potential* related to the i-th element of the observer. The value $W_i(\mathbf{r})$ is the fraction of $L(\mathbf{r})$ which directly or indirectly (after a number of reflections) hits the i-th element of the observer (see chapter 2).

X_r Random variable defined for infinite chains in C (see definition (3.9)). The subindex r denotes an arbitrary element of D which is involved in the definition of the random variable.

X_h Random variable defined for infinite chains in C (see definition (3.51)). The subindex h denotes an arbitrary function, with domain D and real values, which is involved in the definition of the random variable.

Y_r Random variable defined for infinite chains in C' (see definition (3.91)). The subindex r denotes an arbitrary element of D which is involved in the definition of the random variable.

Y_h Random variable defined for infinite chains in C' (see definition (3.110)). The subindex h denotes an arbitrary function, with domain D and real values, which is involved in the definition of the random variable.

Z_r Random variable defined on D (see definition (3.92)). It is the random variable used for direct source sampling.

δ The Dirac delta function. The integral $\int_I \delta(x)f(x)dx$ is equal to $f(0)$ when $0 \in I$, and 0 in any other case.

Δ Function with domain S and real values. The value $\Delta(x)$ is the error involved when approximating $E_g(x)$ by r_i, where i is the index of the surface patch containing x (see definition 4.21).

Δ_j Function with domain S and real values. The value $\Delta_j(x)$ is the error involved when approximating $E_j(x)$ by r_{ij}, where i is the index of the surface patch containing x (see definition 4.20).

Φ Measure function defined for all the subsets of S. For any region $S' \subseteq S$, the real value $\Phi(S')$ is the total power leaving from S'. It is measured in Watts (see chapter 1).

Φ_r Measure function on S. Is the part of Φ due to reflected power.

Φ_e Measure function on S. Is the part of Φ due to reflected power.

Φ_i Total power leaving node number i in the cluster-patch hierarchy described in chapter 5.

σ Measure function in O. Is the solid angle measure in the sphere. For any $O' \subseteq O$, the real value $\sigma(O)$ is the total solid angle covered by O'.

σ_x Measure function in O. The subindex x denotes a point in S involved in its definition. This measure is defined by its relation to the measure σ, because it obeys $d\sigma_x(w)/d\sigma(w) = cos(n_x, w)$. It is called *projected solid angle* measure.

μ Measure function in D. It is obtained as the composition of area measure A and projected solid angle measure σ_x. For any $\mathbf{r} = (x, w) \in D$, it holds that $d\mu(\mathbf{r}) = d\mu(x, w) = dA(x)d\sigma_x(w)$.

ρ Function with domain D and real values (between 0 and 1). It is called *directional-hemispherical reflectivity*. The value $\rho(x, w)$ is the total power reflected from x in all directions, due to a unit of radiance coming from direction w (see definition 1.13).

ρ_d Function with domain D and real values (between 0 and 1). The value $\rho_d(x)$ is the fraction of power diffusely reflected at x.

ρ_p Function with domain D and real values (between 0 and 1). The value $\rho_p(x)$ is an upper bound of the fraction of power reflected at x in the way modeled by the BRDF f_{rp}.

ρ_s Function with domain D and real values (between 0 and 1). The value $\rho_s(x)$ is the fraction of power specularly reflected at x, in the way modeled by the BRDF f_{rs}.

Introduction.

This thesis is devoted to the improvement of Monte-Carlo techniques for the computational solution of the integral equation which characterize the light transport. Although this seems to be a rather technical statement, the basic purpose behind these techniques is very simple: the creation of realistic images of real world objects. In fact, this has been the target of two many efforts in science and art, from the very beginning of the human being. Note that the first visible results of human culture are tools for hunting and pictures on the walls of the caverns.

Through the history of the art, we can see how artists approach, with a growing level of achievement, the creation of realistic images. Some of the techniques used where never explicitly expressed by the artists, and thus couldn't be transfered to other people. The first explicit expressions of these concepts where done during the five-teen and sixteen centuries. In this age, the geometric laws governing perspective projections where found, which allowed the production of pictures showing objects which where perceived at different depths. This effort on depth perception also produced the *sfumato* technique. Here, the attenuation of light intensity (due to atmospheric scattering) was also used to emphasize depth perception in landscapes. Later, at the beginning of the nineteen century, the fusion of color in the human visual system was recognized and used to produce colors pictures composed of simple dots in a small set of basic colors, without mixing the oils.

All these are just a few examples of how the study of human perception, geometry, and physics is needed for the creation of realistic looking images. It can be argued that the invention of photography made un-useful this knowledge. But it is still necessary the creation of this kind of images, for example when the object being depicted still does not exists, or no photos can be taken of it.

As the use of computers do help in almost all of human activities, it is no surprise that they are also used to produce images. Computer Graphics is the branch of Computer Science devoted to the creation and processing of images and animations by using computers. Creation of realistic images is not by far the only goal of Computer Graphics techniques, but this has always been included as one of its subjects. It has been usually named as *rendering* or specifically *realistic rendering*. Realistic rendering is very related to others Computer Graphics subjects, such as modeling, image storage and display, and software engineering for graphics. But rendering also uses results gathered from physics, mathematics, and from the study of the human visual system.

At the beginning of Computer Graphics, the first problem solved (directly related to rendering) was the perspective display of wire-frame models of objects. The next step towards realism was to remove the pieces of the objects not seen because of occlusions by nearer opaque objects (this is called the visibility prob-

lem). Efficient techniques for these goals where proposed and implemented. But the images obtained did not seem realistic because of the use of constant intensity of colors inside the piece of the image covered by the projection of each object. It was soon recognized the need for some kind of simulation of shading, which would aid in the perception of objects shape and relative position. Some simple techniques where proposed in order to produce shading. These techniques where selected because they are simple enough for the available computing power while producing acceptable levels of realism.

As computing power increases, the above techniques where used to produce animations and static images in very short time. The simplicity of the techniques allowed hardware implementation of the involved algorithms. All this led to interactive, real-time display of animations, which has a huge number of applications.

At the same time, other researchers focused on the creation of more realistic looking images. It was perceived that matching real images can only be achieved by paying the necessary attention to the laws governing light transport. These equations are usually stated as an *Fredholm Integral Equation*, and are used to derive algorithms for realistic rendering. The equations where previously used to model radiation heat transfer and the flux of atomic particles, such as neutrons.

Two families of methods are used in order to solve those equations: Finite-Element methods and Monte-Carlo methods. The first are based in the well known finite elements technique, which has been previously used in other contexts for the solution of integral and differential equations. In essence, this technique is based on the reduction of continuous integral equations to discrete matrix equations (also called systems of equations). This can be approximately solved in finite time, and the solution can be stored in a finite amount of computer memory.

Monte-Carlo methods are ultimately rooted in the use of Markov Chains to compute the solutions of integral equations. This can be seen as an extension of the basic Monte-Carlo technique for computing integrals. As in all Monte-Carlo sampling algorithms, it is necessary to define a random variable whose mean is the value we want to compute. Then, this random variable is sampled, and an approximation to its mean is obtained by averaging a sufficient large number of samples.

Both kind of methods offers some advantages and suffer some disadvantages. Finite-Element methods are usually faster for low and medium complexity environments, or when a not so exact solution is desired. Monte-Carlo performs relatively worse in these cases, but this kind of algorithms are not so sensitive to an increase in complexity or desired accuracy. Thus, Monte-Carlo method has been used for solving complex environment, with complex reflection behavior, while Finite-Element methods have been less used in these cases. In spite of this, Monte-Carlo algorithms are still very slow when faced to this problem. The creation of realistic image cannot be done, by far, in real time, and lengthly, resource-consuming processes are still used.

In this dissertation, we propose techniques which try to improve the efficiency of Monte-Carlo algorithms. This lack of efficiency comes mainly from the variance involved in them. The variance is a measure about how close will be the solution obtained to the exact one. Then, the goal is to design algorithms whose underlying

variance is a small as possible. A correct understand of the nature of the variance is needed. This work includes a characterization of the variance of a wide set Monte-Carlo algorithms for solving environments with arbitrary reflection behavior. We find that the variance is characterized by an integral equation. Based on these results, we derive the *ideal* sampling strategy for Monte-Carlo algorithms (that is, the strategy producing zero variance). Finally, we show how to apply these results to produce two-step Monte-Carlo algorithms with reduced variance.

The structure of the thesis is as follows. In chapter one, the integral equations for radiance and radiosity transport are introduced. Based on this equations, in chapter two we state the problem of global illumination and classify the existing algorithms solving it. This is a comparative study which classifies the huge amount of papers in a small set of categories, highlighting the differences and similitude between them. Chapter three focuses on Monte-Carlo algorithms: we study their variance and we show the ideal sampling strategy. Finally, chapters four and five are devoted to the implementation of more efficient Monte-Carlo algorithms, based on the results shown in chapter three. First, a implementation restricted to diffuse environments is presented, and after this, an algorithm for environments with arbitrary reflection functions is proposed.

Chapter 1

Light Transport and related Computational Methods.

1.1 Introduction.

In this chapter, we describe the quantities which model light transport, and we derive the equations relating them. First, we introduce the model for the environment where energy flux takes place, and then the domain of the functions and the measures used in the integrations. By using these mathematical tools, it is easy to derive those relations. These derivations can be found in the literature. We wanted to include them here in order to introduce the notation while showing its coherence. Furthermore, we give a slightly different expression for the rendering equation. This expression has exactly the form of a Fredholm equation on the second kind (on the ray space), which makes easier the introduction of various numerical algorithms solving it.

In this chapter we will use a number of mathematical tools in order to introduce relevant concepts[1]. Detailed introduction of those tools is beyond the scope of this dissertation, and readers interested in any of them can resort to any mathematical text or dictionary of mathematics, such as [Ito93].

1.1.1 Environment model.

Let us assume an environment composed of a number of object in the three-dimensional space. At some of the points in the surface of those objects, energy is emitted in the form of electro-magnetic waves, with constant intensity along time. This waves are scattered at other surfaces. We consider that equilibrium state has been reached in the environment, in the sense that the temperature of all the objects does not changes, and the intensity of radiant flux also remains constant at

[1]Some (but not all) of those tools are: measure spaces, measure functions, probability measure functions, Lebesgue integral, vector spaces and basis set.

all points. Surfaces are all opaque, so no particle travels through any of them (all of it is either reflected or absorbed, but not refracted).

No participating media there exists in the space between the objects, which is vacuum. Thus, any point is either in vacuum or inside a object. Those objects are compact subsets of the three-dimensional space. Furthermore, their surfaces are continuous (except at a null measure subset), and thus, a normal vector exists at all of surface points.

Electro-magnetic waves have a single wavelength. All quantities describing flux intensity are wavelength-dependent, so this dependence will be implicit in our formulations, and the wavelength does not appears in them.

We will consider just closed environments. When faced to an open environment, we can include in it an additional perfectly black surface, which surrounds all the objects. This modified closed environment has exactly the same behavior that the original open one, because waves or particles leaving the open environment are absorbed on the surrounding surface in closed one, and the flux impinging on the rest of surfaces is equal in both of them.

1.1.2 Domain and Measures

We call S to the set of all surface points of the objects described above. For all points $x \in S$, there exists a normal vector to the surface at that point, which will be written as n_x. This implies that on we can use the area measure on S, which we will call A. If $S' \subseteq S$, then $A(S') = \int_{S'} dA(x)$.

The functions we are interested in are defined on the space of direction vectors and points. That is, we define the domain $D = S \times O$, where O is the set of unit directions vectors.

The set O has defined the well known solid angle measure σ. For any region A in the sphere, the value $\sigma(A)$ is the area of that region, as can be seen in figure 1.1, where the region A is shaded. For each point $x \in S$ we can define a projected solid

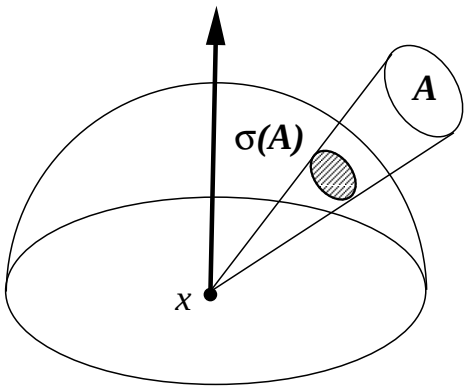

Figure 1.1: The measure σ for a set of directions A.

angle measure on O. This measure is called σ_x. For any unit vector $= O' \subseteq O$, we have the following definition of σ_x:

$$\sigma_x(O') \;=\; \int_{O'_x} cos(w, n_x)\, d\sigma(w) \tag{1.1}$$

where $O'_x = \{v \in O' \mid cos(v, n_x) > 0\}$. That is, O'_x is the subset of O' which is on the hemisphere above x, as defined by n_x. Finally, σ is the known solid angle measure on O. This definition implies that directions in the hemisphere under the point have zero measure. In figure 1.2 it can be seen how (for a given set A of directions) the value $\sigma_x(A)$ is the area of the shaded region. This region lies in the unit radius circle under the hemisphere, and is obtained by a vertical projection of the hemisphere region covered by A

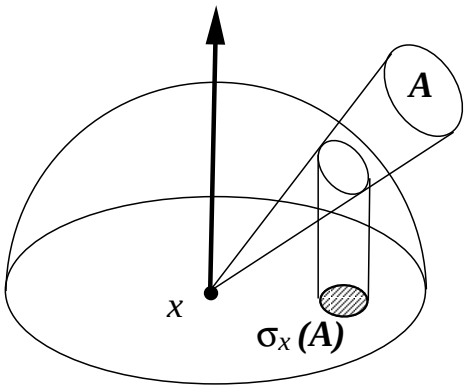

Figure 1.2: The measure σ_x for a set of directions A.

Once the measures A and σ_x have been introduced, we now define a measure μ on D which will be used later in the integral governing radiance transport. Lets consider any element $r \in D$. This implies that $r = (x, w)$, where $x \in S$ and $w \in O$. The measure μ is obtained as the composition of both previously introduced measures:

$$d\mu(r) \;=\; d\mu(x, w) \;=\; dA(x)\, d\sigma_x(w) \tag{1.2}$$

. This measure is called the *throughput measure* [Veach97], and is very often used in Global Illumination, and specially in this text.

1.2 Local light reflection.

1.2.1 Radiance and related functions.

For any subset S' of S, we can measure the total radiant energy leaving per unit of time (or power). This will be called $\Phi(S')$. This power is measured in Watts. The value $\Phi(S')$ can be obtained as the sum of two other functions: $\Phi(S') =$

$\Phi_e(S') + \Phi_r(S')$, where Φ_e measures the total radiant energy per unit of time that is generated on S', and $\Phi_r(S')$ measures the total energy that is reflected in S' due to incoming energy. We can also measure the total incoming energy onto S' coming from outside the surface. This functions is written as Φ_i. All these functions have the same domain and units.

Usually, the amount of energy leaving a surface is different from one point to another. In order to characterize this variation, we can use the *radiant exitance* function M, which is the power per unit area:

$$M(x) = \frac{d\Phi(x)}{dA(x)} \tag{1.3}$$

Where $x \in S$. This way we can define the emitted and reflected radiant exitance as $M_e(x) = d\Phi_e(x)/dA(x)$ and $M_r(x) = d\Phi_r(x)/dA(x)$. The *irradiance* function is defined as the amount of power incident on a surface point per unit area:

$$E(x) = \frac{d\Phi_i(x)}{dA(x)} \tag{1.4}$$

All these functions, M, M_r, M_e and E have units of Watts per square meter ($W\,m^{-1}$). Incoming power which strikes any surface point is either absorbed or reflected. In this later case, we assume that reflected power leaves the surface from the same point where the incoming power arrived. This implies that $M(x) = M_e(x) + M_r(x)$, where $M_r(x)$ is due to reflection of $E(x)$.

The value $M(x)$ measures energy leaving in all directions above x. We can measure the exitance leaving x in a given set of directions $O' \subseteq O$. If we call this value $M(x, O')$, then we view M as a measure function and we are able to define the *radiance* function, as follows:

$$L(r) = L(x, w) = \frac{dM(x, w)}{d\sigma_x(w)} \tag{1.5}$$

where $\mathbf{r} = (x, w)$. Radiance is a function whose domain is D, that is, is defined for every point and direction in D. The radiance function, evaluated at a point $x \in S$ and a direction $w \in O$ gives the power flowing away that point in that direction per unit of time, per unit of area perpendicular to the direction of flow, and per unit of solid angle. This function is measured in watts per stereo-radian and per square meter ($W\,sr^{-1}\,m^{-2}$), and is written as L. In consistency with previous notation, we define:

$$L_e(x, w) = \frac{dM_e(x, w)}{d\sigma_x(w)} \tag{1.6}$$

$$L_r(x, w) = \frac{dM_r(x, w)}{d\sigma_x(w)} \tag{1.7}$$

$$L_i(x, w) = \frac{dE(x, w)}{d\sigma_x(w)} \tag{1.8}$$

all these functions are also measured in $W\,sr^{-1}\,m^{-2}$. The above equalities can also be stated in an integral form:

$$M_r(x) \;=\; \int_O L_r(x,w)\,d\sigma_x(w) \qquad E(x) \;=\; \int_O L_i(x,w)\,d\sigma_x(w) \tag{1.9}$$

Note that we stated that $M_r(x)$ is due to reflection of $E(x)$. This relation does not holds between L_r and L_i. That is: energy hitting a point in a direction can be reflected in other directions. The relation between the energy which is scattered from one direction to another is governed by the *Bidirectional Reflectance Distribution Function* or simply BRDF.

In order to define the BRDF, we can view both $L_r(x,w)$ and $E(x)$ as measures in O'. This is done be considering the amount of irradiance coming from directions in any set $O' \subseteq O$ (lets call this $E(x,O')$. We can also measure the amount of reflected radiance at direction w which is due to energy coming from directions in O' (this is $L_r(x,w,O')$). The BRDF (noted as f_r) is defined as the relation between these two measures:

$$f_r(x,w_o,w_i) \;=\; \frac{dL_r(x,w_o,w_i)}{dE(x,w_i)} \tag{1.10}$$

With this definition, we see that the BRDF is the reflected radiance at direction w_o per unit of irradiance coming from direction w_i. From (1.8) we see that $dE(x,w) = L_i(x,w)d\sigma_x(w)$. Then, we can rewrite the definition of the BRDF:

$$f_r(x,w_o,w_i) \;=\; \frac{dL_r(x,w_o)}{L_i(x,w_i)d\sigma_x(w_i)} \tag{1.11}$$

By using the $BRDF$, we can express the value of L_r as the sum of incoming radiance from every direction. This is done by integration of (1.11) for all directions in O:

$$L_r(x,w_o) \;=\; \int_O f_r(x,w_o,w_i)\,L_i(x,w_i)\,d\sigma_x(w_i) \tag{1.12}$$

in figure 1.3 it can be seen how the reflected radiance L_r leaving a point is obtained as an integral over all hemisphere directions of the incoming radiance from each of them, as is derived from equation (1.12).

From the definition of BRDF, we define a related function, which is called the *directional-hemispherical reflectivity* function. This function gives, for every point x and direction w_i, the fraction of radiance impinging on x which is reflected on the whole hemisphere over x, and is called ρ. This can be seen as the value of $M_r(x)$ when a unit of irradiance comes just from w_i (that is, when $L_i(x,w) = \delta(w-w_i)$). By using this definition, we can substitute L_i in (1.12) and we obtain $L_r(x,w_o) = f_r(x,w_o,w_i)$. Now we can use (1.9) and we get:

$$\rho(x,w_i) \;=\; M_r(x) \;=\; \int_O L_r(x,w_o)\,d\sigma_x(w_o) \;=\; \int_O f_r(x,w_i,w_o)\,d\sigma_x(w_o) \tag{1.13}$$

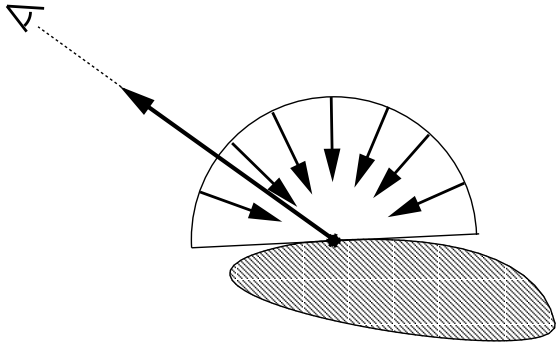

Figure 1.3: Local radiance reflection.

thus, ρ is fully determined by the BRDF. It can also be defined by using differentiation instead of integration:

$$\rho(x, w_i) \;=\; \frac{dM'_r(x, w_i)}{dE(x, w_i)} \;=\; \frac{dM'_r(x, w_i)}{L_i(x, w_i)\, d\sigma_x(w_i)} \tag{1.14}$$

where $M_r(x, O')$ is a measure of power leaving from point x due reflection of power arriving at x from directions in $O' \subseteq O$. where $dM_r(x, w_i)$ is the differential amount of radiant exitance at x due to irradiance from w_i. The integral version of the above differential definition is as follows:

$$\begin{aligned} M_r(x) &= \int_O dM'_r(x, w_i) \\ &= \int_O \rho(x, w_i)\, dE(x, w_i) \\ &= \int_O \rho(x, w_i)\, L_i(x, w_i)\, d\sigma_x(w_i) \end{aligned}$$

1.2.2 Properties of the BRDF.

Up to now, no restriction has been imposed onto the BRDF. But there are some laws in physics which lead to properties that a BRDF function must obey. These are: conservation of energy and symmetry.

Conservation of energy

The total amount of energy reflected on a point, per unit area, cannot be greater than the amount of energy landing on that point, per unit area. This implies that radiant exitance on x is bounded by irradiance at x, that is, for all $x \in S$, it holds that

$$M_r(x) \;\leq E(x) \tag{1.15}$$

the above condition must hold for all possible distributions of incoming radiance. This includes the case when energy is coming from just a direction. This case can be modeled by using a Dirac delta for incoming radiance: $L_i(x, w_i) = \delta(w_i - w_s)$. This equation implies that all the energy on x comes from a single punctual source whose radiance at x is 1 and is seen from x in direction w_s. We can now rewrite the last inequality by substituting this concrete example of L_i. The presence of a delta function simplifies the integrals, then L_s is canceled, and we get:

$$\int_O f_r(x, w_o, w_i)\, d\sigma_x(w_o) \;=\; \rho(x, w_i) \;\leq\; 1 \tag{1.16}$$

this condition is independent of L_s and of the distribution of L_i. Thus, we see that conservation of energy implies that the ρ function is bounded by one.

It can be proved, by integration, that (1.16) implies (1.15), as can be seen in [Lewis93]. We can integrate both sides of (1.16) for all incoming directions, and also multiplying by an arbitrary incoming radiance function L_i. We obtain

$$\int_O L_i(x, w_i)\, d\sigma_x(w_i) \;\geq\; \int_O \rho(x, w_i)\, L_i(x, w_i)\, d\sigma_x(w_i) \tag{1.17}$$

substituting ρ by its definition, we directly derive (1.15). As the implications in both directions have been shown, both (1.15) and (1.16) are equivalent, although (1.16) is simpler to check. The ρ function, thus, must always be bounded by 1.

Reciprocity or Symmetry

The *Helmholtz Reciprocity Rule* states that the chance a photon has for traversing any path through various scattering events is is the same for the reverse traversing order. As surface reflection is due to scattering at the atoms of the surface, we get the following restriction of the BRDF:

$$f_r(x, w_o, w_i) \;=\; f_r(x, w_i, w_o) \tag{1.18}$$

1.2.3 BRDF types.

Traditionally, in the Computer Graphics literature several well known types of local reflection have been used. In this section we show the BRDFs which model these type of scattering and try to guess if the obey the above restrictions. These are simple and widely known models for BRDF. Other more complicated or realistic models have been described, which are not included here. The reader interested in these models can refer to articles such as [Lewis93, Schlick94].

Perfect specular reflection.

Perfect specular reflection is the reflection behavior shown by perfectly polished mirrors. At a point x, the reflected radiance for some outgoing direction w_o is a fraction of the incoming radiance from the reflected direction of w_o with respect to

the normal at x (and independent of the radiance coming from *any* other direction). We have that

$$L_r(x, w_o) \;=\; \rho_s(x)\, L_i(x, \mathit{refl}_x(w_o)) \tag{1.19}$$

where $\mathit{refl}_x(w_o) = 2\cos(n_x, w_o)n_x - w_o$, and ρ_s is a function which controls the fraction of energy which is reflected this way, obeying $0 \leq \rho_s(x) \leq 1$ for all points x.

This implies that just one incoming direction contributes to the outgoing radiance. The BRDF must be zero for all directions except for that one, but its integral is non-zero. Such a property is hold by the Dirac's delta function. We can define the specular BRDF (noted as f_{rs}) as follows [Cohen93]:

$$f_{rs}(x, w_o, w_i) \;=\; \rho_s(x)\, \frac{\delta(w_i - \mathit{refl}_x(w_o))}{cos(n_x, w_i)} \tag{1.20}$$

we can see that (1.20) implies (1.19) by using equation (1.12), as follows:

$$\begin{aligned} L_r(x, w_o) &= \int_O \rho_s(x)\, \frac{\delta(w_i - \mathit{refl}_x(w_o))}{cos(n_x, w_i)}\, L_i(x, w_i)\, d\sigma_x(w_i) \\ &= \rho_s(x) \int_O \delta(w_i - \mathit{refl}_x(w_o))\, L_i(x, w_i)\, d\sigma(w_i) \\ &= \rho_s(x)\, L_i(x, \mathit{refl}_x(w_o)) \end{aligned}$$

here we have used the properties of the delta function, and also we have made the substitution: $d\sigma_x(w) = cos(n_x, w)\, d\sigma(x)$

The symmetry of this BRDF is easy to prove, because of the reciprocity of the reflection function: if w is the reflected vector of w', then w' is also the reflected vector of w, and also it holds that $cos(n_x, w) = cos(n_x, w')$.

Conservation of energy can be proved by computing the directional-hemispherical reflectivity:

$$\begin{aligned} \rho(x, w_i) &= \int_O f_{rs}(x, w_o, w_i)\, d\sigma_x(w_o) \int_O \rho_s(x)\, \frac{\delta(w_i - \mathit{refl}_x(w_o))}{cos(n_x, w_i)}\, d\sigma_x(w_o) \\ &= \rho_s(x) \int_O \delta(w_i - \mathit{refl}_x(w_o))\, d\sigma(w_o) \\ &= \rho_s(x) \\ &\leq 1 \end{aligned}$$

which is true because of the definition of ρ_s. Here we have used again the properties of the δ function. From the above we can derive that, when this kind of reflection takes place, it holds that $\rho(x, w_i) = \rho_s(x)$, and we see that f_{rs} conserves energy, because ρ_s is bounded by 1.

Diffuse reflection.

The diffuse BRDF is independent of the incoming and outgoing directions. That is, the value $f_{rd}(x, w_o, w_i)$ only depends of x. This can be written as $f_{rd}(x, w_o, w_i) =$

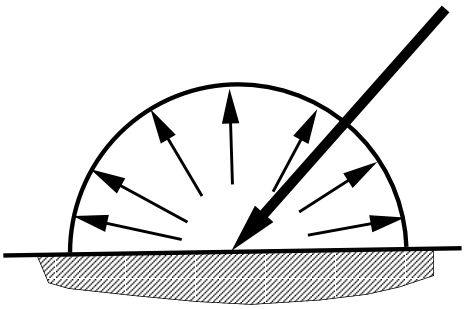

Figure 1.4: The shape of the diffuse BRDF.

$f_{rd}(x)$. In figure 1.4 we can see the shape of this BRDF. Note that, independently of outgoing direction, reflected radiance remains constant. With this definition of f_r, we can use (1.9) and (1.12) to rewrite the radiant exitance M_r as:

$$\begin{aligned}
M_r(x) &= \int_O L_r(x, w_o)\, d\sigma_x(w_o) \\
&= \int_O \left[\int_O f_r(x)\, L_i(x, w_i)\, d\sigma_x(w_i) \right] d\sigma_x(w_o) \\
&= f_{rd}(x) \int_O \left[\int_O L_i(x, w_i)\, d\sigma_x(w_i) \right] d\sigma_x(w_o) \\
&= f_{rd}(x) \int_O E(x)\, d\sigma_x(w_o) \\
&= f_{rd}(x)\, E(x) \int_O d\sigma_x(w_o) \\
&= f_{rd}(x)\, E(x)\, \sigma_x(O) \\
&= f_{rd}(x)\, E(x)\, \pi
\end{aligned}$$

from the above we see that $M_r(x)/E(x) = f_{rd}(x)\pi$. This value is known as the *reflectance* at x. It is the ratio of reflected power to incoming power after diffuse reflection at x, and is noted as $\rho_d(x)$. From all of the above, we can state the definition of $f_{rd}(x)$:

$$f_{rd}(x, w_o, w_i) \;=\; f_{rd}(x) \;=\; \frac{\rho_d(x)}{\pi} \tag{1.21}$$

it is obvious that this BRDF is symmetrical. Conservation of energy is proved by expanding the ρ function. It is easy to prove that

$$\rho(x, w_i) \;=\; \rho_d(x) \tag{1.22}$$

for all incoming directions w_i

Glossy reflection.

Both previous BRDFs are very simple. Although some real materials show that behavior, there are also too many surfaces that reflect energy in other ways. In the Computer Graphics literature it is possible to find many papers devoted to the task of modeling and implementing other types of BRDFs. Probably, the simplest and older model for glossy surfaces is the one proposed by Phong [Phong75]. This was a intermediate solution between efficiency and physical accuracy. The model was named a *shading formula*, that is, an algorithm to compute the intensity of reflected light for an outgoing direction, given a set of punctual light sources. After this, in [Lewis93], that shading formula was interpreted as the model for a BRDF, and it was proven that this BRDF does not obeys conservation of energy. In that paper, some modified BRDF models where proposed obeying both conservation of energy and reciprocity. Finally, in [Lafortune94], it was analyzed the use of those models in the context of Monte-Carlo algorithms, and another BRDF was proposed, also based on the Phong shading formula. This model has been called the *modified Phong* model. Using the notation introduced in this text, it can be expressed as

$$f_{rp}(x, w_o, w_i) \;=\; \rho_p(x)\,\frac{n+2}{2\pi}\,cos^n(w_o, \mathit{refl}_x(w_i)) \tag{1.23}$$

where $n \geq 1$ is a real value called the *Phong exponent*, with higher values producing surfaces with a narrower brightness peak centered at the reflection direction. Figure 1.5 shows an approximate polar diagram of the BRDF for $n = 1$. For greater

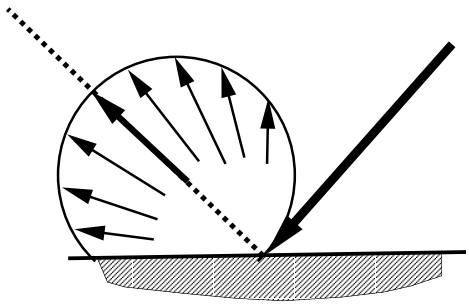

Figure 1.5: The shape of glossy BRDFs for $n = 1$

values of n, the BRDFs gets closer to the shape shown in figure 1.6, which exhibits a narrower cosine lobe. The function ρ_p gives, for each point in the surfaces, a value which is an upper bound of the ratio of reflected. This is easy to show, because the integral of cosine term, for the hemisphere, is smaller than the integral for the whole sphere, and this is $2/\pi(n+2)$. Thus, conservation of energy can be ensured if $\rho_p(x) \leq 1$. In the case that $w_i = n_x$, then the directional-hemispherical reflectivity reaches its maximum value, which is equal to $\rho_p(x)$. It is difficult to give an exact analytical expression for it because the integration region boundary is complicated [Lafortune94]. Reciprocity holds obviously, because $cos(w_o, \mathit{refl}_x(w_i)) = cos(\mathit{refl}_x(w_o), w_i)$.

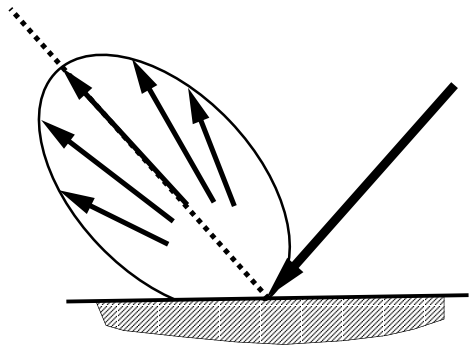

Figure 1.6: The (approximate) shape of glossy BRDFs for $n > 1$

It has been proposed also a model whose cosine term is defined by using the *halfway vector*, called h, which is defined as the halfway between w_i and w_o (that is $h = (w_i + w_o)/|w_i + w_o|$). Now the cosine term becomes $cos^n(h, n_x)$.

Mixed reflection behavior.

Usually, surfaces do not show a perfect specular, diffuse or glossy behavior. Some surfaces can reflect energy simultaneously in more than one of these ways. This is the case, for instance, of materials with a transparent layer above an opaque one. Some of the energy can be reflected in the transparent layer, causing specular or glossy reflections, and some in the underlying opaque surface (after penetrating transparent layers), causing diffuse reflection.

If we assume that, for all points x in an environment, it is true that

$$\rho_s(x) + \rho_d(x) + \rho_p(x) \quad \leq \quad 1 \tag{1.24}$$

then the mixed BRDF defined as $f_r = f_{rs} + f_{rd} + f_{rp}$ obeys conservation of energy, as can be easily deduced form previous formulations.

Finally, the relative weights of each kind of BRDF can depend also of the incoming direction w_i, not just on the surface point x. A typical example of this are some polished stone floors. When viewed from above ($cos(w_o, n_x)$ near 1), they are perceived as diffuse surfaces, but when the viewing direction is nearly perpendicular to the normal ($cos(w_o, n_x)$ near 0), they look like perfect specular or glossy surfaces [Lafortune97]

1.3 Global light transport.

The previous section was named as *local light transport* because equation (1.12) models how incoming light is reflected locally on a point x. But our goal is to state the equation governing *global* light transport, that is, light transport taking into account all the points and directions on an environment. This equation is equivalent

to that introduced by Kajiya, although in a slightly modified form. Kajiya equation is ultimately rooted on the Boltzmann transport equation [Kajiya86, Arvo90].

1.3.1 Integral equations for radiance.

The first step towards the full integral equation is to note that incoming radiance on a point is due to outgoing radiance from other point on the scene. As radiance is conserved along straight lines, we have that the radiance incoming at point x from direction w_i is equal to the radiance outgoing from another point x' at direction $-w_i$. We can assume that both x and x' are points on the surfaces. For this to be true, it is necessary that no other object blocks the light between x and x', that is, x' is the first visible point from x in the direction w_i. This can be stated formally by using the function p, which gives, for any point x and direction w, the first visible point in S [Arvo95a]. Now, we state the previous rule as $L_i(x, w_i) = L(p(x, w_i), -w_i)$. By using this equality we can rewrite equation (1.12), as follows:

$$L_r(x, w_o) \;=\; \int_O f_r(x, w_o, w_i)\, L(p(x, w_i), -w_i)\, d\sigma_x(w_i) \tag{1.25}$$

the previous equation can be written in operator form as $L_r = \mathcal{T}L$, where $\mathcal{T}$ is an integral operator. Note that $L = L_e + L_r$. Expanding L_r by using the previous equality, we obtain:

$$L \;=\; L_e \;+\; \mathcal{T}L \tag{1.26}$$

this is an integral equation. For any given f_r and L_e functions, there is only just one radiance function L which obeys the above, proven that f_r conserves the energy.

In order to introduce existing computational methods which solve for equation (1.25), it is convenient to write it as a *Fredholm Integral Equation of the Second Kind*, because such equations have been studied previously in order to find solution methods for them. These equations are equations with the form:

$$f(x) \;=\; g(x) \;+\; \int_D k(x, y)\, f(y)\, dm(y) \tag{1.27}$$

where D is a domain, m a measure, and g is an integrable function under the measure m in domain D. If k obeys some properties, then there exists a unique f obeying the equation. We can see in equation (1.27) that the unknown function f is defined in the same domain of integration D, while in (1.25), L is defined on $S \times O$ but integration is done on O. Another difference comes from the occurrence of the p function in the arguments of L in (1.25), while in (1.27) the argument of L is just y (no function is applied to it).

However, we can write an equation which is equivalent to (1.25), but has the same form of (1.27). This can be done in two step: first, we remove the p function from 1.25, by changing the variable of integration. Then, by using an additional change, we transform the previous integration in an integration in D.

With respect to the first step, instead of integration for directions w_i over x, will make integration for points y in the surfaces.

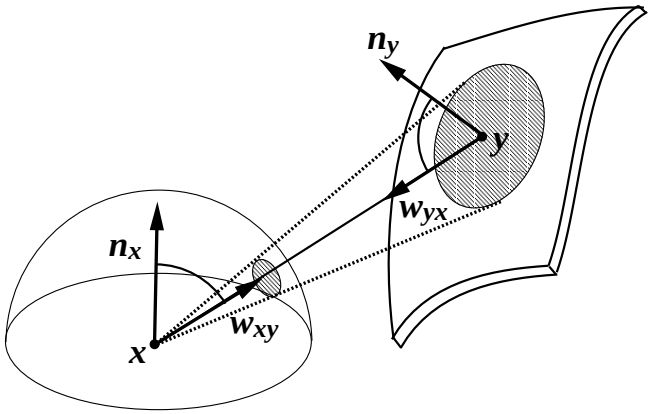

Figure 1.7: The relation between the measures σ_x and A.

In figure 1.7 we see two points x and y in the objects surfaces, with $y = p(x, w_i)$. For any two arbitrary points in the surfaces, we define w_{ab} as the unit length vector from a to b, that is, $w_{ab} = (b-a)/|b-a|$. Then, we have $w_{xy} = w_i$ y $w_{yx} = -w_{xy}$. For any set O' of unit length vectors, the value $\sigma(O')$ is the small shaded region over the hemisphere in the figure. This set of direction covers a region (the bigger shaded region in the figure) S' surrounding the point y, whose area is $A(S')$. Taking limits when $\sigma(O')$ goes to zero, we obtain the following relation between differential elements:

$$d\sigma_x(w_i) \;=\; H(x,y)\, dA(y) \tag{1.28}$$

where $y = p(x, w_i)$, and H is a purely geometric term defined as

$$H(x,y) \;=\; \frac{cos(n_x, w_{xy})\, cos(n_y, w_{yx})}{|x-y|^2} \tag{1.29}$$

As radiance is conserved along lines, we also have, that

$$L_i(x, w_{xy}) \;=\; L(y, w_{yx}) \tag{1.30}$$

then we can change the variable of integration in (1.12) and we obtain:

$$L_r(x, w_o) \;=\; \int_{S_x} f_r(x, w_o, w_{xy})\, H(x,y)\, L(y, w_{xy})\, dA(y) \tag{1.31}$$

where $S_x \subseteq S$ is the set of points in S which are visible from x. The integral can be extended to the whole S, by using another geometric term $G(x,y)$ defined by

$$G(x,y) \;=\; H(x,y)\, V_{xy} \tag{1.32}$$

where $V(x,y)$ is 1 when x and y are mutually visible, and 0 otherwise. By using G, we get:

$$L_r(x, w_o) \;=\; \int_{S} f_r(x, w_o, w_{xy})\, G(x,y)\, L(y, w_{xy})\, dA(y) \tag{1.33}$$

In figure 1.8 we see now the reflected radiance expressed as an integral over all points S of the outgoing radiance, instead of an integral over all directions of incoming radiance.

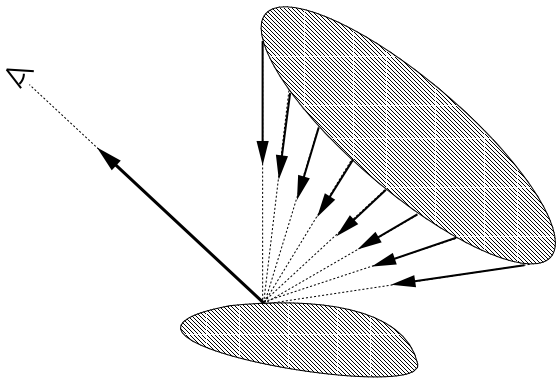

Figure 1.8: Reflected radiance as an integral of outgoing radiance.

The previous integration is done over S, instead of O. We still have to transform the previous integration on the domain $S \times O$. This second step is done by using a delta function. By using the properties of the δ function, we get

$$L(y, w_{yx}) \;=\; \int_O \delta(w - w_{yx})\, L(y, w)\, d\sigma(w) \tag{1.34}$$

where δ is the Dirac delta function with respect to solid angle measure σ. Then, we can substitute $L(y, w_{yx})$ by the previous expression in (1.33), obtaining:

$$\begin{aligned} & L_r(x, w_o) \\ = \;& \int_S f_r(x, w_o, w_{xy})\, G(x, y) \left[\int_O \delta(w - w_{yx})\, L(y, w)\, d\sigma(w) \right] dA(y) \\ = \;& \int_{S \times O} f_r(x, w_o, w_{xy})\, \frac{cos(n_x, w_{xy})}{|x - y|^2}\, V_{xy}\; \delta(w - w_{yx}) L(y, w)\, d\mu(y, w) \end{aligned}$$

to simplify the notation, we isolate all the dependencies of both x and y in one kernel function defined as:

$$K(x, w_o, y, w) \;=\; f_r(x, w_o, w_{xy})\, \frac{cos(n_x, w_{xy})}{|x - y|^2}\, V_{xy}\, \delta(w - w_{yx}) \tag{1.35}$$

We will use symbols to denote elements in $D = S \times O$, instead of pairs in $S \times O$. Thus, we rewrite $\mathbf{r} = (x, w_o)$, $\mathbf{s} = (y, w)$, and $K(\mathbf{r}, \mathbf{s}) = K(x, w_o, y, w)$. All this leads to the following equation:

$$L_r(\mathbf{r}) \;=\; \int_D K(\mathbf{r}, \mathbf{s})\, L(\mathbf{s})\, d\mu(\mathbf{s}) \tag{1.36}$$

In figure 1.9, we see how the integration is done for *all* the rays in D. Only those rays pointing to x do contribute to the integral, and this is ensured by the presence of the δ function.

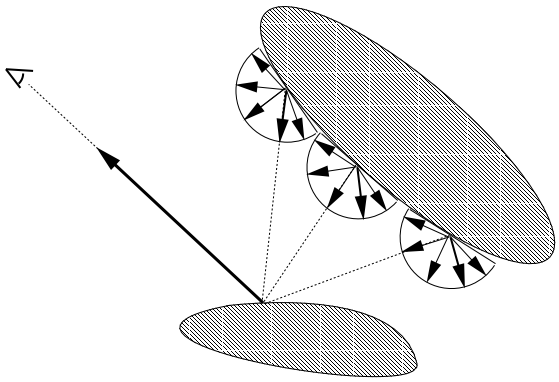

Figure 1.9: Radiance as an integral in ray space.

The previous integral equation (1.36) is equivalent to previous formulations, that is, to equations (1.25), (1.26) and (1.33). Finally, equation (1.36) has the same form that the generic equation (1.27), where L plays the role of f, L_e the one of g, and K the one of k. Note that in equation (1.26) we introduced the transport operator $\mathcal{T}$. From (1.36), we can define the action of that operator over the L function as follows:

$$L_r(\mathbf{r}) \;=\; (\mathcal{T}L)(\mathbf{r}) \;=\; \int_D K(\mathbf{r},\mathbf{s})\, L(\mathbf{s})\, d\mu(\mathbf{s}) \tag{1.37}$$

The K function is called the *kernel* of the transport operator $\mathcal{T}$. Equation (1.36) can also be expressed in a differential form

$$K(\mathbf{r},\mathbf{s}) \;=\; \frac{dL_r(\mathbf{r})}{L(\mathbf{s})\, d\mu(s)} \tag{1.38}$$

and we can state that $K(\mathbf{r},\mathbf{s})$ is the differential radiance of $\mathbf{r}$ due to a unit of energy at $\mathbf{s}$, (differential amount of energy leaving $dA(y)$ trough the projected solid angle $d\sigma_y(w)$).

1.3.2 The radiosity equation.

In the case that an environment has surfaces with just diffuse reflection, as defined in (1.21), the integral equation which governs radiance transport simplifies, yielding the classic radiosity equation. Let us assume that $L_e(x,w)$ is independent of w (that is, emitted radiance is the same for all directions at any given point). It can

be shown that $L_r(x,w)$ is also independent of w, by using both (1.21) and (1.12):

$$L_r(x,w) = \int_O \frac{\rho_d(x)}{\pi} L_i(x,w_i)\, d\sigma_x(w_i) \quad (1.39)$$

$$= \rho_d(x)\, \frac{E(x)}{\pi} \quad (1.40)$$

thus we can remove the w from the radiance functions, and write $L_e(x)$,$L_r(x)$ and $L(x)$. Usually, these functions are called *radiosity* functions, and written using a B. Then, we have emitted radiosity, $B_e(x) = L_e(x)$, reflected radiosity, $B_r(x) = L_r(x)$, and (total outgoing) radiosity, $B(x) = L(x)$. We can rewrite equation (1.33) by using the previous definitions:

$$B_r(x) = \frac{\rho_d(x)}{\pi} \int_S G(x,y)\, B(y)\, dA(y) \quad (1.41)$$

As we know that $B(x) = B_e(x) + B_r(x)$, then we can expand B_r and obtain:

$$B(x) = B_e(x) + \rho_d(x) \int_S \frac{cos(n_x, w_{xy})\, cos(n_y, w_{yx})}{\pi\, |x-y|^2}\, B(y)\, dA(y) \quad (1.42)$$

This is the classic *radiosity equation*, which can be viewed as a simplification of (1.33).

1.3.3 Integral operators and solutions to integral equations.

Lets consider any integral equation with the form of (1.27). It can be rewritten by using operator form. For instance, let us define the operator $\mathcal{A}$ as the integral operator induced by the k function:

$$(\mathcal{A}f)(x) = \int_D k(x,y)\, f(y)\, dy \quad (1.43)$$

then, equation (1.27) can be simply rewritten as:

$$f = g + \mathcal{A}f \quad (1.44)$$

If we know both g and k (and thus $\mathcal{A}$), we can wonder about the existence and uniqueness of a function f obeying the previous equation. In order to answer this, we briefly review here function and operators norms, as described in [Arvo95a], and then we use those results to find the solution to that equation.

Operators and norms.

The L_p-norm of a function f, under a measure m, is noted as $|f|_p$ and can be defined as:

$$|f|_p = \left[\int f^p(x)\, dm(x) \right]^{1/p} \quad (1.45)$$

function norms induce operator norms, defined for all operators acting over those functions. For an operator $\mathcal{A}$, its L_p norm is defined as:

$$|\mathcal{A}|_p \;=\; \inf\{a \,:\, |\mathcal{A}f|_p \leq a|f|_p \text{for all } f\} \tag{1.46}$$

note that, from the above definition, we can derive the following property, which holds $\forall f$:

$$|\mathcal{A}f|_p \;\leq\; |\mathcal{A}|_p\,|f|_p \tag{1.47}$$

this is an interesting property which will be used later.

Note that operators can be added and composed. For any two operators $\mathcal{A}$ and $\mathcal{B}$, and a function f, we have $(\mathcal{A}+\mathcal{B})f = \mathcal{A}f + \mathcal{B}f$, and $(\mathcal{A}\mathcal{B})f = \mathcal{A}(\mathcal{B}f)$. It is easy to show that:

$$|\mathcal{A}\mathcal{B}|_p \;\leq\; |\mathcal{A}|_p\,|\mathcal{B}|_p \tag{1.48}$$

$$|\mathcal{A}+\mathcal{B}|_p \;\leq\; |\mathcal{A}|_p \,+\, |\mathcal{B}|_p \tag{1.49}$$

A *linear* operator is one which obeys $\mathcal{A}(f+g) = \mathcal{A}f + \mathcal{B}g$ for all f and g. If $\mathcal{A}$ is a linear operator, the distributive property holds, that is, $\mathcal{A}(\mathcal{B}+\mathcal{C}) = \mathcal{A}\mathcal{B} + \mathcal{A}\mathcal{C}$, for all operators $\mathcal{B}$ and $\mathcal{C}$. A special operator is the identity, noted as $\mathcal{I}$. This is the one mapping every function to itself, that is, for all functions f, we have $\mathcal{I}f = f$. The norm of the identity operator is always 1. We also define the operator $\mathcal{A}^n$, for any base operator $\mathcal{A}$ and natural number n. This is done iteratively: $\mathcal{A}^0 = \mathcal{I}$, and $\mathcal{A}^{n+1} = \mathcal{A}\mathcal{A}^n$. From (1.48) we derive:

$$|\mathcal{A}^n|_p \;\leq\; |\mathcal{A}|_p^n \tag{1.50}$$

Finally, we define the operator $\mathcal{A}^{(n)}$ as $\sum_{i=0}^{n}\mathcal{A}^i$. Note that, when A is linear, the following property holds:

$$\mathcal{A}^{(n+1)} \;=\; \mathcal{I} \,+\, \mathcal{A}\mathcal{A}^{(n)} \tag{1.51}$$

Existence of the solution

The existence and uniqueness of a function f obeying equation (1.44), and such that $\int_D f^p(x)\,dm(x)$ exists, is guaranteed when $\mathcal{A}$ is linear and $|\mathcal{A}|_p < 1$. This can be shown be considering the series of operators $\mathcal{A}^{(n)}$. From previous results we see that

$$|\mathcal{A}^{(n)}|_p \;\leq\; \sum_{i=0}^{n} |\mathcal{A}|_p^i \tag{1.52}$$

when $|\mathcal{A}|_p < 1$, the series of norms $|\mathcal{A}^{(n)}|_p$ converges, and

$$\lim_{n\to\infty} |\mathcal{A}^{(n)}|_p \;\leq\; \frac{1}{1-|\mathcal{A}|_p} \tag{1.53}$$

thus, for any function h, the series of functions $\mathcal{A}^{(n)}h$ converge to a function h' such that $|h'|_p \leq 1/(1-|\mathcal{A}|_p)$. We can define the operator $\mathcal{A}^+$ as the one which maps h onto h'. Function h' is the limit of a function series

$$h' \;=\; \mathcal{A}^+h \;=\; \lim_{n\to\infty} \mathcal{A}^{(n)}\,h \tag{1.54}$$

If we apply both sides of (1.51) to any function and then take limits when $n \to \infty$, we can derive an interesting property of $\mathcal{A}^+$

$$\mathcal{A}^+ \;=\; \mathcal{I} \,+\, \mathcal{A}\,\mathcal{A}^+ \tag{1.55}$$

now, applying both sides of this equality to the function g in (1.44), we get

$$(\mathcal{A}^+ g) \;=\; g \,+\, \mathcal{A}\,(\mathcal{A}^+ g) \tag{1.56}$$

and we deduce that $\mathcal{A}^+ g$ obeys equation (1.44). From this we know that its solution exists, and can be expressed as $f = \mathcal{A}^+ g$. The norm of $\mathcal{A}^+$ obeys

$$|\mathcal{A}^+|_p \;\leq\; \frac{1}{1 - |\mathcal{A}|_p} \tag{1.57}$$

Operator $\mathcal{A}^+$ is called the *resolvent operator* of $\mathcal{A}$, as can be seen in [Arvo95a].

The norm of linear integral operators.

Let us assume that the $\mathcal{A}$ operator is defined as in (1.43). We can look for the norm of this kind of operators. First, we define

$$m_k \;=\; \operatorname*{ess\,sup}_{y \in D} \left\{ \int_D k(x,y)\, dm(x) \right\} \tag{1.58}$$

where *ess sup* stands for *essential supremum*[2] . We include a k as a subscript of m_k because this value depends on the k function. Now, lets expand the value $|\mathcal{A}h|_1$ for any function h

$$\begin{aligned}
|\mathcal{A}h|_1 &= \int_D (\mathcal{A}h)(x)\, dm(x) && (1.59)\\
&= \int_D \left[\int_D k(x,y)\, h(y)\, dm(y) \right] dm(x) && (1.60)\\
&= \int_D \left[\int_D k(x,y)\, dm(x) \right] h(y)\, dm(y) && (1.61)\\
&\leq \int_D m_k\, h(y)\, dm(y) && (1.62)\\
&= m_k\, |h|_1 && (1.63)
\end{aligned}$$

the previous inequality is true for any h, then we can obtain a bound $|A|_1$. By using (1.46) we get

$$|A|_1 \leq m_k \tag{1.64}$$

[2]The notion of essential supremum is briefly described in [Arvo95a]. It is the supremum of the set (that is, the infimum of all its upper bounds), except for a null measure set with respect to measure m.

1.3.4 The solution to radiance transport equation.

Once we know the conditions which ensure the solution to *any* integral equation, we can question about the solution to the radiance transport equation, in the form given in (1.26). From previous results, we know that the function L exists if $\mathcal{T}$ is linear and $|\mathcal{T}|_1 < 1$. Linearity holds for all integral operators, as can be easily deduced from (1.37). With respect to the norm, we can use (1.64). If $m_K < 1$ then $|\mathcal{T}|_1 < 1$. Thus, we have to check whether $m_K < 1$, where K is as defined in (1.35). To do this, we expand $\int_D K(\mathbf{r},\mathbf{s})\, d\mu(\mathbf{r})$, where $\mathbf{r} = (x, w_o)$ and $\mathbf{s} = (y, w)$ (note that y and w are fixed values)

$$\begin{aligned}
& \int_D K(\mathbf{r},\mathbf{s})\, d\mu(\mathbf{r}) \\
= & \int_{S\times O} f_r(x, w_o, w_{xy})\, \frac{cos(n_x, w_{xy})}{|x-y|^2}\, V_{xy}\, \delta(w - w_{yx})\, dA(x)\, d\sigma_x(w_o) \\
= & \int_S \left[\int_O f_r(x, w_o, w_{xy})\, d\sigma_x(w_o) \right] \delta(w - w_{yx})\, V_{xy}\, \frac{cos(n_x, w_{xy})}{|x-y|^2}\, dA(x) \\
= & \int_O \left[\int_O f_r(x, w_o, -w')\, d\sigma_x(w_o) \right] \delta(w - w')\, d\sigma(w') \\
= & \int_O f_r(x, w_o, -w)\, d\sigma_x(w_o) \\
= & \rho(x, -w)
\end{aligned}$$

here, we have transformed the integration in S in an integration in 0, by using the equality (1.28). From this last result we deduce an expression of m_K

$$m_K = \underset{(x,w)\in D}{\textit{ess sup}} \{\, \rho(x, w) \,\} \tag{1.65}$$

from here we see that, if the underlying BRDF obeys conservation of energy then $m_K < 1$ and equation (1.25) has a solution.

1.4 Adjoint transport equations.

The solution function f of equation (1.27) cannot be expressed analytically for almost all of the cases. Usually, the target is to know the inner product of f and any other function h, written as $\langle f \mid h \rangle$. In this section we introduce the adjoint transport operator and show how that inner product can be obtained from the solution of two dual integral equations.

1.4.1 Inner products and adjoint operators.

The inner product of two functions f and h, is a real value[3] defined as:

$$\langle f \mid h \rangle = \int_D f(x)\, h(x)\, dm(x) \tag{1.66}$$

[3] We will assume that f and h are two arbitrary integrable functions in D (under measure m)

as can be seen, inner product is relative to a measure, used in the integration. This can be stated by using the name of the measure function as a subscript, when any confusion may arise. From its definition, it is easy to show the following properties of the inner product, which holds for all functions f,h and h', and for all real values a

$$\begin{aligned} \langle f \mid h \rangle &= \langle h \mid f \rangle & (1.67) \\ \langle f \mid h + h' \rangle &= \langle f \mid h \rangle + \langle f \mid h' \rangle & (1.68) \\ \langle af \mid h \rangle &= a \langle f \mid h \rangle & (1.69) \end{aligned}$$

The adjoint operator of an arbitrary operator $\mathcal{A}$ is written as $\mathcal{A}^*$, and is an operator which obeys

$$\langle \mathcal{A}f \mid h \rangle = \langle f \mid \mathcal{A}^* h \rangle \quad (1.70)$$

for all functions f and h. Note that not all of the operators have an adjoint. In the case that $\mathcal{A}$ is an integral operator with a kernel k, as defined in (1.43), then there exists $\mathcal{A}^*$, which can be defined as

$$(\mathcal{A}^* h)(x) = \int_D k(x,y)\, h(x)\, dm(x) \quad (1.71)$$

it can be shown that $\mathcal{A}^*$ obeys (1.70), by expanding $\langle \mathcal{A}f \mid h \rangle$

$$\begin{aligned} \langle \mathcal{A}f \mid h \rangle &= \int_D (\mathcal{A}f)(x)\, h(x)\, dm(x) \\ &= \int_D \left[\int_D k(x,y)\, f(y)\, dm(y) \right] h(x)\, dm(x) \\ &= \int_{D\times D} k(x,y)\, f(y)\, h(x)\, dm(y) dm(x) \\ &= \int_D \left[\int_D k(x,y)\, h(x)\, dm(x) \right] f(y)\, dm(y) \\ &= \int_D (\mathcal{A}^* h)(y)\, f(y)\, dm(y) \\ &= \langle f \mid \mathcal{A}^* h \rangle \end{aligned}$$

Lets assume that $\mathcal{A}$ and $\mathcal{B}$ are two arbitrary operators with adjoint, and f and h two arbitrary functions. It is easy to prove that $\langle (\mathcal{A}+\mathcal{B})f \mid h \rangle = \langle f \mid (\mathcal{A}^* + \mathcal{B}^*)h \rangle$ and $\langle \mathcal{A}\mathcal{B}f \mid h \rangle = \langle f \mid \mathcal{B}^*\mathcal{A}^* h \rangle$. Then we obtain the following equalities

$$\begin{aligned} (\mathcal{A}+\mathcal{B})^* &= \mathcal{A}^* + \mathcal{B}^* & (1.72) \\ (\mathcal{A}\mathcal{B})^* &= \mathcal{B}^*\mathcal{A}^* & (1.73) \\ (\mathcal{A}^n)^* &= (\mathcal{A}^*)^n & (1.74) \\ (\mathcal{A}^{(n)})^* &= (\mathcal{A}^*)^{(n)} & (1.75) \end{aligned}$$

1.4.2 The dual integral equation.

As stated, lets assume that we are interested in the value $\langle f \mid h \rangle$, where f is the solution to (1.44) and h is an arbitrary function. We know that the linear integral operator $\mathcal{A}$, as defined in (1.43), has an adjoint $\mathcal{A}^*$. This adjoint is also a linear integral operator, as shown in the previous section. Lets assume that $|\mathcal{A}^*|_1 < 1$. Thus, exists the resolvent operator of $\mathcal{A}^*$, which is written as $(\mathcal{A}^*)^+$. In these conditions the following integral equation

$$l \;=\; h + \mathcal{A}^* \, l \tag{1.76}$$

has a solution, that is, exists an integrable function l obeying it, and can be obtained as $l = (\mathcal{A}^*)^+ h$. Note that g and h in equations (1.44) and (1.76) are arbitrary integrable functions. From the property (1.75), we can obtain the following equalities

$$\begin{aligned}
\left\langle \mathcal{A}^+ g \mid h \right\rangle &= \left\langle \lim_{n\to\infty} \mathcal{A}^{(n)} g \mid h \right\rangle \\
&= \lim_{n\to\infty} \left\langle \mathcal{A}^{(n)} g \mid h \right\rangle \\
&= \lim_{n\to\infty} \left\langle g \mid (\mathcal{A}^{(n)})^* h \right\rangle \\
&= \lim_{n\to\infty} \left\langle g \mid (\mathcal{A}^*)^{(n)} h \right\rangle \\
&= \left\langle g \mid \lim_{n\to\infty} (\mathcal{A}^*)^{(n)} h \right\rangle \\
&= \left\langle g \mid (\mathcal{A}^*)^+ h \right\rangle
\end{aligned}$$

so, we can state that $(\mathcal{A}^*)^+$ is the adjoint of $\mathcal{A}^+$, that is

$$(\mathcal{A}^+)^* \;=\; (\mathcal{A}^*)^+ \tag{1.77}$$

and we get

$$\langle f \mid h \rangle \;=\; \langle l \mid g \rangle \tag{1.78}$$

then, computation of $\langle f \mid h \rangle$ can be viewed also as the computation of $\langle l \mid g \rangle$. This has a number of applications in Global Illumination.

1.4.3 Importance transport equations.

Once stated the notion of inner product and adjoint operators, we can apply them to the radiance function and the integral equation defining it. Lets assume that W_e is any function used to measure the radiance function, that is, we want to compute the value $\langle L \mid W_e \rangle$. This function has been called *emitted importance* function, or *emitted potential* function.

For example, we can define $W_e(\mathbf{r})$ as non-zero only for rays going through a pixel in the viewplane and pointing towards the viewpoint, and zero for any other. Then, the value $\langle L \mid W_e \rangle$ is the average radiance of a pixel. Other examples can be given.

The inner product $\langle L \mid W_e \rangle$ can be seen as a weighted sum of the values of the radiance function at the different rays in the domain. The value $W_e(\mathbf{r})$, for any ray $\mathbf{r}$, can be taken as the fraction of $L(\mathbf{r})$ which contributes to that weighted sum. In the example given above, W_e is the function which gives, for each ray, the fraction of its radiance which contributes to the pixel intensity.

We know that when the BRDF conserves energy, then exists $\mathcal{T}^+$ and the radiance function L. It is easy to show that, in these conditions, and assuming also that the BRDF is symmetric, then $|\mathcal{T}^*|_1 < 1$. This implies that the following equation

$$W \;=\; W_e \,+\, \mathcal{T}^* W \tag{1.79}$$

has a solution, that is, exists the operator $(\mathcal{T}^*)^+$ which is the adjoint of $\mathcal{T}^+$, and also exists the integrable function W, which can be obtained as $W = (\mathcal{T}^*)^+ W_e$. Then the value $\langle L \mid W_e \rangle$ can be rewritten as

$$\langle L \mid W_e \rangle \;=\; \left\langle \mathcal{T}^+ L_e \mid W_e \right\rangle \;=\; \left\langle L_e \mid (\mathcal{T}^*)^+ W_e \right\rangle \;=\; \langle L_e \mid W \rangle \tag{1.80}$$

equation (1.79) is the adjoint of (1.26). It can also be written in integral form, instead of in operator form:

$$W(\mathbf{s}) \;=\; W_e(\mathbf{s}) \;+\; \int_D K(\mathbf{r},\mathbf{s})\, W(\mathbf{r})\, d\mu(\mathbf{r}) \tag{1.81}$$

It is usually called the importance transport equation. As can be seen, the structure of both equations is the same. W_e plays the role of L_e (this is the reason way it is called emitted importance), and W plays the role of L. So, importance can be seen as a quantity which is transported, by operator $\mathcal{T}^*$, as radiance is transported by $\mathcal{T}^*$. Finally, function W is called simply as the importance or potential function. The value $W(\mathbf{r})$ is the fraction of $L_e(\mathbf{r})$ which contributes to the inner product in (1.80).

1.5 Computational methods.

Once the equations defining radiance and importance transport have been explained, in this section we briefly introduce the basics of the methods used to obtain an approximation to that function. These methods can be classified in two families: Monte-Carlo methods and Finite-Element methods. Both techniques are rooted in very different principles, which are introduced here.

1.5.1 Basis sets and projection operators.

Integrable functions in a arbitrary domain can be added between them and can be multiplied by real values. This gives that set the structure of a vector space. Lets consider any finite set $\mathbf{B} = \{b_1, b_2 \ldots b_n\}$ of n integrable functions in domain D. Lets assume that this is an orthonormal set of functions, that is, $\langle b_i \mid b_j \rangle = \delta_{ij}$ for all $i, j \leq n$, where δ_{ij} is 1 when $i = j$ and 0 otherwise. The set $\mathbf{B}$ can be taken

as the basis of a $n-$dimensional vector space $V_{\mathbf{B}}$. For any function h in $V_{\mathbf{B}}$, there exists a set of n real values $\{c_1, \ldots c_n\}$ such that

$$h \;=\; \sum_{i=1}^{n} c_i \, b_i \tag{1.82}$$

it is usually said that $V_{\mathbf{B}}$ is the vector space spawned by $\mathbf{B}$. Function h is said to be a *linear combination* of the functions b_i.

For any integrable function f, we can find the nearest function f' to f such that f is in $V_{\mathbf{B}}$. Here the term *nearest* is used in the sense induced by the L_1-norm. That is, the value $|f' - f|_1$ is smaller than any other value $|g - f|_1$ where $g \in V_{\mathbf{B}}$. In the case that $\mathbf{B}$ is orthonormal, then we can obtain f' directly from f and $\mathbf{B}$, because

$$f' \;=\; \sum_{i=1}^{n} \langle f \mid b_i \rangle \; b_i \tag{1.83}$$

The function f' is called the *projection* [4] of f onto $V_{\mathbf{B}}$. We can consider the operator which takes any function g and returns its projection onto $\mathbf{B}$. This operator is written as $\mathcal{P}_{\mathbf{B}}$, and then $f' = \mathcal{P}_{\mathbf{B}} f$. This operator is defined by

$$(\mathcal{P}_{\mathbf{B}} f)(x) \;=\; \sum_{i=1}^{n} \langle f \mid b_i \rangle \; b_i(x) \tag{1.84}$$

It is easy to show that these operators are always linear, by using the linearity of inner products (1.68) and the previous definition. If $f \in V_{\mathbf{B}}$ then $\mathcal{P}_{\mathbf{B}} f = f$. This implies that, for all $n > 0$, $\mathcal{P}_{\mathbf{B}}^n = \mathcal{P}_{\mathbf{B}}$.

The values $\langle f \mid b_i \rangle$ are called the *coordinates* of f with respect to $\mathbf{B}$. Note that any function in $V_{\mathbf{B}}$ is fully determined by its coordinates, and this allows to represent such functions in a computer memory using a finite amount of memory.

1.5.2 Finite elements methods.

The key of Finite-Elements methods (FE in what follows) is the conversion of equation (1.44) into a discrete matrix equation. This is done by trying to solve for a equation which is related to (1.26), although not exactly equal. The approximated equation solved is

$$f' \;=\; \mathcal{P}_{\mathbf{B}} \left(g + \mathcal{A} f' \right) \tag{1.85}$$

Function g must be given as an input data, then it has to be stored in a computer memory. Usually this done by using a g which is in $V_{\mathbf{B}}$, and then it is stored as a vector of coefficients. Then $g = \mathcal{P}_{\mathbf{B}} g$. As $\mathcal{P}_{\mathbf{B}}$ is linear then we can rewrite (1.85) as

$$f' \;=\; g \;+\; (\mathcal{P}_{\mathbf{B}} \mathcal{A}) \, f' \tag{1.86}$$

[4] The term *projection* is used here because this operation, when done on the three dimensional vector space of points, is the perpendicular projection of a point on a plane or a line.

Equation is very similar to that in (1.44) except for the use of the operator $\mathcal{P}_{\mathbf{B}}\mathcal{A}$ instead of $\mathcal{A}$. The solution function f' is in $V_{\mathbf{B}}$, that is $\mathcal{P}_{\mathbf{B}}f' = f'$. This method is called the *Galerkin* method in the literature about finite element methods. There exist other algorithms for discretizing the original continuous equation, although this is the most frequently used in Global Illumination.

It is easy to check that equation (1.86) leads to Galerkin method, because the projection of residual function is zero. The residual function r is defined as $(\mathcal{I} - \mathcal{A})f' - g$, and is taken as a measure of the error involved in approximating f by using f'. Note the real error can not be known because f is unknown, but the residual function r is only based in known entities, and goes to zero when f' is equal to f. Each finite element methods imposes a condition on the residual function. In the case of Galerkin method, it is established that its projection must be zero, that is, $\mathcal{P}_{\mathbf{B}}r = 0$. To check this, we can expand r, and then use the equality (1.85)

$$\begin{aligned}
\mathcal{P}_{\mathbf{B}}\, r &= \mathcal{P}_{\mathbf{B}}((\mathcal{I} - \mathcal{A})f' - g) \\
&= (\mathcal{P}_{\mathbf{B}}f' - \mathcal{P}_{\mathbf{B}}\mathcal{A}f') - \mathcal{P}_{\mathbf{B}}g \\
&= (f' - \mathcal{P}_{\mathbf{B}}\mathcal{A}f') - g \\
&= g - g \\
&= 0
\end{aligned}$$

where this last 0 means the function equal to zero in all the domain.

Lets call $\{f_1 \ldots f_n\}$ to the coefficients of f' with respect to $\mathbf{B}$. Then the function $(\mathcal{P}_{\mathbf{B}}\mathcal{A})f'$ can be rewritten as follows

$$\begin{aligned}
(\mathcal{P}_{\mathbf{B}}\mathcal{A})\, f' &= (\mathcal{P}_{\mathbf{B}}\mathcal{A}) \sum_{i=1}^{n} f_i\, b_i \\
&= \sum_{i=1}^{n} f_i\, \mathcal{P}_{\mathbf{B}}\, (\mathcal{A}b_i) \\
&= \sum_{i=1}^{n} \left[f_i \sum_{j=1}^{n} \langle \mathcal{A}b_i \mid b_j \rangle\, b_j \right]
\end{aligned}$$

note that we have used the linearity of both $\mathcal{P}_{\mathbf{B}}$ and $\mathcal{A}$, and then the definition of $\mathcal{P}_{\mathbf{B}}$. Equation (1.86) can now be expanded

$$\sum_{i=1}^{n} f_i\, b_i = \sum_{j=1}^{n} g_j\, b_j + \sum_{k=1}^{n} f_k \sum_{l=1}^{n} c_{lk} b_l \tag{1.87}$$

where $c_{lk} = \langle \mathcal{A}b_k \mid b_l \rangle$ and $\{g_1 \ldots g_n\}$ are the coordinates of g with respect to $\mathbf{B}$. This last equation is a functional equation, which also holds when we consider the evaluation of both sides at any point x of the domain. Reorganizing the terms, we obtain, for all points x, the following equality

$$\sum_{i=1}^{n} \left[f_i - g_i - \sum_{j=1}^{n} c_{ij}\, f_j \right] b_i(x) = 0 \tag{1.88}$$

this last equation can only hold when the multipliers of $b_i x(x)$ are all zero, that is, when

$$f_i \;=\; g_i \;+\; \sum_{j=1}^{n} c_{ij}\, f_j \tag{1.89}$$

for all $i \in \{1, \ldots, n\}$. This equation can be written as a matrix equation

$$\begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix} \;=\; \begin{pmatrix} g_1 \\ \vdots \\ g_n \end{pmatrix} \;+\; \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & & \vdots \\ c_{n1} & \cdots & c_{nn} \end{pmatrix} \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix} \tag{1.90}$$

Let us define $F = (f_1, \ldots, f_n)^T$, that is, F is the column vector of the coefficients of f with respect to $\mathbf{B}$. We also define $G = (g_1, \ldots, g_n)^T$ and

$$A \;=\; \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & & \vdots \\ c_{n1} & \cdots & c_{nn} \end{pmatrix} \tag{1.91}$$

then, equation (1.90) can be more compactly rewritten as

$$F \;=\; G \;+\; A\,F \tag{1.92}$$

the previous equation is equivalent to (1.85) although it does not involves functions but their coefficients with respect to $\mathbf{B}$

1.5.3 Monte Carlo Methods.

An alternative to the previous solution is to use Markov Chains to solve (1.44). Instead of finding a projected approximation of f, by using Monte Carlo methods we obtain a stochastic approximation (with a given variance) of the scalar product of f and any other function h, that is $\langle f \mid h \rangle$. [Rubinstein81].

To develop this, we need to introduce Markov Chains. A Markov Chain (or Random Walk) c is any infinite series of elements in D, $c = \{c_0, \ldots c_i, \ldots\}$. Usually, we will call states to the elements of a chain.

We consider the set of every possible Markov Chain. In MC methods, we need to randomly select a subset of Markov Chains from that set. This is done by using a probability density function, *pdf*. This *pdf* is separable; in other words, the probability of selecting a single chain c is:

$$Q(c) \;=\; p(c_0)P(c_0, c_1)P(c_1, c_2)\ldots$$

where $p_0(x)$ is the probability to select the first point, and $p(y, z)$ is the conditional probability of selecting z as the next point given that the previous one was y. These functions obey the following properties ($\forall x, y \in D$)

$$\int_D p(y, z)dz \quad = \quad 1$$

$$\begin{aligned} \int_D p_0(z)dz &= 1 \\ h(x) \neq 0 &\rightarrow p_0(x) > 0 \\ k(x,y) \neq 0 &\rightarrow p(x,y) > 0 \end{aligned}$$

With these definitions, we may introduce the following function for every Markov Chains c [Rubinstein81]

$$V(c) = \frac{h(c_0)}{p_0(c_0)} \sum_{i=0}^{\infty} U_i \, g(c_i)$$

where:

$$U_i = \frac{K(c_0,c_1)}{p(c_0,c_1)} \cdots \frac{K(c_{i-1},c_i)}{p(c_{i-1},c_i)}$$

We define the random variable obtained as the set of values $V(c)$ for every Markov Chain c. Each value $V(c)$ has probability $Q(c)$. The fundamental fact about this is that the average value of that random variable is exactly $\langle h \mid \mathcal{A}^+ g \rangle$. This gives the following procedure to compute that value:

1. Select the probability density functions p and P obeying the required properties.
2. Randomly select a set of n chains $\{c^1, \ldots, c^n\}$ with probability density Q, with n large enough.
3. Estimate the result by using the average of those samples, that is:

$$\langle h \mid f \rangle \approx \frac{1}{n} \sum_{i=1}^{n} V(c^i) \tag{1.93}$$

Note that we have to deal with infinite Markov Chains, which is not computationally affordable. This is solved by either truncation to finite Markov Chains, which yields some residual error, or by expanding D with a new element called *absorption state*, say a, such that for every x in D, $p(x,a) > 0$, $p(a,x) = 0$, $p(a,a) = 1$, $p_0(a) = 0$, $k(a,x) = k(x,a) = 0$, $g(a) = 0$. With these, the chain will fall almost surely in the absorption state after a number of finite steps.

Adjoint Formulation.

In global illumination, we do not wish to compute just $\langle h \mid f \rangle$, but the set of values $\langle h_i \mid f \rangle$ for a given set of m functions $\{h_1 \ldots h_i \ldots h_m\}$. By using the available results, we have two options to achieve this. The first of them is to find, for every i independently, the value $\langle h_i \mid f \rangle$, as stated in the previous section. This requires the creation of mn chains in total.

Another option is to use the adjoint of $\mathcal{A}^+$, which is $(\mathcal{A}^*)^+$. By using the properties of the adjoint, we know that:

$$\langle h_i \mid f \rangle = \langle h_i \mid \mathcal{A}^+ g \rangle = \langle g \mid (\mathcal{A}^*)^+ h_i \rangle \tag{1.94}$$

By using the last form, the following procedure is obtained:

1. By using a selected p, obtain n random chains (c^j) with that probability distribution.

2. For every h_i, estimate the scalar product as:

$$\langle h_i \mid f \rangle = \langle g \mid \mathcal{A}_a^+ h_i \rangle \approx \frac{1}{n} \sum_{j=1}^{n} V_i(c^j) \tag{1.95}$$

where

$$V_i(c) \;=\; \frac{g(c_0)}{p_0(c_0)} \sum_{j=0}^{\infty} U_j \, h_i(c_j)$$

and

$$U_j \;=\; \frac{K(c_1, c_0)}{p(c_0, c_1)} \cdots \frac{K(c_j, c_{j-1})}{p(c_{j-1}, c_j)}$$

This requires n chains, instead of mn. Note that the arguments of k are reversed with respect to the previous formulation. This is due to the use of the adjoint operator, instead of the original one. (see equation 1.71)

1.6 Conclusions.

This chapter includes the definitions of the functions which characterize the flux of electro-magnetic radiation, by using the particle model of light. The definition of these function is done by using ratios of measure function (that is, differentiation).

After these functions have been introduced, we derive an integral radiance equation which has the form of a second order integral equation of the second kind. This is done by deriving the expression for function K which is the kernel of the integral operator involved in that equation. A number of numerical algorithm have been previously designed and applied for that kind of integral operators, and can be also applied to that radiance equation. Previous forms for the radiance equation do not have this form, because the kernel is a three variable function as in [Kajiya86], or because the transport operator is defined as the composition of other two, as in [Arvo95].

In this chapter we also use the previous formulation in order to briefly introduce existing computational methods for integral equations, that is, finite element methods (we restrict to Galerkin method) and Monte-Carlo methods (path-tracing and particle tracing). This introduction to numerical methods uses standard formulation which can be applied to any second order integral equation. Thus, we observe that the given expression for the radiance equation is useful in order to derive numerical methods which solve it.

Chapter 2

Survey of Global Illumination Methods.

2.1 Image and observer definition.

Once L is characterized, this gives the luminous radiance at every pair of points in an environment. But the goal of GI techniques is to obtain images. An image, I, is defined as a function of D, such that $I(\mathbf{r})$ is the amount of radiance transported through $\mathbf{r}$ which a given observer perceives directly. In order to store and display I on a computer, we need to sample it at a discrete resolution, by using some projection operator onto a basis which yields a finite set of coefficients or samples. These values are used to set pixels colors.

The selected basis which is used to build this projection operator is an orthonormal set of n functions $\mathbf{W} = \{W_{e1}, \ldots, W_{en}\}$, such that the scalar value $W_{ei}(\mathbf{r})$ gives the fraction of radiance traveling through $\mathbf{r}$ which contributes to the i-th sample of the image. When we consider the projection operator $\mathcal{P}_{\mathbf{W}}$, the image function can be write as follows:

$$I = \mathcal{P}_{\mathbf{W}}\, L \tag{2.1}$$

that is I belongs to $V_{\mathbf{W}}$ and can be represented by a finite set of values (which are its coefficients with respect to $\mathbf{W}$). The previous equality can be written by considering the evaluation of I in a ray $\mathbf{r}$ in D

$$I(\mathbf{r}) \;=\; (\mathcal{P}_{\mathbf{W}}\, L)(\mathbf{r}) \;=\; \sum_{i=1}^{n} W_{ei}(\mathbf{r})\, l_i$$

where $l_i = \langle W_{ei} \mid L \rangle$.

Ideally (in the traditional perspective camera model), the each W_{ei} function must be non zero for all the rays passing through the focus and the rectangle (on the viewplane) related to pixel number i. Unfortunately just a set of discrete

samples can be taken at each pixel. This implies that usually, the W_{ei} functions are approximated as follows:

$$W_{ei}(\mathbf{r}) \;=\; \sum_{j=1}^{m} w_{ij}\delta(\mathbf{r}-\mathbf{s}_{ij})$$

where $\mathbf{s}_{ij}$ is the ray passing through the focus and the j-th sample point in the i-th pixel on the viewplane. The set of weights w_{ij} are normalized in the sense that, for all pixels i, $\sum_{j=1}^{m} w_{ij} = 1$. In the case of some special camera, the focus may have some extension (the image appears *blurred*), but the above expression remains unmodified, except that the position of the samples $\mathbf{s}_{ij}$ must account for a point sampling of focus area. Similar considerations arose when considering lenses in the camera model.

With all these results, the global illumination problem is stated as follows: for a given radiance emission distribution L_e, a given transport operator $\mathcal{T}$, and a given observer $\mathbf{W}$, compute a finite representation of the image I the observer perceives, by using the following relation:

$$I \;=\; \mathcal{P}_{\mathbf{W}}\,\mathcal{T}^{+}\,L_e \tag{2.2}$$

We obtain I as a discrete set of coordinates with respect to the basis $\mathbf{W}$. These coordinates are then interpreted as pixels intensities, and then the image can be displayed on any device. This formulation is a general one, and includes any geometry for the observer, and an arbitrary transport operator at a given environment.

2.2 Solutions.

Once the problem has been stated, and the tools used to solve it have been presented, we can now give the GI methods in terms of those tools.

Local methods.
: This family of methods does not really solve for GI, but just for local illumination This allows simple and efficient algorithms to be used, at the cost of lower realism. The computed image accounts for a single reflection of light onto the objects, and a local operator (T) is used instead of the global one ($\mathcal{T}^{+}$). Formally: $I = \mathcal{P}_{\mathbf{W}}\; T\; L_e$. Simple ray-tracing, z-buffer and scan-line methods fall into this category.

Discretization or Finite Element (FE) Methods.
: We include here every method based on using a projected operator instead of the original one. The problem is then converted into the solution of a given system of equations, obtaining a projected version of L. Then, we project L onto $\mathbf{W}$ to obtain the image. Formally, these two steps are as follows (1) $L = (\mathcal{P}_{\mathbf{B}}\mathcal{T})^{+}L_e$ and (2) $I = \mathcal{P}_{\mathbf{W}}\; L$. Note that although these are called FE methods, we can use MC methods to obtain the projected kernel coefficients (form factors).

Stochastic or Monte Carlo methods.

These methods use Markov Chains to obtain the solution of the equation (2.2). We get a stochastic approximation to the solution, with some given variance. MC methods are subdivided into two sub-categories:

- **MC path tracing from the observer.** In this case, for every W_{ei} (independently) we obtain $l_i = \langle W_{ei} \mid \mathcal{T}^+ L_e \rangle$, by using the formulation given in ???
- **Photosimulation or MC path tracing from light sources.** We obtain a simultaneous solution for $l_i = \langle L_e \mid \mathcal{T}_a^+ b_i \rangle$ (for a given basis $\mathbf{B} = \{b_1 \dots b_n\}$) by using the second option in sub section 1.5.3. Formally, we compute $L' = \mathcal{P}_{\mathbf{B}} \mathcal{T}^+ L_e$. Once this is obtained, we projected the L function by using some given observer: $I = \mathcal{P}_{\mathbf{W}} L'$. Note that although we use a basis $\mathbf{B}$, this is necessary just to get a finite computer representation L' of L, and no projected operator is used.

In following sections, further details about each family of methods are given, and the main papers related to them are indicated.

2.3 Local Methods.

These methods were the first ones developed in order to synthesize realistic images. The restriction to a local operator yields a significant income in computing time and storage. We subdivide these algorithms in projection methods (find at which pixels each object is seen) and ray-tracing methods (find what object is seen at each pixel).

2.3.1 Simple projection methods.

In these methods, we compute the portion of screen covered by each scene object. At each pixel displayed, we compute the direct contribution of light onto the observer and the contribution after a single bounce. Formally:

$$I \;=\; \mathcal{P}_{\mathbf{W}} (\mathcal{T} + \mathcal{I}) L_e$$

usually, the following assumptions are taken to simplify computations:

- L_0 is restricted to a set of punctual light sources, (with intensity l_i and position x_i) that is, $L_e(\mathbf{r}) \;=\; \sum_{i=1}^{n} l_i \, \delta(x_i - \mathbf{r}_s)$
- The operator $\mathcal{T}$ includes a Phong like term and a diffuse term, that is $\mathcal{T} = \mathcal{G} + \mathcal{D}$, and does not account for occlusions.

Here we include the well known z-buffer and scan-line algorithms for synthesizing images, and using Phong or Gouroud shading. A classical reference for this is [Phong75].

2.3.2 Simple Ray Tracing.

This well-known technique uses a tree of rays at every pixel to follow up (in the reverse order) the path which light travels. Its advantage, with respect to projection methods, is that we are now able to trace perfect specular reflections. The assumptions taken about $\mathbf{W}$ and L_0 are similar to projection methods. Operator $\mathcal{T}$ now includes a global specular component. This yields the following image:

$$I \;=\; \mathcal{P}_{\mathbf{W}}\,(\mathcal{I}+\mathcal{D}+\mathcal{G}+\mathcal{S}^{+})\,L_e$$

The first reference for a full ray-tracing method (including specular reflections) is [Whitted80]

2.4 Discretization or Finite Elements Methods

This section includes every method in which a projected operator is used to solve the GI problem. This has probably been the option taken by the largest number of researchers. Once given $\mathbf{W}$, $\mathcal{T}$ and L_e, we have to select some finite basis function set $\mathbf{B}$ and then obtain $I = \mathcal{P}_{\mathbf{W}}\,(\mathcal{P}_{\mathbf{B}}T)^{+}\,L_e$, by solving the associated equations system. Due to the large number of papers in this category, we have subdivided it in several sections, and are listed them in chronological order.

2.4.1 Classic radiosity.

In this section we will discuss the methods which deal with pure diffuse radiosity and constant functions for the basis. We name $\mathbf{C}$ the set of basis functions $\{C_i\}$ such that there exist a disjoint set of surface areas $\{A_i\}$ covering the surfaces, and such that

$$C_i(\mathbf{r}) \;=\; \begin{cases} 1/A_i & \text{when } \mathbf{r}_s \in A_i \\ 0 & \text{in any other case} \end{cases} \tag{2.3}$$

The operator used, D, obeys the property that yields a L function which does not depend on ray direction. Therefore the problem is stated just on surface points, and we talk about radiosity (which is not defined in terms of solid angle) instead of radiance. If we call B to that radiosity function, the problem is a follows: find B such that $B = (\mathcal{P}_{\mathbf{C}}D)^{+}B_e$. The function B_e is given as a set of coefficients e_i called emission of the patches. The coefficients of the projected kernel are expressed as $k_{ij} = F_{ji}R_i$, where R_j is called reflectivity of patch j and F_{ij} is called form-factor. All these entities may be assigned a physical meaning, which is not developed here. The method proceeds in two steps, as follows:

$$B' \;=\; (\mathcal{P}_{\mathbf{C}}D)^{+}\,B_e \tag{2.4}$$

$$I \;=\; \mathcal{P}_{\mathbf{W}}\,B' \tag{2.5}$$

Once B' (the approximated radiosity function) is obtained, the second step may be repeated for further observers. The solution to the equation system is usually done by iterative numerical methods [Gortler94]. Mainly one of the following two methods is used:

Reference	Form.Fact.	Solv.Eqn.Sys.
[Goral84]	Stokes	Gauss-Seidel
[Cohen85]	Hemicube	Gauss-Seidel
[Cohen88]	Hemicube	Progressive
[Malley88]	Local MC	
[Wallace89]	Quadrature	Progressive
[Baum89]	Stokes+HC	Progressive
[Sbert93]	Global MC	

Table 2.1: Classic radiosity methods.

- Gauss-Seidel (GS) (Jacobi method with in place updates)
  This method requires simultaneous storage of every form factor.

- Progressive Radiosity or Southwell Iteration (PR).
  Just a row of form factors needs to be stored at each step. Allows intermediate results to be presented.

Form factor computation usually demands a significant amount of CPU time. There are great many of options for this calculation, such as:

- Hemicube methods.
  The computational device used to project B' onto $\mathbf{W}$ is also used to project D onto $\mathbf{C}$, by resorting to z-buffer techniques. A row of form factors is obtained simultaneously.

- Quadrature or Random Quadrature.
  Every FF is (independently) obtained by taking a finite number of samples of the involved integral.

- Local hit-miss Monte-Carlo.
  By using stochastic ray-casting, a row of form factors is obtained at each step (rays from a patch are used to compute form factors from that patch).

- Global hit-miss Monte-Carlo.
  Related to previous methods, but the full FF matrix is obtained (every ray is used to compute form-factor of arbitrary patches pairs). This has been called the *swords* method.

- Stokes methods.
  By applying Stokes theorem, the FF computation may be done using analytical methods with no errors, in the case of no occlusions. This is merged with other methods in order to handle occlusions.

Table 2.1 lists the main contributions in this section, including the options (from above) used in each paper.

Reference	Basis	Init.Guess
[Immel86]	Hemicube	Emitted
[Shao88]	Hemicube	Diffuse
[Buckalew89]	Links	Emitted

Table 2.2: One-pass methods for non-diffuse environments

2.4.2 One-pass methods for non-diffuse environments.

These methods are direct extensions of classic radiosity, attempting to include perfect specular and/or glossy operators additional to the diffuse one, that is, the following image

$$I = \mathcal{P}_{\mathbf{W}} (\mathcal{P}_{\mathbf{B}} T)^{+} L_e \tag{2.6}$$

is computed, where $\mathcal{T} = \mathcal{S} + \mathcal{D} + \mathcal{G}$. A two step computation is carried out, similar to classic radiosity, as follows:

$$\begin{aligned} (1)\ L' &= (\mathcal{P}_{\mathbf{B}}(\mathcal{S} + \mathcal{D} + \mathcal{G}))^{+} L_e \\ (2)\ I &= \mathcal{P}_{\mathbf{W}} L' \end{aligned}$$

In this case, we deal with radiance instead of radiosity, and the problem has an additional dimension with respect to classic radiosity. This implies larger computation time. There are two options on the basis $\mathbf{B}$ selected to achieve this:

- Hemicubes
  Each B_i is related to a pixel on a hemicube over a patch center (or vertex). $B_i(\mathbf{r})$ is non zero for rays throughout that pixel from the vertex, 0 in any other case.

- Links
  A high density structure of rays (links) is built. Each links connects two patches. Each basis B_i is related to each of these links (a ray called $\mathbf{l}_i$). Then we define the basis as follows: $B_i(\mathbf{r}) = \delta(\mathbf{r} - \mathbf{l}_i)$.

Once stated, the equations system is usually solved by Gauss-Seidel. This requires setting an initial estimation of radiances before the first step, and there are two options for this:

- Use L_e, the emitted radiance function.

- Use $L' = (\mathcal{P}_{\mathbf{B}}\mathcal{D})^{+} L_e$, that is, radiance computed with the assumption of just diffuse transport (by classic radiosity). In the case that the number of non-diffuse surfaces is small, this initial guess is close to the final solution, and the process converges faster.

Table 2.2 shows some contributions in this section. Two problems arise by using these methods:

- Requires large amounts of memory and computation time.
- The perfect specular operator is hardly implemented, because the associated BRDF is based on a Dirac delta function [Cohen93] such that projection of S on any finite basis yields a significant error.

2.4.3 Two-pass methods for non-diffuse environments.

Two-pass methods are used in order to solve the problems stated above. These methods are based on the separation of a general $\mathcal{T}$ operator as the addition of another two

$$\mathcal{T} \;=\; \mathcal{T}_1 + \mathcal{T}_2 \tag{2.7}$$

the use of the subindex 1 and 2 is due to the fact that $\mathcal{T}_1$ is dealt with in the first pass and $\mathcal{T}_2$ in the second. In order to do this, first we show an interesting property which this decomposition of $\mathcal{T}$ holds. This property is

$$\mathcal{T}^+ \;=\; \mathcal{T}_2^+ \,(\mathcal{T}_1\,\mathcal{T}_2^+)^+ \;=\; \mathcal{T}_2^+\,\mathcal{U}^+ \tag{2.8}$$

where $\mathcal{U} = \mathcal{T}_1\;\mathcal{T}_2^+$ With these we can compute an approximation to $\mathcal{U}^+\;L_e$ by using projection methods, and then compute the image by a ray-tracing like algorithm. This can be expressed as the following two stage algorithm

$$\begin{array}{lrcl} (1) & L' & = & (\mathcal{P}_{\mathbf{B}}\,\mathcal{U})^+\,L_e \\ (2) & I & = & \mathcal{P}_{\mathbf{W}}\,\mathcal{T}_2^+\,L' \end{array} \tag{2.9}$$

Usually, the $\mathcal{D}$ (diffuse) operator is included in $\mathcal{T}_1$, and the $\mathcal{S}$ (perfect specular) operator is included in $\mathcal{T}_2$. Step (1) is equal to previous finite element methods, except for the coefficients of $\mathcal{P}_{\mathbf{B}}\mathcal{U}$, which are called *extended form factors*, because they also account for the $\mathcal{T}_2^+$ term in $\mathcal{U}$. The function L' accounts for all the transport paths ending in a reflection modeled by operator $\mathcal{T}_1$. Step (2) is formally equal to a ray-tracing method (or to a Monte-Carlo path tracing from the observer), except for the use of L' instead of L_e. This second step accounts for all transport paths beginning after a $\mathcal{T}_1$ reflection, going through $\mathcal{T}_2$ reflections and ending at the observer. These methods are shown as a combination of ray-tracing and radiosity.

There are two options for the selection of $\mathcal{T}_1$ and $\mathcal{T}_2$ operators, based on which of the two accounts for the glossy ($\mathcal{G}$) term:

- $\underline{\mathcal{T}_1 = \mathcal{D} \text{ and } \mathcal{T}_2 = \mathcal{G} + \mathcal{S}.}$
  In this case L' (the result of the first step) is called the diffuse emission. We have to resort to stochastic ray tracing at second step, in order to handle the $\mathcal{G}$ term in $\mathcal{T}_2$.
- $\underline{\mathcal{T}_1 = \mathcal{D} + \mathcal{G} \text{ and } \mathcal{T}_2 = \mathcal{S}.}$
  Here L' is directionally dependent. Consequently the first step is more complicated whereas the second step is just a simple ray tracing procedure accounting for just perfect specular reflections

Reference	$\mathcal{T}_1$	$\mathcal{T}_2$	Basis Functions	Extend.Form factors
[Wallace87]	$\mathcal{D}$	$\mathcal{S}+\mathcal{G}$	Constant	Virtual Patches
[Sillion89]	$\mathcal{D}$	$\mathcal{S}+\mathcal{G}$	Constant	Recursive ray-casting
[Sillion91]	$\mathcal{D}+G$	$\mathcal{S}$	Spherical Harmonics	Virtual Patches

Table 2.3: Two-pass methods for non-diffuse environments

A progressive method is used to solve the equation system. At each step, energy from the shooting patch must be relayed throughout specular bounces before a diffuse patch is found, in order to account for extended form factors. Two options are available for this:

- We can use a *virtual patch* behind every mirror to duplicate the shooting patch, and add the form factor from this to the extended form factor.
- Use recursive ray-casting in the case that a shooting patch "sees" a perfect specular one.

Table 2.3 summarizes the work that has been done.

2.4.4 Wavelet and Importance methods.

The complexity order for all of the previous finite elements methods is at least $O(n^2)$, where n is the number of basis functions in the set $\mathbf{B}$. As this number increases, the quality of the approximation does so. However the complexity grows quadratically. In practical applications, the number of bases functions (patches) required to obtain a reasonable quality is so large that those methods become unusable. The complexity comes mainly from computation of the projected operator.

One option to solve this in the context of FE methods is the use of wavelet or hierarchical basis functions sets. These techniques yield lower complexity for projected operator computation and equation system solution. They have been used in other computing fields such as curve design, robot planning, and image analysis and compression.

2.4.5 Wavelet projection.

As described in ([Mallat89]), a Multiresolution Analysis (MRA) is a set of vector spaces $\{V_i\,, i \in N\}$ such that $V_i \subset V_{i+1}$ and that $V_i \subset \mathcal{L}^2(D)$. In other words, each V_i is a set of square integrable functions on a domain D. Let us call $\mathbf{S}_i$ to any orthonormal basis of V_i. The definition of a MRA includes additional restrictions which ensure that $P_{\mathcal{S}_i} f$ converges to f when i grows to infinity. Each $\mathbf{S}_i$ is a finite basis which approximates any function at increasing resolutions when i grows. [1]

[1] for simplicity, we assume here that we are dealing with functions over the unit interval, although the definition of a MRA stands for functions on the whole real line.

As $V_i \subset V_{i+1}$ we can consider the complementary vector space of V_i in V_{i+1}. We call it U_i. When we add functions of V_i with functions in U_i, we get functions in V_{i+1}. We can state that functions in U_i encode the differences between V_i and V_{i+1}. Lets now call $\mathbf{D}_i$ to an orthonormal basis of U_i. Every element in $\mathbf{S}_i$ is orthonormal to any in $\mathbf{D}_i$, and $\mathbf{S}_i \cup \mathbf{D}_i$ is also a basis for V_{i+1}. The functions in every $\mathbf{S}_i$ are called *scaling* functions, and those in $\mathbf{D}_i$ are called *detail* functions.

Now we are able to build an hierarchical basis function set $\mathbf{H}_n$ from those basis, as follows:

$$\mathbf{H}_n = \mathbf{S}_0 \bigcup_{i=0}^{n} \mathbf{D}_i$$

When we project f against $\mathbf{H}_n$, we get a coarse description of f and successive differences with successive finer descriptions, up to resolution n. In the case that f is well described at a given resolution, the coefficients (differences) for finer resolutions are too small. In what follows we use the symbol $\mathcal{P}_n$ for the operator which maps any function to its projection on $\mathbf{H}_n$.

For any given operator $\mathcal{T}$, the modified operator $\mathcal{P}_n\mathcal{T}$ is represented by an array of coefficients. If we set to zero all the entries below a predefined constant threshold value, then we obtain a very sparse matrix [Beylkin91, Alpert93], such that operations on that matrix are done in $O(n)$ complexity, where n is the number of basis functions.

Hierarchical methods take advantage of that property, and are based on computing a projected operator against some given basis $\mathbf{H}$, by using a recursive approach which avoids the computations of the entries below a given threshold, and such that this step shows $O(n)$ complexity.

2.4.6 Families of wavelets.

The simplest wavelet basis used in Global Illumination is the Haar wavelet. For simplicity, we assume that the domain of the wavelets is the unit interval, $D = [0, 1]$. The set $\mathbf{S}_i$ has 2^i elements (basis functions), and can be defined as:

$$\mathbf{S}_i \;=\; \{\phi_{ij}(x) = 2^{i/2}\,\phi_{00}(2^i x - j) \quad | \quad j = 0, \ldots, 2^i - 1\}$$

where $\phi_{00}(x)$ is 1 when $x \in [0, 1]$, 0 otherwise. That is, every scaling function is a scaled and translated version of ϕ_{00}. These Haar wavelet are easy to deal with. Unfortunatelly, the spaces V_i just include discontinuous, piecewise constant functions. In order to overcome these limitations, Haar basis can be further generalized in two ways:

- Instead of using just one constant basis in $\mathbf{S}_0$, we can include in it a set of orthonormal polynomial functions, which are a basis for polynomials of degree less than k on the unit interval. Then, V_i would include piecewise polynomial, discontinuous, functions. This wavelets are called *Multiwavelets* of order k, and are described in [Alpert93].

- Use also polynomials of degree k, but, in this case, ensure they are k times continuous differentiable on the unit interval. This can be obtained by using

the B-Spline blending functions of degree k as basis functions. These are called *Spline Wavelets*, and they can be continuous and differentiable. Unfortunately, they are not orthonormal wavelets, but semi-orthonormal wavelets.

2.4.7 BI-Refinement and Hierarchical methods.

In theory, there is just one set of scaling functions at the coarser level, but in GI problems, this number of coarser functions is at least k, the number of surfaces in the environment. This implies in fact a $O(k^2)$ complexity. Even in the case of using hierarchical basis functions, the number k^2 is (for very complex environments) so large that the resulting complexity $O(k^2)$ implies long execution times. A solution for this is the use of brightness and importance refinement (BI-refinement).

In this context, importance is the function $f = (\mathcal{T}^*)^+ W_e$ (where $W_e = \sum_i W_{ei}$), which yields for every ray $\mathbf{r}$ the fraction of energy traveling through that ray which reaches the observer (after an arbitrary number of bounces). As this definition includes the operator $(\mathcal{T}^*)^+$, computation of importance is as difficult as computation of radiance. But the importance functions gives a weight of the contribution of the error at each ray to the error in the final image for a given observer. Recall that hierarchical methods are based on discarding coefficients below a given threshold. This threshold can be lowered when the error induced by each coefficient is greater, and this error is a function of both radiance and importance.

BI-refinement is based on the simultaneous computation of radiosity and importance. This is done by using Jacobi iteration on successive approximations of both functions. In this case, at each step, the threshold error for form-factor computation is diminished, and new significant entries (links) are included in the array. This error is based on the estimation for radiosity (or radiance) and importance, which are available from all the previous Jacobi steps.

In fact, this approach yields a solution with increasing resolution in those areas which are perceived by the observer and are highly illuminated. This has been called a view-dependent resolution solution.

Hierarchical methods have only been used to solve diffuse radiosity ($\mathcal{D}$ operator) and to account for glossy reflections ($\mathcal{G}$ operator) also. In the later case, radiance may be defined on the domain of the points and direction vectors (PD), or on the domain of emitter and receiving point (PP). The table 2.4 shows some contributions in this field:

2.5 Stochastic or Monte-Carlo methods.

All the methods in this section are based on the use of Markov Chains as the basic tool to solve for GI equation. Even when Markov Chains are used, we have to resort to some finite basis in order to store the solution, although no projected operator is used.

Reference	Oper.	Basis	BI-refin.	Domain
[Hanrahan91]	$\mathcal{D}$	Haar	No	P
[Smits92]	$\mathcal{D}$	Haar	Yes	P
[Gortler93]	$\mathcal{D}$	Multiwavelets	No	P
[Aupperle93]	$\mathcal{D}+\mathcal{G}$	Haar	Yes	PP
[Pattanaik94]	$\mathcal{D}+\mathcal{G}$	Haar	No	PP
[Christensen94]	$\mathcal{D}+\mathcal{G}$	Haar	Yes	PD
[Schroeder94]	$\mathcal{D}+\mathcal{G}$	Multiwavelets	Yes	PP
[Yu95]	$\mathcal{D}$	B-Spline	No	P

Table 2.4: Hierarchical methods.

2.5.1 Monte-Carlo path-tracing from the observer.

These methods may be viewed as an extension of ray-tracing, or as an instance of MC methods for integral equations. We know that every image pixel value l_i is obtained as $l_i = \langle W_{ei} \mid \mathcal{T}^+ L_e \rangle$. The method proceeds by sequentially processing each pixel, and computing an approximation (with a given variance) to the previous value. This value is obtained by the evaluation of (1.93) for a given set of Markov Chains or paths. We may use importance sampling, such that the probability density function Q (at pixel i) is defined by using the following p_0 and p functions:

$$p_0(\mathbf{r}) = W_{ei}(\mathbf{r}) \tag{2.10}$$

$$p(\mathbf{r},\mathbf{s}) = K(\mathbf{r},\mathbf{s}) \tag{2.11}$$

$$p(\mathbf{r},a) = 1 - \int_D K(\mathbf{r},\mathbf{x}) d\mu(\mathbf{x}) \tag{2.12}$$

The first ray is selected with a pdf proportional to W_{ei} and every other ray is created from the previous one by using the K function as a conditional probability function. The absorption probability (that is, the probability to reach the a state) is made equal to the fraction of energy absorbed at the previous state. There are other options to define $P(\mathbf{r},a)$, such that the technique called *Russian roulette* [Arvo90]. When using this pure importance sampling, the equation (1.93) becomes:

$$V(c) = \sum_{i=0}^{k} L_e(c_i)$$

where k is the last state before absorption. That is, $c_j = a \ \forall j > k$. In order to further reduce variance, some techniques are usually used. One of these techniques is direct source sampling, which is based on estimating via the following formulation:

$$\begin{aligned} I &= \mathcal{P}_{\mathbf{W}} \mathcal{T}^+ L_e \\ &= \mathcal{P}_{\mathbf{W}} (\mathcal{T}^+ \mathcal{T} + \mathcal{I}) \ L_e \\ &= \mathcal{P}_{\mathbf{W}} \mathcal{T}^+ (\mathcal{T} \ L_e) \ + \ \mathcal{P}_{\mathbf{W}} \ L_e \end{aligned}$$

That is, we use $\mathcal{T}\ L_e$ instead of L_e as the source term, and we add the $\mathcal{P}_{\mathbf{W}}\ L_e$ term (light from sources which strikes the observer directly) separately. Here we have used the equality $\mathcal{T}^+ = \mathcal{I} + \mathcal{T}\mathcal{T}^+$, which was previously stated in (1.55). The evaluation of the $\mathcal{T}\ L_e$ term at every sample can be done by adding the emitted radiance at the first hit on a surface of every chain. This technique reduces the variance because usually the L_e function is a sparse function with isolated peaks at light sources, and this implies high variance. On the other hand, the $T\ L_e$ is a smooth function with lower variance (a comparison of these options is found in [Shirley92]).

There are other options available for the p functions. When the light sources do not illuminate the surfaces directly, but do so indirectly, the method shows a very high variance even with source sampling, due to the fact that even the $T\ L_e$ function shows isolated peaks. In this case, we are able to use a different P function, such that chains are built up of two sub-chains, one starting from the observer (*pdf* for the first point proportional to W_{ei}) and the other starting from the light sources (*pdf* for the last point proportional to L_e). We'll call these a *bidirectional* pdf, in contrast with a *forward* pdf.

These techniques do not use absorption states nor infinite chains, so a finite number of iterations of the operator are computed, hence, we use $\mathcal{T}^{(k)}$ instead of $\mathcal{T}^+$, for some finite length k. Formally, this technique is rooted in the properties of the adjoint operator, because

$$
\begin{aligned}
l_t &= \left\langle \mathcal{T}^{(k)}\ L_e \mid W_{et} \right\rangle && (2.13)\\
&= \sum_{n=0}^{k} \langle \mathcal{T}^n\ L_e \mid W_{ei} \rangle && (2.14)\\
&= \sum_{n=0}^{k} \sum_{i+j=n} w_{ij}\ \left\langle \mathcal{T}^i\ L_e \mid (\mathcal{T}^*)^j\ W_{et} \right\rangle && (2.15)
\end{aligned}
$$

where w_{ij} is a set of weights such that $\sum_{i+j=n} w_{ij} = 1$ for any $n \geq 0$. These weights can be arbitrarily selected, although in [Veach95], a method for optimal selection (in the sense of reducing the variance) has been introduced (see column *Optimal Sel.* on table 2.5)

As explained, these methods proceed pixel by pixel independently, and do not use any coherence of the image at all. In the case that the target function does not show very high pixel-to-pixel variations, we may store solutions attached to the surfaces and reuse them when necessary.

Table 2.5 shows some of the published work in this topic.

2.5.2 Photosimulation or Particle-Tracing Methods.

This method, (also called Photon-tracking, MC path tracing from the light sources, shooting random walk, or backward ray-tracing) is also based on Markov Chains. In fact, it uses the same principles as the previous methods, although in this case we resort to the adjoint of the transport operator ($\mathcal{T}_a^+$ instead of $\mathcal{T}^+$). We can

Reference	PDF	Coherence	Optimal Sel.
[Cook84]	forward	No	No
[Kajiya86]	forward	No	No
[Ward88]	forward	Yes	No
[Veach94]	bidirectional	No	No
[Lafortune94]	bidirectional	No	No
[Veach95]	bidirectional	No	Yes

Table 2.5: Path tracing from the observer.

compute the function $L' = \mathcal{P}_{\mathbf{B}}\mathcal{T}^{+}L_0$ (for any basis $\mathbf{B}$) as a set of coefficients l_i by using the following:

$$l_i \;=\; \langle b_i \mid \mathcal{T}^{+}L_e \rangle \;=\; \langle L_e \mid (\mathcal{T}^{*})^{+}b_i \rangle$$

So the above computation can be viewed as an instance of (1.95) where $g = L_e$ and $h_i = b_i$. In this context, the selection of p_0 and p are usually done as follows:

$$p_0(\mathbf{r}) \;=\; \frac{L_e(\mathbf{r})}{\Phi} \tag{2.16}$$

$$p(\mathbf{r},\mathbf{s}) \;=\; K(\mathbf{r},\mathbf{s}) \tag{2.17}$$

$$p(a,\mathbf{s}) \;=\; 1 - \int_D K(\mathbf{x},\mathbf{s})d\mathbf{x} \tag{2.18}$$

where Φ is a scaling factor introduced to normalize p, that is, $\Phi = \int_D L_e(\mathbf{r})d\mu(\mathbf{r})$, the total energy emitted in the environment. In the case of using constant, disjoint basis functions (patches) as defined in (2.3), then the value $b_i(\mathbf{c}_j)$ is either 0 or $1/A_i$, and the computation of the summation over the set of basis is no longer necessary. Instead, we just need to perform the selection of the patch in which $\mathbf{c}_j$ is included. If we call n_i the number of chain states in the domain of patch number i for all the m chains, then (1.95) is further simplified to:

$$l_i \;\approx\; \frac{\Phi\, n_i}{m\, A_i}$$

We can give a physical interpretation for the process described above. Each chain is viewed as a photon, which is created based on L_e, which then bounces through the environment based on K (which is viewed as the probability function for photon state changes) and is finally absorbed at a given patch. Each photon has an energy equal to Φ/m. The division by A_i is done to obtain average energy per unit area instead of total energy. So the above equation is viewed as a simple count of the energy of the photons leaving the patch. Given that this method was first described in this context [Spanier69, Kalos86], its name is "photosimulation".

Instead of this basic selection, we may use an importance based P function, such that a higher chain density is obtained in those areas of the scene which are

Reference	Operator	Imp.	Direct.Sol.
[Arvo86]	$\mathcal{S}^+D\mathcal{S}^+$	No	No
[Heckbert90]	$\mathcal{S}^+D\mathcal{S}^+$	Yes	No
[Pattanaik92]	$(\mathcal{S}+\mathcal{D}+\mathcal{G})^+$	No	No
[Pattanaik93]	$(\mathcal{S}+\mathcal{D}+\mathcal{G})^+$	Yes	No
[Dutre94]	$(\mathcal{D}+\mathcal{G})^+$	Yes	Yes

Table 2.6: Path tracing from light sources (Shooting Random Walk)

viewed by the observer. There are several options to modify the P function in order to achieve this, see, for example, [Pattanaik93].

Once the L' function is obtained, we still have to project it against a given observer in order to obtain the image. This last step is done by using any simple projection method, as those detailed in section 2.4. If the T operator includes any perfect specular term, then L' function does not captures the details of the exact solution $L = \mathcal{T}^+L_e$. This is solved by using two-pass methods, as described in section 6.3. If $\mathcal{T} = \mathcal{T}_1 + \mathcal{T}_2$ then the process is: (recall that $U = \mathcal{T}_1\mathcal{T}_2^+$)

$$\begin{aligned} L' &= \mathcal{P}_{\mathbf{B}}\,\mathcal{U}^+\,L_e \\ I &= \mathcal{P}_{\mathbf{W}}\,\mathcal{T}_2^+\,L' \end{aligned}$$

Again, simple ray-tracing is used to achieve the second step.

Finally, a one-step photosimulation method is available for environments with no perfect specular reflections. In this case, the image is obtained directly. It may be viewed as the counterpart of direct source sampling in MC ray-tracing methods. Instead of using Markov Chains to project against a basis function set and then project against the observer, it is possible to project directly against the observer (the base $\mathbf{W}$, introduced in section 2.1). The image is directly obtained as a result of the photosimulation process.

Table 2.6 includes the articles where implementations of photosimulation has been described. The column headed as *Direct.Sol.* refers to whether the above technique has been used or not.

2.6 Conclusions.

In this chapter we introduce a formalization of the Global Illumination problem. We state that this problem is the computation of a set of functionals or inner products of the radiance function, which in turn is obtained after applying the global transport operator to the emitted radiance function.

The set of inner products is done against a set of basis functions. We call this set the *observer*. This formulation is used to introduce a number of algorithms which have been previously designed for realistic rendering. The algorithms are classified

according to the basic method used to obtain the discrete image. This allows to highlight both the differences and similarities between them.

The main benefit we obtain from the notion of observer is in the introduction of rendering algorithms. All of them can be mapped to a given expression (involving the global transport operator, and the projection operator) which states the dependence of the image function with respect to the emitted radiance function.

Chapter 3

The Variance of Monte-Carlo Methods.

3.1 Introduction.

In this chapter, we characterize the variance of Monte-Carlo methods which are based on Markov Chains.

First we define a probability measure, defined in the space of all possible Markov Chains. This probability is used to introduce a random variable evaluated over Markov Chains. Path-tracing algorithm can be viewed, in essence, as the process of sampling that random variable. By analyzing its mean, we show that path-tracing is unbiased. Moreover, we also analyze its variance, and we show how the variance function obeys an integral equation, as the radiance does. This allows to characterize the variance for a some Monte-Carlo algorithms.

A related method is used to find the variance of path-tracing with direct light source sampling.

3.2 Path-Tracing Variance

In this section we consider simple particle tracing and path-tracing methods. We show how both methods are ultimately based in sampling a given distribution of Markov-Chains. Then we analyze their variance, and we see how they apply to problems in Global Illumination. The results about the variance of this technique can be found in [Mikhailov92], although in that book the demonstration does not appear explicitly, and the results are not applied to Global Illumination. The first study of the variance of Monte-Carlo method (in the context of Global Illumination techniques) is in [Shirley91]. In this case, the photosimulation algorithm for diffuse environments is analyzed (with particles whose contribution to the estimators takes place at their last impact). Here it was introduced a bound for the variance and then it was proved that photosimulation has linear complexity with respect the number

of patches.

A deeper study of the diffuse case is in [Sbert97]. Here, the variance for both photosimulation and path-tracing are given (including estimators with just last impact contribution and estimators with contribution at all impacts). Bounds for the variance of the estimators where presented, and also it was shown linear complexity with respect the number of patches. However, this study is limited to a special case of Markov Chains. In these chains, after a particle or path hits a patch, the next exit point is randomly selected in that patch. This implies that the solution we obtain is no longer an approximation to the projection of the exact radiosity function, but it is an approximation to the solution of the discretized equation which involves a discrete matrix. Although further work must be done to extend these results to classic chains, this can be done as it is briefly described in [Sbert97b].

In this chapter, we study the problem of the variance for an arbitrary environment, with any reflection behavior, and with classic Markov Chains including any valid transition and absorption probability functions. We characterize the variance of estimators at all impacts for a general integral equation, the we apply the result to Global Illumination problems.

3.2.1 The density of Markov-Chains

The transition probability function.

Lets consider the set $D' = D \cup \{a\}$, where a is a special element called the *absorption state*, with $a \notin D$. Recall that the D is the domain where the integration of equation (1.27) is done.

Integrations in D can be extended to integrations in D' by extending the definition of the arbitrary measure m used in (1.27) to a new measure m' defined on D'. For any set $A \subseteq D'$, m' is defined as

$$m'(A) \;=\; \begin{cases} m(A) & \text{if } a \notin A \\ m(A - \{a\}) + 1 & \text{if } a \in A \end{cases} \tag{3.1}$$

the previous definition implies that $m'(\{a\}) = 1$, that is, the measure of the adsorption state is always 1, independently of the measure of the rest of elements. This definitions implies that, for any function f defined on D', whose restriction to D is integrable, we have

$$\int_{D'} f(x)\, dm'(x) \;=\; f(a) \;+\; \int_{D} f(x)\, dm(x) \tag{3.2}$$

By using the above measure, we define a *transition probability function* as any function p defined on $D' \times D'$, with the following properties, which must holds for all $r, s \in D'$

$$1 \;=\; \int_{D'} p(r,t)\, dm'(t) \tag{3.3}$$

$$1 \;=\; p(a,a) \tag{3.4}$$

$$k(r,s) > 0 \;\rightarrow\; p(r,s) > 0 \tag{3.5}$$

$$p(r,a) \;>\; 0 \tag{3.6}$$

the value $p(r,s)$ is the conditional probability for making a transition to state s, when the previous state was r, in a chain of states in D'. Equality (3.3) ensures that the function p is normalized and can be used as a probability measure. The equality (3.4) ensures that all the transitions from the adsorption state go again to the adsorption state. Equality (3.5) implies that all states s such that $k(r,s) > 0$ have a chance for being selected after r. Finally, inequality (3.6) implies that there is always a chance to fall into absorption state after any state. This, in turn, implies that it is impossible to find a chain which does not visits never the absorption state (that is, the set of infinite chains with no absorption state have a zero probability measure).

A probability measure for Markov-Chains.

Let us define C as the set of all possible infinite lists of elements in D' Formally, we can defined C as the smallest solution to the following set equation

$$C \;=\; D' \times C$$

Consider now any chain $c \in C$ with $c = \{c_0, c_1, c_2, \ldots\}$ and a element $x \in D'$. Obviously it holds that $(x,c) \in C$. In what follows, we will write xc to mean the chain (x,c). This implies that xc is the infinite chain obtained by placing x in front of c, that is $xc = \{x, c_0, c_1, \ldots\}$.

In figure 3.1 you can see you can see an example of a chain for a given environment. Note that r is an element which points towards the observer.

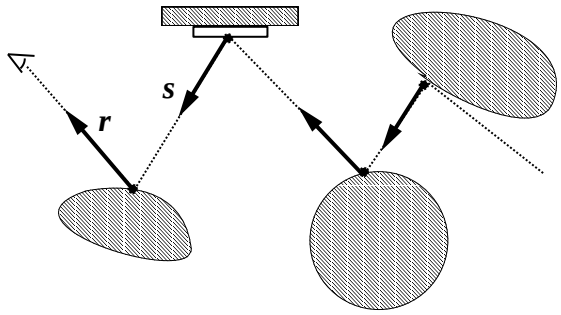

Figure 3.1: An example of a simple chain.

Once C is defined, we define a probability measure on C, named U_r, which is induced by the transition probability function. This probability measure can be used to integrate in C, in the Lebesgue sense. If we have $A \subseteq C$, we can find the measure of A by Lebesgue integration:

$$U_r(A) \;=\; \int_C dU_r(c) \tag{3.7}$$

here, $dU_r(c)$ is a differential quantity which express the differential measure of a chain c. We can view $dU_r(c)$ as the conditional probability that a chain visits $c_0, c_1, c_2, \ldots$ just after visiting r, where $c = \{c_0, c_1, c_2, \ldots\}$.

Once we have selected a transition probability function, and a element r in D', we define U_r as the measure which obeys the following properties

$$\frac{dU_r(sc)}{dm'(s)\, dU_s(c)} \;=\; p(r,s) \quad \wedge \quad U_r(C) = 1 \tag{3.8}$$

Equation (3.8) defines the relation between U_r and p. It can be used for change the variable of integration, in integrals over C. In fact it will be used to change the integration on C by integration on $D' \times C$ (both spaces are the same). The value $dU_r(sc)$ is the probability to go from state r to state s and then over all states in chain c. This probability is proportional to $p(r,s)\, dm'(s)$ (the differential probability to go from r to s) and to $dU_s(c)$, which is the probability to go over chain c after visiting s. Condition $U_r(C) = 1$ ensures that U_r is normalized and thus is a probability measure.

3.2.2 A random variable for Markov Chains

Once we have a probability measure dU_r, we can define random variables in C, with respect that probability measure. Let us define the random variable X_r. The r in the subscript stands for any $r \in D'$. This is done so because the definition of X_r is parametrized with respect to a given state r. This definition is:

$$X_r(sc) \;=\; g(r) \;+\; \frac{k(r,s)}{p(r,s)}\, X_s(c) \tag{3.9}$$

The g and k functions used here are those of (1.27). We need to extend the definition of these functions to account for the adsorption state. We make, by definition, $g(a) = 0$ and $k(r,a) = 0$, for all $r \in D'$. The mean of the random variable X_r with respect to the probability measure dU_r is:

$$E(X_r) \;=\; \int_C X_r(c)\, dU_r(c) \;=\; f(r) \tag{3.10}$$

this is an important result, because allows to use Monte-Carlo methods for finding the value $f(r)$, where f is the solution of equation (1.27). These kind of estimators has been widely used both in neutron transport problems and in Computer Graphics. They are called *gathering random walk* estimators.

In order to show the above and analyze the variance, we define the set of functions E_n for $n \geq 1$ as:

$$E_n(r) \;=\; E(X_r^n) \tag{3.11}$$

We will be interested mainly in the functions E_1 and E_2. The mean of X_r can be find as follows

$$E_1(r) \quad = \quad E(X_r) \tag{3.12}$$

$$
\begin{aligned}
&= \int_C X_r(sc)\, dU_r(sc) && (3.13)\\
&= \int_C \left[g(r) + \frac{k(r,s)}{p(r,s)}\, X_s(c) \right] dU_r(sc) && (3.14)\\
&= \int_C g(r)\, dU_r(sc) + \int_C \frac{k(r,s)}{p(r,s)}\, X_s(c)\, dU_r(sc) && (3.15)\\
&= g(r) \int_C dU_r(sc) + \int_{D' \times C} k(r,s)\, X_s(c)\, dm'(s)\, dU_s(c) && (3.16)\\
&= g(r) + \int_{D'} k(r,s) \left[\int_C X_s(c)\, dU_s(c) \right] dm'(s) && (3.17)\\
&= g(r) + \int_{D'} k(r,s)\, E_1(s)\, dm'(s) && (3.18)
\end{aligned}
$$

The integration in equation (3.18) can be limited to D instead of D', because $k(r,a) = 0$. Now, we rewrite (3.18):

$$E_1(r) = g(r) + \int_D k(r,s)\, E_1(s)\, dm(s) \qquad (3.19)$$

Now we observe that the above equation is exactly equal to (1.27). As that equation has a unique solution, then we get that $E_1 = f$, and then, at all points r, we have that $E(X_r) = f(r)$. This implies that sampling X_r yields unbiased estimators for the f function.

3.2.3 Analysis of the variance

Although we have an unbiased estimator for radiance, this estimator can be impractical due to the variance. In this section we analyst the variance involved in the estimation of $f(r)$ by using the random variable X_r. The expression for the variance is:

$$
\begin{aligned}
Var(X_r) &= \int_C \left[X_r(c) - E(X_r) \right]^2 dU_r(c) && (3.20)\\
&= E(X_r^2) - E^2(X_r) && (3.21)\\
&= E_2(r) - f^2(r) && (3.22)
\end{aligned}
$$

We explicitly define the function $V(r) = Var(X_r)$, as this is useful for our purposes. We will also use a linear integral operator $\mathcal{U}$, defined as follows:

$$\mathcal{U}F(r) = \int_D \frac{k^2(r,s)}{p(r,s)}\, F(s)\, dm(s) \qquad (3.23)$$

In what follows, we will prove that the function V can be found as the solution of an integral equation of the second kind which involves the operator $\mathcal{U}$. We now develop the expression of $E_2(r)$:

$$E_2(r) = E(X_r^2) \qquad (3.24)$$

$$= \int_C X_r^2(sc)\, dU_r(sc) \tag{3.25}$$

$$= \int_C \left[g(r) + \frac{k(r,s)}{p(r,s)} X_s(c) \right]^2 dU_r(sc) \tag{3.26}$$

$$= \int_C \left[g^2(r) + 2g(r) \frac{k(r,s)}{p(r,s)} X_s(c) + \frac{k^2(r,s)}{p^2(r,s)} X_s^2(c) \right] dU_r(sc) \tag{3.27}$$

If we do separate integration for the three terms above, we get, for the first term, that:

$$\int_C g^2(r)\, dU_r(sc) = g^2(r) \int_C dU_r(sc) \tag{3.28}$$

$$= g^2(r) \tag{3.29}$$

With respect to the second term in (3.27), we can rewrite it as follows

$$\int_C 2g(r) \frac{k(r,s)}{p(r,s)} X_s(c)\, dU_r(sc) = 2g(r) \int_C \frac{k(r,s)}{p(r,s)} X_s(c)\, dU_r(sc) \tag{3.30}$$

Using the same reasoning which converts equation (3.15) into (3.18), we find that the value (3.30) can be converted into:

$$2\, g(r) \int_D k(r,s)\, f(s)\, dm(s) = 2\, g(r)\, \mathcal{A}f(r) \tag{3.31}$$

Now, we expand the third term in (3.27):

$$\int_C \frac{k^2(r,s)}{p^2(r,s)} X_s^2(c) dU_r(sc) \tag{3.32}$$

$$= \int_{D\times C} \frac{k^2(r,s)}{p^2(r,s)} X_s^2(c) \left[p(r,s)\, dm(s)\, dU_s(c) \right] \tag{3.33}$$

$$= \int_D \frac{k^2(r,s)}{p(r,s)} \left[\int_C X_s^2(c)\, dU_s(c) \right] dm(s) \tag{3.34}$$

$$= \int_D \frac{k^2(r,s)}{p(r,s)} E_2(s)\, dm(s) \tag{3.35}$$

$$= \mathcal{U}E_2(r) \tag{3.36}$$

Putting together the three terms in equations (3.29), (3.31) and (3.36), we can rewrite (3.27), as follows:

$$E_2(r) = g^2(r) + 2\, g(r)\, \mathcal{A}f(r) + \mathcal{U}M(r) \tag{3.37}$$

putting the above as a functional equation we get

$$E_2 = g^2 + 2g\mathcal{A}f + \mathcal{U}E_2 \tag{3.38}$$

$$= (g + \mathcal{A}f)^2 - (\mathcal{A}f)^2 + \mathcal{U}E_2 \tag{3.39}$$

$$= f^2 - (\mathcal{A}f)^2 + \mathcal{U}E_2 \tag{3.40}$$

Then we can find a similar expression for V

$$\begin{aligned} V &= E_2 - f^2 \\ &= \mathcal{U}E_2 - (\mathcal{A}f)^2 \\ &= \mathcal{U}(V + f^2) - (\mathcal{A}f)^2 \\ &= \mathcal{U}f^2 - (\mathcal{A}f)^2 + \mathcal{U}V \end{aligned}$$

Here we have used the identity $\mathcal{A}f = f - g$. Now we define the function V_e as

$$V_e \;=\; \mathcal{U}f^2 \;-\; (\mathcal{A}f)^2 \tag{3.41}$$

and we get the following integral equation whose solution is the variance function

$$V \;=\; V_e \;+\; \mathcal{U}V \tag{3.42}$$

the previous equation is a second order integral equation of the second kind, as equation (1.27). This variance equation has a solution (that is, V function exists) if the norm of the involved operator is bounded by 1, that is $|\mathcal{U}|_1 < 1$. Then, we can impose an additional condition to the p function:

$$\int_D \frac{k^2(r,s)}{p(r,s)}\, dm(s) \quad < \quad 1 \tag{3.43}$$

the above must hold for all points r in domain D. This ensures that $|\mathcal{U}|_1 = 1$.

3.2.4 Transition probabilities proportional to the kernel.

In the previous section, it was introduced an expression for the variance when an arbitrary transition probability function was used, on an arbitrary domain D. In this subsection, we apply those result to Path-Tracing, in the context of global Illumination. The most usual transition probability function used is proportional to the kernel function K, that is $p(r,s) = cK(r,s)$, where c is a constant. The most obvious selection is to make c equal to 1. Then $p(r,s) = K(r,s)$ for all s in D, and we get

$$\int_{D'} p(r,s)\, dm'(s) \;=\; p(r,a) \;+\; \int_D k(r,s)\, dm(s) \tag{3.44}$$

Normalization of the function p, as stated in equality (3.3) is ensured because $m_k < 1$, as stated in the first chapter. Then we know that

$$\int_D k(r,s)\; dm(s) \;\leq\; m_k \;< 1 \tag{3.45}$$

for all $\mathbf{r}$ and $\mathbf{s}$ in D, and we get. The value $p(r,a)$ is determined by equality (3.3), from which we deduce

$$p(r,a) \;=\; 1 \;-\; \int_D k(r,s)\, dm(s) \tag{3.46}$$

now, the p function is fully determined. This instance of p obeys all necessary restrictions, and also (3.43), so we obtain finite variance. With this definition of p, we can find the value for V, that is, the variance function. In order to do this, we first find the value of the function $\mathcal{U}h$ for an arbitrary function h

$$\begin{aligned} \mathcal{U}h &= \int_D \frac{k^2(r,s)}{p(r,s)}\, h(s)\, dm(s) \\ &= \int_D k(r,s)\, h(s)\, dm(s) \\ &= \mathcal{A}h \end{aligned}$$

and we deduce that, for this transition function, the operators $\mathcal{U}$ and $\mathcal{A}$ are the same, then the variance is

$$V = V_e + \mathcal{A}V \tag{3.47}$$

where

$$V_e = \mathcal{A}f^2 - (\mathcal{A}f)^2 \tag{3.48}$$

the variance function V is thus integrable in D because it can be obtained as the solution to (3.47) and $|\mathcal{A}|_1 < 1$. This means that V has finite values in D except for a zero-measure set of points. Furthermore, the variance is given by

$$V = \mathcal{A}^+ V_e = \mathcal{A}^+ (\mathcal{A}f^2 - (\mathcal{A}f)^2) \tag{3.49}$$

and from (3.40) we get that $E_2 = (f^2 - (\mathcal{A}f)^2) + \mathcal{T}E_2$ and thus

$$E_2 = \mathcal{A}^+ (f^2 - (\mathcal{A}f)^2) \tag{3.50}$$

3.2.5 Estimation of inner products.

In the previous subsection, we have seen the variance involved in punctual estimation of the function f. But usually, in Global Illumination the target is the estimation of the inner product $\langle f \mid h \rangle$ of f and an arbitrary function h, as stated in chapter 2. This can be done also by using MC methods. In order to formalize this, we introduce a new random variable X_h whose mean is $\langle f \mid h \rangle$. The definition is

$$X_h(rc) = \frac{h(r)}{p_0(r)}\, X_r(c) \tag{3.51}$$

where p_0 is a probability density function, obeying

$$h(r) > 0 \;\Rightarrow\; p_0(r) > 0 \tag{3.52}$$

$$\int_D p_0(r)\, d\mu(r) = 1 \tag{3.53}$$

the function p_0 induces a probability measure U_h on C (the space of all chains), which obeys

$$dU_h(rc) = p_0(r)\, dm(r)\, dU_r(c) \tag{3.54}$$

Now, we expand the expected value of X_h with respect the probability measure U_h

$$\begin{aligned}
E(X_h) &= \int_C X_h(rc)\, dU_h(rc) && (3.55)\\
&= \int_{C\times D} h(r)\, X_r(c)\, dm(r)\, dU_r(c) && (3.56)\\
&= \int_D h(r) \left[\int_C X_r(c)\, dU_r(c)\right] dm(r) && (3.57)\\
&= \int_D h(r)\, E(X_r)\, dm(r) && (3.58)\\
&= \int_D h(r)\, L(r)\, dm(r) && (3.59)\\
&= \langle f \mid h\rangle && (3.60)
\end{aligned}$$

With this, we know that the random variable X_h can be sampled to guess a value for $\langle f \mid h\rangle$. The efficiency of the method depends on the variance. To obtain an expression for $Var(X_h)$, we begin by expanding $E(X_h^2)$

$$\begin{aligned}
E(X_h^2) &= \int_D X_h^2(rc)\, dU_h(rc) && (3.61)\\
&= \int_{C\times D} X_h^2(rc)\, p_0(r)\, dm(r)\, dU_r(c) && (3.62)\\
&= \int_D \left[\int_C X_r^2(c)\, dU_r(c)\right] \frac{h^2(r)}{p_0(r)}\, dm(r) && (3.63)\\
&= \int_D E(X_r^2)\, \frac{h^2(r)}{p_0(r)}\, dm(r) && (3.64)\\
&= \left\langle E_2 \mid h^2/p_0\right\rangle && (3.65)
\end{aligned}$$

And then, the variance is

$$\begin{aligned}
Var(X_h) &= E(X_h^2) - E^2(X_h) && (3.66)\\
&= \left\langle E_2 \mid h^2/p_0\right\rangle - \langle f \mid h\rangle^2 && (3.67)
\end{aligned}$$

The value $p_0(r)$ is the differential probability for selecting r as the first element in a chain. Once this is done, transitions are governed by the p function.

In the case that it is possible, we select p_0 proportional to h, because when no other information is available, this is an optimal selection. In this case we have $p_0 = h/c$ where c is a normalization constant, that is, $c = \int_D h(r)dm(r)$. With this example of p_0, we get the following variance:

$$\begin{aligned}
Var(X_h) &= c\left\langle \mathcal{A}^+(f^2 - (\mathcal{A}f)^2) \mid h\right\rangle - \langle f \mid h\rangle^2 && (3.68)\\
&= c\left\langle f^2 - (\mathcal{A}f)^2 \mid (\mathcal{A}^*)^+ h\right\rangle - \langle f \mid h\rangle^2 && (3.69)
\end{aligned}$$

3.2.6 An ideal transition probability with zero variance

From the above section we can deduce when the variance involved in the estimator X_r is zero. This can only happen if the V_e function is zero at all points. From (3.41) we get that $V_e = 0$ if and only if $\mathcal{U}f^2 = (\mathcal{A}f)^2$. We select the following transition probability function:

$$p(r,s) \;=\; K(r,s)\,\frac{f(s)}{\mathcal{A}f(r)} \tag{3.70}$$

it is easy to prove that the above function p obeys equations (3.3) and (3.5). With respect to the variance, we find $\mathcal{U}f^2$:

$$\begin{aligned}
\mathcal{U}f^2(r) &= \int_D \frac{K^2(r,s)}{p(r,s)}\, f^2(s)\, ds && (3.71)\\
&= \int_D K(r,s)\, \mathcal{A}f(r)\, f(s)\, ds && (3.72)\\
&= \mathcal{A}f(r) \int_D K(r,s)\, f(s)\, ds && (3.73)\\
&= (\mathcal{A}f(r))^2 && (3.74)
\end{aligned}$$

As expected, we have found that $\mathcal{U}f^2 = (\mathcal{A}f)^2$, and then $V_e = 0$ when p is as defined in (3.70). Obviously we can not select chains with the probability density induced by this p, as its definition involves the unknown values $f(s)$ and $\mathcal{A}f(r)$.

3.2.7 Applications to Global Illumination

In this subsection we see how the results in the previous subsections can be applied to analyze the variance for a number of problems in GI.

Computation of a pixel intensity

In this case, the problem is the computation of samples of the image function, as stated in chapter 2. Recall that the set of functions $\mathbf{W} = \{W_{e1}, \ldots, W_{en}\}$ is used to project the radiance function and obtain a discrete image. The intensity at each pixel is $l_i = \langle W_{ei} \mid L \rangle$. Thus, the problem can be stated as the inner product of the radiance function and other function, and path-tracing can be applied. This can be done by sampling the random variable X_h as defined in (3.51), with $h = W_{ei}$, $\mathcal{A} = \mathcal{T}$, $g = L_e$, $f = L$, $\mathcal{A}f = \mathcal{T}L = L_r$ and $m = \mu$. Lets assume that W_{ei} is normalized, in the sense that $\int_D W_{ei}(\mathbf{r})\, d\mu(\mathbf{r}) = 1$, then we can use $p_0 = W_{ei}$. We also use transition probabilities proportional to the kernel, and hence $p = K$ and $\mathcal{T} = \mathcal{U}$. Now, we have that $E(X_h) = l_i$, and the method is unbiased. With respect to variance, we get from (3.50), (note that $\mathcal{U} = \mathcal{T}$) the value of E_2

$$E_2 \;=\; \mathcal{T}^+ \,(L^2 - L_r^2) \tag{3.75}$$

by using $p_0 = h = W_{ei}$, equation (3.67) can be rewritten as

$$Var(X_h) = \langle L^2 - L_r^2 \mid (\mathcal{T}^*)^+ W_{ei} \rangle - l_i^2 \quad (3.76)$$
$$= \langle L^2 - L_r^2 \mid W_i \rangle - l_i^2 \quad (3.77)$$

note that $W_i = (\mathcal{T}^*)^+ W_{ei}$, is the potential (or importance) induced by the i-th pixel in the image, as defined in section 2.1.

Computation of the radiosity of a patch

In this case, the goal is the obtention of the radiosity of a patch, either by shooting power carrying particles from the light sources, as stated in sub section 2.5.2, either by gathering random walk, by using paths starting from the patch.

We assume that L_e has a diffuse distribution (that is, $L_e(x, w)$ is independent of w) and that $\mathcal{T}$ is the diffuse transport operator. In this circumstances, (as can be seen in sub section 1.3.2) radiance transport is modeled by the radiosity equation. The radiosity of a patch is the average value of the radiosity function for all its area. Lets call A to the patch region, and b to its radiosity, then we have

$$b = \frac{1}{\mu(A \times O)} \int_{A \times O} L(\mathbf{r})\, d\mu(\mathbf{r}) = \frac{1}{\pi |A|} \int_{A \times 0} L(\mathbf{r})\, d\mu(\mathbf{r}) \quad (3.78)$$

by defining the following W_e function

$$W_e(\mathbf{r}) = \begin{cases} \frac{1}{\pi |A|} & \text{when } \mathbf{r} \in A \times O \\ 0 & \text{otherwise} \end{cases} \quad (3.79)$$

we can write b as an inner product

$$b = \int_D W_e(\mathbf{r})\, L(\mathbf{r})\, d\mu(r) = \langle L \mid W_e \rangle \quad (3.80)$$

Let us define $W = (\mathcal{T}^*)^+ W_e$, that is, W is the potential function. We can state these two adjoint equations

$$B = B_e + \mathcal{T} B \quad (3.81)$$
$$W = W_e + \mathcal{T}^* W \quad (3.82)$$

and also express b in two ways

$$b = \langle B \mid W_e \rangle = \langle \mathcal{T}^+ B_e \mid W_e \rangle = \langle B_e \mid (\mathcal{T}^*)^+ W_e \rangle = \langle B_e \mid W \rangle \quad (3.83)$$

then there exists two different MC methods to estimate b. Both are based in the random variable X_h, as defined in subsection 3.2.5, which are described:

Particle tracing. In this case we can make $f = W$, $\mathcal{A} = \mathcal{T}^*$, and $h = B_e$. This implies that $E(X_h) = \langle B_e \mid W \rangle = b$. This is called particle tracing algorithms. The variance of particle tracing methods can be obtained by

assuming that transition probabilities are made proportional to the kernel, that is $p(s,r) = K(r,s)$. The function p_h is made also proportional to B_e, by doing $p_0(\mathbf{r}) = B_e(\mathbf{r})/\phi_e$ where ϕ_e is the total power emitted in the environment, that is, $\phi_e = \int_D B_e(\mathbf{r})\, d\mu(\mathbf{r})$. Then $h^2/p_0 = \phi_e B_e$. With these definitions, we get

$$Var(X_h) = \left\langle W^2 - W_r^2 \mid \mathcal{T}^+ B_e \right\rangle \phi_e - b^2 \quad (3.84)$$

$$= \left\langle W^2 - W_r^2 \mid B \right\rangle \phi_e - b^2 \quad (3.85)$$

where $W_r = \mathcal{T}^* W$.

Path-tracing. In this other case we make $f = L$, $\mathcal{A} = \mathcal{T}$, and $h = W_e$. In this case we get a gathering random walk process (or path-tracing) and $E(X_h) = \langle B \mid W_e \rangle = b$. The variance of gathering random walk can be obtained when $p(\mathbf{r},\mathbf{s}) = K(\mathbf{r},\mathbf{s})$ and $p_h(\mathbf{r}) = W_e(\mathbf{r})$ (note that W_e is normalized). In this conditions, we obtain

$$Var(X_h) = \left\langle \mathcal{T}^+ (B^2 - B_r^2) \mid W_e \right\rangle - b^2 \quad (3.86)$$

$$= \left\langle B^2 - B_r^2 \mid W \right\rangle - b^2 \quad (3.87)$$

3.3 Path-Tracing with direct light source sampling.

As was explained in the survey about Global Illumination methods, direct light source sampling is an enhancement to raw Monte-Carlo Path Tracing. In this case, instead of adding the emitted radiance, the reflected radiance (at each bounce) is sampled, also by using Monte-Carlo techniques. This was proposed in [Kajiya86]. After this, several sampling distributions have been proposed with this purpose ([Shirley96]). In this section we try to find an expression for the variance when this method is used. The first step is to introduce the density of Markov-Chains, then we show that the method is unbiased, and we find its variance. Finally, we show how this method is applied in Global Illumination.

3.3.1 Extended density of Markov Chains.

Now, we introduce a probability measure for Markov-Chains which is involved in direct source sampling. This density is very similar to that used in section 3.2.1, although this is defined in an extended space. Lets consider the space C' of all infinite chains of pairs of elements in D', that is, if $c \in C'$ then $c = (r_1, r_2, \ldots)$ where $r_i \in (D \bigcup \{a\})^2$. We will also use the notation rc, where $r \in D'^2$ and $c \in C'$, thus if $c = (r_1, r_2, \ldots)$ then $rc = (r, r_1, r_2, \ldots) \in C'$. Each state s in a chain is a pair whose two elements are called s_d and s_i, respectively, that is $s = (s_d, s_i)$. The element s_d is used for direct source sampling, while s_i is the next element in the chain. Assume that current state in a chain is $q = (r_d, r)$. The probability for the next state to be $s = (s_d, s_i)$ is depends of r and is independent of r_d. Furthermore,

this probability can be decomposed as a product of the two components in s (it is separable with respect to s). Thus, we can write this probability as

$$p_d(r, s_d)\, p(r, s_i)\, dm(s_d)\, dm(s_i)$$

where $p_d(r, s_d)$ is the probability of selecting p_d for direct source sampling, and $p(r, s_i)$ is the probability of continuing the chain through s_i. Figure 3.2 shows an example of an extended chain. It starts in r (is the ray pointing to the observer), and then continues with (s_d, s_i). The origin of s_d usually is in a light source.

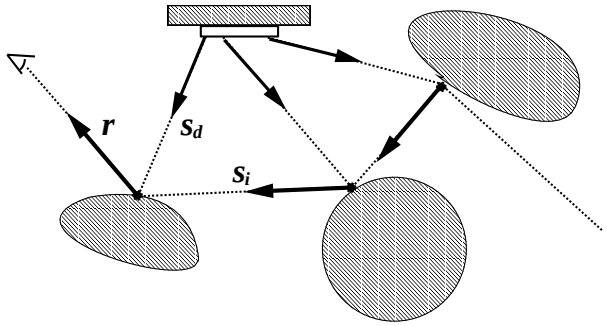

Figure 3.2: An example of an extended chain.

With all this, we can define the probability measure Q_r, for every $r \in D'$, is defined by the following equality

$$dQ_r(sc) \;=\; p_d(r, s_d)\, dm(s_d)\, p(r, s_i)\, dm(s_i)\, dQ_{s_i}(c) \tag{3.88}$$

The value $p_d(r, s_d)\, dm(s_d)$ is the (differential) probability for selecting s_d for direct source sampling, while $p(r, s_i)\; dQ_{s_i}(c)$ is the differential probability for making a transition to state s_i and then going through all states in chain c The function p here is also the transition probability function, and obeys properties defined in (3.3) and (3.4). The function p_d is used to make direct source sampling, and obeys

$$1 \;=\; \int_D p_n(r, s)\; dm(s) \tag{3.89}$$

$$k(r, s)g(s) \;>\; 0 \;\;\Rightarrow\;\; p_d(r, s) \;>\; 0 \tag{3.90}$$

Extended chains used in this section are supposed to follow the distribution induced by the probability measure Q_r.

3.3.2 A random variable on extended chains.

Now we define the random variable Y_r, for all elements $r \in D$, and evaluated on extended chains

$$Y_r(sc) \;=\; Z_r(s_d) \;+\; \frac{k(r, s_i)}{p(r, s_i)}\, Y_{s_i}(c) \tag{3.91}$$

where Z_r is another random variable defined as

$$Z_r(s_d) \;=\; \frac{k(r,s_d)}{p_d(r,s_d)}\, g(s_d) \tag{3.92}$$

Note that Z_r is defined on D instead of C'. Its mean with respect the density $p_d(r,s)\; dm(s_d)$ is

$$\begin{aligned} E(Z_r) &= \int_D Z_r(s_d)\, p_d(s,d_m)\, dm(s_d) && (3.93)\\ &= \int_D k(r,s_d)\, g(s_d)\, dm(s_d) && (3.94)\\ &= (\mathcal{A}g)(r) && (3.95) \end{aligned}$$

The mean of Y_r with respect to measure Q_r is

$$\begin{aligned} E(Y_r) &= \int_{C'} Y_r(sc)\, dQ_r(sc) && (3.96)\\ &= \int_{C'} Z_r(s_d)\, dQ_r(sc) \;+\; \int_{C'} \frac{k(r,s_i)}{p(r,s_i)}\, Y_{s_i}(c)\; dQ_r(sc) && (3.97) \end{aligned}$$

the first term above can be written as

$$\begin{aligned} & \int_{C'} Z_r(s_d)\, dQ_r(sc) \\ = \;& \int_{D^2\times C'} Z_r(s_d)\, p_d(r,s_d)\, dm(s_d)\, p(r,s_i)\, dm(s_i)\, dQ_{s_i}(c) \\ = \;& \int_D Z_r(s_d) \left(\int_D p(r,s_i) \left[\int_{C'} dQ_{s_i}(c) \right] dm(s_i) \right) p_d(r,s_d) dm(s_d) \\ = \;& \int_D Z_r(s_d)\, p_d(r,s_d)\, dm(s_d) \\ = \;& E(Z_r) \\ = \;& (\mathcal{A}g)(r) \end{aligned}$$

while the second term in (3.97) is

$$\begin{aligned} & \int_{C'} \frac{k(r,s_i)}{p(r,s_i)}\, Y_{s_i}(c)\, dQ_r(sc) \\ = \;& \int_{D^2\times C'} k(r,s_i)\, Y_{s_i}(c)\, p_d(r,s_d)\, dm(s_d)\, dm(s_i)\, dQ_{s_i}(c) \\ = \;& \int_D k(r,s_i) \left[\int_{C'} Y_{s_i}(c) \left(\int_{D'} p(r,s_d)\, dm(s_d) \right) dQ_{s_i}(c) \right] dm(s_i) \\ = \;& \int_D k(r,s_i) \left[\int_{C'} Y_{s_i}(c) dQ_{s_i}(c) \right] dm(s_i) \\ = \;& \int_D k(r,s_i)\, E(Y_{s_i})\, dm(s_i) \end{aligned}$$

$$\begin{aligned} &= \int_D k(r,s_i)\, E_1(s_i)\, dm(s_i) \\ &= (\mathcal{A}E_1)(r) \end{aligned}$$

here, we have defined the function $E_1(r) = E(Y_r)$, as in previous section. Joining again the two terms in (3.97) we get

$$E(X_r) \;=\; E_1(r) \;=\; (\mathcal{A}g)(r) \;+\; (\mathcal{A}E_1)(r) \tag{3.98}$$

or, in a functional form

$$E_1' \;=\; \mathcal{A}g \;+\; \mathcal{A}E_1' \tag{3.99}$$

and thus, we deduce

$$E_1' \;=\; \mathcal{A}^+(\mathcal{A}g) \;=\; \mathcal{A}(\mathcal{A}^+ g) \;=\; \mathcal{A}f \tag{3.100}$$

we have used the identity $\mathcal{A}^+\mathcal{A} = \mathcal{A}\mathcal{A}^+$, which is easily derived from the definition of $\mathcal{A}^+$. Then we have that $E(Y_r) = (\mathcal{A}f)(r)$ and we can use this random variable to guess that value. The target is to compute the values of f instead of $\mathcal{A}f$, and this is done by sampling Y_r' defined as

$$Y_r'(c) \;=\; g(r) \;+\; Y_r(c) \tag{3.101}$$

and obviously

$$E(Y_r') \;=\; g(r) + E(Y_r) \;=\; g(r) + (\mathcal{A}f)(r) \;=\; f(r) \tag{3.102}$$

obviously, it holds $Var(Y_r) = Var(Y_r')$. In the next section, this variance is analyzed, as it was done with the variance of X_r.

The variance of Y_r

With respect to variance, we have $Var(Y_r) = Var(Y_r')$, so we analyze the first of them. As in previous section, we define $E_2'(r) = E(Y_r^2)$. This value can be expanded as the sum of three terms. Each of them can be simplified by using the same steps involved in the derivation of the mean value.

$$\begin{aligned} E_2(r) &= E(Y_r^2) \\ &= \int_{C'} Y_r^2(sc)\, dQ_r(sc) \\ &= \int_D Z_r^2(s_d)\, p_d(r,s_d)\, dm(s_d) \;+\; \int_D \frac{k^2(r,s_i)}{p(r,s_i)}\, E(Y_{s_i}^2)\, dm(s_i) \\ &\quad + 2\int_D Z_r(s_d)\, \frac{k(r,s_i)}{p(r,s_i)} Y_{s_i}(c)\, dQ_r(sc) \end{aligned}$$

The first two terms above can be written as $E(Z_r^2) + (\mathcal{U}E_2')(r)$. The third term can be expanded as

$$2\int_D Z_r(s_d)\, \frac{k(r,s_i)}{p(r,s_i)} Y_{s_i}(c)\, dQ_r(sc)$$

$$
\begin{aligned}
&= \left[\int_D Z_r(s_d)\, p_d(r,s_d)\, dm(s_d)\right] \left[\int_D k(r,s_i) \left[\int_{C'} Y_(s_i)(c)\, dQ_{s_i}(c)\right] dm(s_i)\right] \\
&= 2\, E(Z_r)\, (\mathcal{A}E_1')(r) \\
&= 2\, (\mathcal{A}g)(r)\, (\mathcal{A}^2 f)(r)
\end{aligned}
$$

Joining again the three terms, we can rewrite the expression of E_2' as

$$E_2'(r) \;=\; E(Z_r^2) \;+\; (\mathcal{U}E_2')(r) \;+\; 2\,(\mathcal{A}g)(r)\,(\mathcal{A}^2 f)(r) \tag{3.103}$$

the value $E(Z_r^2)$ can be written as $(\mathcal{F}g^2)(r)$ where $\mathcal{F}$ is the integral operator with kernel k^2/p_d, that is

$$(\mathcal{F}g^2)(r) \;=\; \int_D \frac{k^2(r,s_d)}{p_d(r,s_d)}\, g^2(s_d)\, dm(s_d) \tag{3.104}$$

For simplicity, we define $f_n = \mathcal{A}^n f$ and $g_n = \mathcal{A}^n g$, for all $n \geq 0$. Applying $\mathcal{A}^n$ on both sides of (1.44), we get $f_n = g_n + f_{n+1}$. As $g_n \geq 0$, then we get $f_{n+1} \leq f_n$. With all these definitions, equation (3.103) can be written as

$$
\begin{aligned}
E_2' &= \mathcal{F}g^2 \;+\; 2\,(\mathcal{A}g)(\mathcal{A}^2 f) \;+\; \mathcal{U}E_2' \\
&= \mathcal{F}g^2 \;+\; 2\, g_1\, f_2 \;+\; \mathcal{U}E_2' \\
&= \mathcal{F}g^2 \;+\; f_1^2 \;-\; g_1^2 \;-\; f_2^2 \;+\; \mathcal{U}E_2'
\end{aligned}
$$

The previous equation is an integral equation, because E_2' is in both sides of the equality. If we assume that $\mathcal{U}$ obeys inequality (3.43), then $|\mathcal{U}|_1$, and then exists $\mathcal{U}^+$. By using this operator, we can write E_2' as:

$$E_2' \;=\; \mathcal{U}^+ \left[\, \mathcal{F}g^2 \;-\; g_1^2 \;+\; f_1^2 \;-\; f_2^2 \,\right] \tag{3.105}$$

If we define the function V' as $V'(r) = Var(Y_r)$ then we have $E_2'(r) = V'(r) + E^2(Y_r) = V'(r) + (f_1(r))^2$. If we substitute in equation (3.103) the function E_2' by this value, then we obtain the following expression for V'

$$
\begin{aligned}
V' &= \mathcal{F}g^2 \;-\; g_1^2 \;-\; f_2^2 \;+\; \mathcal{U}(V' + f_1^2) \\
&= \mathcal{F}g^2 \;-\; g_1^2 \;-\; f_2^2 \;+\; \mathcal{U}f_1^2 \;+\; \mathcal{U}V'
\end{aligned}
$$

this is equivalent to

$$V' \;=\; \mathcal{U}^+ \left[\, \mathcal{F}g^2 \;-\; g_1^2 \;-\; f_2^2 \;+\; \mathcal{U}f_1^2 \,\right] \tag{3.106}$$

So we see that random variable Y_r has a variance V' which is the solution of an integral equation of the second kind.

In the case that the transition probability function p is proportional to the kernel k then (as we saw in previous section) it holds $\mathcal{A} = \mathcal{U}$, and then we can write E_2' and V' as:

$$E_2' \;=\; \mathcal{A}^+ \left[\, \mathcal{F}g^2 \;-\; g_1^2 \;+\; f_1^2 \;-\; f_2^2 \,\right] \tag{3.107}$$

$$V' \;=\; \mathcal{A}^+ \left[\, \mathcal{F}g^2 \;-\; g_1^2 \;-\; f_2^2 \;+\; \mathcal{A}f_1^2 \,\right] \tag{3.108}$$

3.3.3 Estimation of inner products.

As has been stated previously, frequently we want to compute inner products of f and an arbitrary function h. This can be done by using extended chains, by using an algorithm similar to the one used for simple chains. In what follows, we define a probability measure and a random variable used to do this.

With respect to the probability measure, it will be named as Q_h and is defined in the space $D \times C'$. This means that the sample space is an element in D and a chain. For simplicity, we will write rc to mean the element (r, c) in $D \times C'$, where r is an element in D and c is an extended chain in C'. We will define the new measure by using its relation to Q_r, as follows:

$$dQ_h(rc) \;=\; p_0(r)\, dm(r)\, dQ_r(c) \tag{3.109}$$

where p_0 obeys the properties established in (3.52) and (3.53). The value $p_0(r)$ is the probability for selecting r as the first ray in the extended chain. The random variable Y_h is defined also in $D \times C'$, as follows:

$$Y_h(rc) \;=\; \frac{h(r)}{p_0(r)}\, Y'_r(c) \tag{3.110}$$

the mean value of Y_h with respect to Q_h is obtained as:

$$\begin{aligned}
E(Y_h) &= \int_{D\times C'} Y_h(rc)\, dQ_h(rc) \\
&= \int_D h(r) \left[\int_{C'} Y'_r(c)\, dQ_r(c)\right] dm(r) \\
&= \int_D h(r)\, E(Y'_r)\, dm(r) \\
&= \int_D h(r)\, f(r)\, dm(r) \\
&= \langle h \mid f \rangle
\end{aligned}$$

then we see that sampling Y_h is useful for obtaining unbiased estimators of $\langle f \mid h \rangle$. With respect to variance, we know that $Var(Y_h) = E(Y_h^2) - \langle f \mid h \rangle^2$. The, if we expand $E(Y_h^2)$, we obtain:

$$\begin{aligned}
E(Y_h^2) &= \int_{D\times C} Y_h^2(rc)\, dQ_h(rc) \\
&= \int_{D\times C} \frac{h^2(r)}{p_0(r)}\, Y_r'^2(c)\, dQ_h(rc) \\
&= \int_D \frac{h^2(r)}{p_0(r)} \left[\int_C Y_r'^2(c)\, dQ_r(c)\right] dm(r) \\
&= \int_D \frac{h^2(r)}{p_0(r)}\, E(Y_r'^2)\, dm(r) \\
&= \langle h^2/p_0 \mid E'_2 + f^2 - f_1^2 \rangle
\end{aligned}$$

where we have used the equality $E(Y_r'^2) = E_2'(r) + f^2 - f_1^2$, which is obtained easily from the definition of Y_r', given in (3.101).

If we assume $\int_D h(r)\, dm(r) = 1$ (that is, the function h is normalized), that p_0 is equal to h at all points, and we select p proportional to k then we get $\mathcal{A} = \mathcal{U}$ and $h^2/p_0 = h$. For this particular conditions the value $E(Y_h^2)$ can be written as follows:

$$\begin{aligned} E(Y_h^2) &= \langle h \mid E_2' + f^2 - f_1^2 \rangle \\ &= \langle h \mid E_2' \rangle + \langle h \mid f^2 - f_1^2 \rangle \\ &= \langle (\mathcal{A}^+)^*\, h \mid \mathcal{F}g^2 - g_1^2 + f_1^2 - f_2^2 \rangle + \langle h \mid f^2 - f_1^2 \rangle \\ &= \langle l \mid \mathcal{F}g^2 - g_1^2 + f_1^2 - f_2^2 \rangle + \langle h \mid f^2 - f_1^2 \rangle \end{aligned}$$

where $l = (\mathcal{A}^*)^+\, h$ is the solution to the equation $l = h + \mathcal{A}^* l$, which is the adjoint of equation (1.44).

3.4 Conclusions.

In chapter three we focus on existing Monte-Carlo methods for second order integral equations. We present a probability measure on the space of all infinite chains, and a random variable defined over this probability measure. Then, some Monte-Carlo methods can be viewed as the obtention of samples from that random variable.

By using those tools, we obtain a characterization of the variance, by using a general formulation. A interesting result is that the variance function obeys a second order integral equation. Finally, we apply that result to path tracing for radiance, particle tracing for radiosity, and gathering random walk for radiosity. In all the cases we obtain expressions for the variance, and those expressions involve just the potential function and the unknown radiance function.

Related characterizations of the variance where previously known [Mikhailov92], but did not included finite chains ended in absorption. We also introduce ideal estimators with zero variance.

The main benefit is in a higher comprehension of the nature of the variance function. This makes easier the design of better algorithms with reduced variance.

Chapter 4

Improved Final-Gather for Radiosity.

4.1 Introduction.

The computation of radiosity on arbitrary environments is a problem which has been solved by either projection or Monte-Carlo methods, as has been explained in the second chapter. The former algorithms can be improved if hierarchical and/or higher order basis functions sets are used [Gortler93], instead of constant non-overlapping basis. The main drawback is that the process yields a projected (discrete) approximation to the radiosity function, which may not capture some details of the exact function. The process is summarized in figure 4.1: A simulation of energy transfers between pairs of elements yields a vector of radiosities B_i, each of them is related to each of the elements.

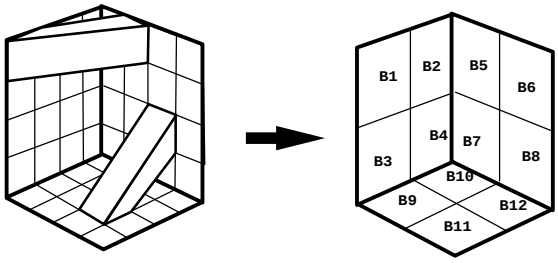

Figure 4.1: Classic finite elements radiosity.

This can be solved by using a basis function set which is known to approximate more accurately the target radiosity function. A proposed solution is to use discontinuity meshing [Lischinski92], in which the domain of each basis function extends to an area which does not includes any discontinuity up to some given order. Another option is the *shadow masks* technique [Zatz93], for higher order basis

functions. Unfortunately, the computation of either the discontinuity mesh or the shadow masks is complicated for environments with a large number of surfaces.

With respect to Monte-Carlo methods, they rely on sampling a distributions of Markov Chains [Spanier69, Kajiya86]. The computation process involves some given variance for the estimators used. In all the cases, a huge number of chains is needed to reduce this variance to acceptable levels.

In order to overcome these limitations, we consider two pass methods, in which the first pass yields a low resolution approximation to the radiosity function, and in the second pass (also called *rendering pass*) this approximation is refined by using final gather at the pixel level. The process of final gather can be stated as the computation of irradiance in each given visible point during rendering. This implies gathering radiosity values from all the possible sources in the environment (see figure 4.2)

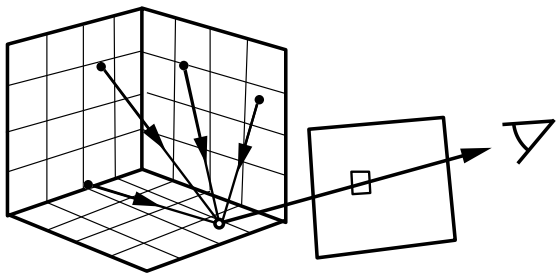

Figure 4.2: The final gather step.

Several papers deal with this problem. In [Rushmeier88]. a Monte-Carlo method for final gather was presented, and in [Reichert92] it was described a system in which hemicubes and graphics hardware was used. Both methods require a large amount of computation at each pixel.

An improvement to the raw scheme comes from sampling just for direct lighting as described in [Shirley90], and use the stored radiosity for indirect illumination. In order to further improve the sampling method, it is possible to guess which sources or reflectors contribute the most to the irradiance at the target point. Then a bigger effort can be made for those selected areas. In [Kok91], a method is shown in which the first pass yields, for each patch, a set of other patches that have been classified as sources for the first. At the rendering pass, those other patches are directly sampled by adaptive ray-casting. Ward describes [Ward91] a method to avoid sampling all light sources, for scenes with many of them. An spatial index can be used to compute the area of influence of sources, as described in [Zimmerman95, Shirley96]. Others approaches to compute irradiance include the use of Evolutionary Algorithms [Beyer94], or simplified geometry for the source function [Rushmeier93, Zimmerman95].

In this chapter we propose a simple yet efficient Monte-Carlo method to compute irradiance over target points in the second pass. We use information gathered at the first pass, to guess the distribution of irradiance over each patch, with respect to

the source patch. We show how this data can be used to built a valid *pdf*, yielding an unbiased estimator of irradiance with low variance. The method does not need complicated data structures, neither modifications to standard techniques for low resolution radiosity computations. The number of samples that each patch receives is (approximately) proportional to the irradiance it induces on the patch where the receiver point belongs. This number of samples can be reduced, and in fact, it can be lower than the number of patches. Thus, by using a simple and well defined *pdf*, we avoid the explicit sampling of every patch.

In the following sections, we introduce the notation and then we give the details about various sampling distributions for final gather, including the weighed distributions that use information from the first pass. Finally, we give sample images and numerical comparisons which show the performance of this method as compared with other standard method.

4.2 Monte-Carlo Final Gather.

The methods which use final gather to refine a low resolution radiosity function can be expressed as a two-step computation, by using the notation introduced in chapter 2.

$$\begin{array}{lrcl} (1) & B' & = & (\mathcal{P}_{\mathbf{B}}\,\mathcal{D})^{+}\,B_e \\ (2) & I & = & \mathcal{P}_{\mathbf{W}}\,(\mathcal{D}\,B' + B_e) \end{array} \tag{4.1}$$

The first step is a computation of a projected version of the radiosity function B'. This is done by using a basis $\mathbf{B}$, which yields a low resolution result. At the second step, for each pixel, we compute the first visible point from that pixel, then we perform final gather over x to guess the value $E_g(x)$ (which is an approximation to the irradiance), and add the emitted radiosity. Thus, the radiosity is obtained as

$$B(x) \;=\; B_e(x) \;+\; \frac{\rho(x)}{\pi}\,E_g(x) \tag{4.2}$$

The problem of final gather over points can be stated as follows. We are given B', which is our current approximation to B, the emitted radiosity function B_e and a target point x. With this data, we have to do a final gather over x to get an estimate of irradiance . Note that the exact result of the final gather is different from the exact irradiance on x. This is due to the usage of B' instead of B as the source term. In fact, we define the exact result of the final gather over x as $E_g(x)$. This value is given by the following expression:

$$E_g(x) \;=\; \int_S G(x,y)\,B'(y)\,dA(y) \;=\; \int_S T(x,y)\,dA(y) \tag{4.3}$$

where $G(x,y)$ is as defined in (1.32), and we have defined, for simplicity, $T(x,y) = G(x,y)B'(y)$. Although we would like to obtain E_g, this is not affordable because the previous integration cannot be done in short time for real environments. As usually, we can resort to MC sampling to achieve this goal. The result we get is an approximation (under a given variance) to E_g, which we call E'_g.

4.3 Sampling distributions for Final gather.

In order to approximate the integral (4.3) by using Monte Carlo techniques, we have to select a valid probability distribution function [1] p_x over the domain of integration, S, whose integral is one, that is:

$$\int_S p_x(y)\, dy \;=\; 1 \tag{4.4}$$

and assigns non-zero probability to any point y with a non-zero contribution to the integral, that is, for all points x in S and direction $w \in \Omega$, it holds that:

$$T(x,y) > 0 \quad \Longrightarrow \quad p_x(y) > 0 \tag{4.5}$$

Any function meeting both previous restrictions is valid for our purposes. Once p_x has been selected, we can sample surface points, and get an approximation to $E_g(x)$. This is done by selecting a number of samples n, and then n points on S with distribution p_x (these are named y_i). Then, as stated by Monte-Carlo integration rules, the approximated value can be computed by:

$$E'_g(x) \;=\; \sum_{i=0}^{n-1} \frac{T(x,y_i)}{P_x(y_i)} \tag{4.6}$$

The set of possible values we obtain for $E'_g(x)$ have a distribution whose mean is exactly $E_g(x)$, when the points y_i are distributed according to $P_x(y)$.

The problem stated above is closely related to the problem of direct light source sampling. In final gather, every other object must be sampled, while in distributed ray-tracing algorithms, just the light sources are sampled. Except for this, the two problems are similar, as the irradiance is gathered from the sampled points to the target point. Numerous papers in the context of realistic rendering are devoted to the problem of finding an efficient function $P_x(y)$, and the importance of this is stated in [Shirley96]. The basic goal is that the amount of samples taken for each source [2] is as proportional as possible to the energy that the source radiates toward the target point. But in fact this implies some knowledge about the irradiance at the target, which is an unknown value.

4.3.1 Variance of the distributions.

As stated, any P_x is valid, but the efficiency is not equal for all of them, because each p_x induces a different variance in the estimation. The goal is then to select a sampling distribution with low variance, using known data. The variance is called $V(E_g(x))$, although we simply write $E(x)$ in the context of this chapter. This value

[1] this function has as subscript the target point where we want to perform final gather, because the definition of p_x usually includes x as a parameter.

[2] In our implementation we use the word *source* in the sense of objects which radiate toward the target point x, not just objects with self emitted radiant energy

is the mean of the squared distances from the samples to the exact mean value. It can be expressed as:

$$V(x) \;=\; \frac{1}{n}\left(\int_S \frac{T^2(x,y)}{P_x(y)}\,dy \;-\; E_g^2(x)\right) \tag{4.7}$$

If we use $P_x(y) \;=\; T(x,y)/E_g(x)$, then the above term is just zero. This is the ideal sampling strategy, but it is impossible to implement because the definition of p_x involves the target (unknown) value $E_g(x)$. Our goal is to use a function P_x whose shape is as similar as possible to the above, in order to reduce variance.

4.3.2 Solid angle sampling.

An option for selecting P_x is to assign a probability to any y which is proportional to the solid angle subtended by y when projected over x. This is usually called hemisphere sampling. Here p_x is expressed as follows:

$$P_x(y) \;=\; \frac{1}{2\pi}\,\frac{cos(n_y, w_{yx})}{|x-y|^2}\,V_{xy} \tag{4.8}$$

note that the factor $1/2\pi$ is introduced in order to meet condition (4.4), as 2π is the surface of a hemisphere with unit radius. In figure 4.3 it can be seen the shape of this PDF

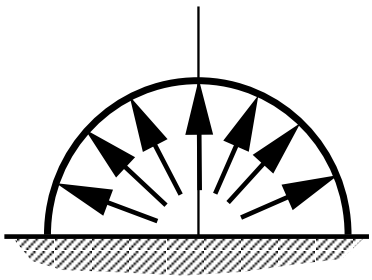

Figure 4.3: Shape of PDF for solid angle sampling.

The variance induced by the previous distribution is obtained by substituting (4.8) into (4.7)

$$V_1(x) \;=\; \frac{1}{n}\left(2\int_S cos(n_x, w_{xy})\,G(x,y)\,B'^2(y)\,dy \;-\; E_g^2(x)\right) \tag{4.9}$$

This expression of the variance cannot be further simplified, as we cannot make any assumptions about the values of B'^2_p. All we can do is to make some assumption about the scene, and then compare the variances when that assumption holds. In our case, we assume that the radiosity function B'_p is constant for all the points y at the environment. Let's call b to that constant value. To analyze the variances

we can see how the differential area element dy at y is related to a differential area element dp at a point p on the unit radius circle under the hemisphere centered at x. This relation is obtained by projecting the point y onto the unit hemisphere, and then projecting the resulting point again onto the unit circle. The relation allows to change the variable of integration. Instead of integration over S, we integrate over points in the circle, C. The differential elements are related, as $dp = \pi G(x,y)dy$, then we obtain:

$$2\int_S cos(x,y)\,G(x,y)\,B'^2_p(y)\,dy \;=\; \frac{2\,b^2}{\pi}\int_C cos(p)\,dp \;=\; \frac{2\,b^2}{\pi}\left(\frac{2}{3}\pi\right) \;=\; \frac{4}{3}b^2 \tag{4.10}$$

where $cos(p) = (1 - p_x^2 - p_y^2)^{1/2}$. The irradiance at x is also obtained by using a change of variable of integration:

$$E_g(x) \;=\; \int_S G(x,y)\,B'_p(y)\,dy \;=\; b\int_S G(x,y)dy \;=\; \frac{b}{\pi}\int_C dp \;=\; \frac{b}{\pi}\,\pi \;=\; b \tag{4.11}$$

With all these, we can simplify (4.7), and we get:

$$V_1(x) \;=\; \frac{b^2}{3n} \tag{4.12}$$

To obtain a point y with the above distribution, we select a random unit length vector w uniformly distributed in the hemisphere, then we obtain y as the first visible point from x in the direction of w, by ray-casting.

4.3.3 Projected solid angle sampling.

We know that the contribution of directional irradiance from direction w to the radiant irradiance at a point x is weighed by the cosine of the angle formed by normal at x and w. This implies that the contribution from any direction is weighed by this cosine term. Then, a better selection is to sample taking into account this cosine weight. Directions near the normal to the surface are sampled with more probability that directions away from it. The expression for $P_x(y)$ is now:

$$P_x(y) \;=\; G(x,y) \tag{4.13}$$

figure 4.4 shows the shape of this PDF. Note that direction away from the normal have a smaller chance to be selected.

It's easy to show that this distribution is normalized. Again, the variance is obtained by substituting (4.13) into (4.7)

$$V_2(x) \;=\; \frac{1}{n}\left(\int_S G(x,y)\,B'^2_p(y)\,dy \;-\; E_g^2(x)\right) \tag{4.14}$$

Now, we can made the same assumption we used for hemisphere sampling. If we suppose that B'_p is constant and equal to b at all points, then this sampling method has zero variance. In fact, in this case, the probability distribution function $P_x(y)$ is

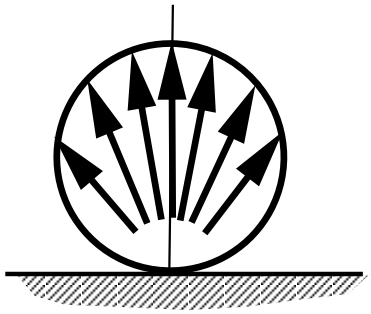

Figure 4.4: Shape of PDF for projected solid angle sampling.

proportional to the target unknown function $T(x, y)$, so cosine weighed sampling is the perfect ideal sampling method, in the sense described in section 4.3, and under the assumption of constant radiosity.

For selecting points y_i with this distribution, we first select a random point uniformly distributed in the unit circle. Then we find the point on the hemisphere which is above the first in the direction of Y axis. This point is interpreted as a ray direction. By using ray-casting we select the first visible point in that direction from x, and that point becomes the selected y.

4.3.4 Area sampling.

We can assign a constant probability density to every point of the surfaces. This distribution is independent of x, and is easy to select any point. Now, we have that:

$$P_x(y) \;=\; \frac{1}{|S|} \tag{4.15}$$

The variance of this distribution is obtained by substituting P_x in (4.7), which gives:

$$V(x) \;=\; \frac{|S|}{n} \left(\int_S G^2(x,y)\, B_p'^2(y)\, dy \;-\; E_g^2(x) \right) \tag{4.16}$$

The algorithm for selecting points with this distribution has two steps: first, we must select an object or surface in the scene. The probability for each of them is proportional to its area. After this, a point in the surface of the object is selected. The actual algorithm to do this depends of the geometry of the object. We must ensure that regions with equal areas have equal probability of including the selected point. For simple objects such as planes, spheres, cones and cylinders this is simple, while for others (as spline surfaces) this is rather complicated.

4.3.5 Parametric area sampling.

For some object shapes, area sampling becomes difficult. When the objects in the scene are composed of parametric surfaces, we can select the sample point in

parametric space, and then find the surface point y by using the image through the parametrization function of the original sample in 2D. We assume that the scene is built up of m surfaces $S_1 \dots S_m$, each of the with area $|S_i|$, and parametrization function f_i. Let u be a point in $[0,1]^2$ (the unit plane in 2D). Then $f_i(u)$ is a point in surface number i. Lets write A_i for area measure on surface number i, and A for the standard area measure on $[0,1]^2$. The function J_i, which depends on f_i, is defined as:

$$J_i(y) \;=\; \frac{dA_i(y)}{dA(u,v)} \;=\; \left| \frac{df_i(u,v)}{du} \times \frac{df_i(u,v)}{dv} \right| \tag{4.17}$$

that is, $J_i(u)$ is the relation between differential areas around point (u,v) and its image y, and is obtained as the module of the cross-product of tangent vectors at y. Once J_i is defined, we can define $p_x(y)$ as follows:

$$p_x(y) \;=\; \frac{|S_i|}{|S|} \frac{1}{J_i(y)} \tag{4.18}$$

here, i is the index of the surface where y is included. We have assumed that each surface is selected with a probability proportional to its area, as in area sampling. The point u is selected uniformly in $[0,1]^2$.

4.4 Probability Distribution Functions with reuse of information.

It is possible to interleave the computation of E'_g with the collection of information about the distribution of radiant irradiance. This information can be used during final gather to obtain probability distributions with lower variance. In this section, we see how we can easily extend photosimulation in order to obtain approximation of the average irradiance landing on a path i coming from another patch j. Then, we define probability distributions which are parametrized with these values.

4.4.1 Approximations to the distribution of irradiance.

The distribution of radiant irradiance over any point is exactly described by the function $T(x,y)$, which gives the amount of radiance arriving at x from y. Obviously, this function can not be known exactly. But we can extend the process of computing B' to also approximate the above function. We have to resort to some kind of projection on any basis function set. For simplicity, we use a partition of S in a set of m disjoint areas, which are called A_i. We assume that each A_i is planar, with area $|A_i|$. We also assume that I_p is projected by using the above set of areas or patches. The problem now is how to modify the method used to compute I'_p in order to get also an approximated, projected version of $T(x,y)$. This can be done in the case of using progressive radiosity and also with Monte-Carlo photosimulation. The algorithm starts with an array of values r_{ij} initialized to zero.

- In the case of progressive radiosity, at each shot from patch A_j to the patch A_i, we add the transferred radiosity to the current value of b_i. We can also add this energy to r_{ij}.
- In the case of photosimulation, every time a particle leaves patch A_j and hits patch A_i we will add its current weigh to r_{ij} (even in the case the particle is absorbed at A_i)

As can be seen, in any case we obtain an array of values. Each r_{ij} is an approximated value of the average irradiance over A_i due to radiosity coming from A_j. Computing these values does not increments very much computing time. Unfortunately, the storage required for this is proportional to the squared number of patches. But we do not need a huge number of patches, as they are not directly used in the obtained image.

The exact irradiance at a point x due to energy coming from patch number j is given by the function E_j, defined as:

$$E_j(x) \;=\; \int_{A_j} G(x,y)\, B'_p(y)\, dy \tag{4.19}$$

obviously, it holds that $E_g(x) = \sum_j E_j(x)$.

We also define the approximated average irradiance on patch i as $r_i = \sum_{j=1}^{m} r_{ij}$. The difference between the exact and the approximate radiance is an error function Δ_j defined as:

$$\Delta_j(x) \;=\; r_{ij} \;-\; E_j(x) \tag{4.20}$$

where i is the index of the patch which contains x. The error in the total average irradiance is Δ, whose expression is:

$$\Delta(x) \;=\; r_i \;-\; E_g(x) \;=\; \sum_{j=1}^{m} \Delta_j(x) \tag{4.21}$$

4.5 Weighed probability distributions.

Once the values r_{ij} are obtained in the first step, we can use then during final gather. The goal is to use a probability distribution whose shape is as similar as possible to the actual target function $T(x,y)$, as explained in section 4.3.

Assume that the target point is x, which is placed in patch A_i. Then we can assign a probability distribution such that the probability to select a point y on path A_j is proportional to r_{ij}. This implies that P_x holds the following restriction

$$\int_{A_j} P_x(y)\, dy \;=\; \frac{r_{ij}}{r_i} \;=\; \frac{E_j(x) + \Delta_j(x)}{I_g(x) + \Delta(x)} \tag{4.22}$$

The above equation does not fully characterizes P_x, because it just fixes its integral at each patch, not its values at each point. In fact, we can define a probability distribution function Q_{jx} which gives the distribution of the samples in the interior

of patch j when the target point is x. This *pdf* is defined on patch A_j and is normalized, that is:

$$\int_{A_j} Q_{xj}(y)\, dy \;=\; 1 \tag{4.23}$$

Then, the function P is defined as follows:

$$P_x(y) = \frac{r_{ij}}{r_i}\, Q_{xj}(y) \tag{4.24}$$

where i is the patch of x and j is the patch of y. It is easy to show that the function P obeys the required properties. In order to select a point y with this distribution, we first select the patch to sample by using a discrete pdf, in such a way that patch i is selected with probability r_{ij}/r_j. After this, the actual point y in the interior of patch is selected with distribution Q_{xj}. Thus, we can view $Q_{xj}(y)$ as the conditional pdf for selecting y knowing that we have previously selected patch j.

We can use various distributions for the probability at the interior of the patches. In the following subsections, we examine some of them.

4.5.1 Weighted sampling of area and parametric area.

The simplest selection is to choose a constant value for q_{xj} in the interior of patches j. All the points in a patch have the same chance for being selected. Then, q_{xj} is fully characterized by equation (4.23), and we obtain:

$$Q_{xj}(y) \;=\; \frac{1}{|A_j|} \tag{4.25}$$

This implies that regions with equal areas are selected with equal probability. For surfaces with simple geometry, area sampling is easy. In the case that area sampling is difficult to achieve, we can use parametric area sampling. We will assume that the patch number j is the image under a function f of a region $H_j \in [0,1]^2$, with area $A(H_j)$. Then, the expression for Q_{xj} is as follows:

$$Q_{xj}(y) \;=\; \frac{1}{A(H_j)}\, \frac{dA(u,v)}{dA_j(y)} \tag{4.26}$$

where $y = f(u_y)$ and A_j is the standard area measure on surface number j. In order to select a point y with this distribution, we first select a point (u,v) uniformly distributed on H_j, then we obtain y as the result of evaluating f on u.

4.5.2 Weighted sampling of solid angle.

In this case, the probability of selecting a point y is equal to the differential solid angle subtended by y when projected on x. This implies that:

$$Q_{xj} \;=\; \frac{1}{G_j(x)}\, \frac{cos(y, w_{xy})}{|x-y|^2} \tag{4.27}$$

where

$$G_j(x) = \int_{S_j} \frac{cos(y, w_{xy})}{|x-y|^2}\, dy$$

note that $G_j(x)$ is the solid angle subtended by patch j when projected on the unit radius sphere around x (not taking into account possible occlusions). The figure 4.5 show an example of the shape of this PDF in a closed environment with no occlusions.

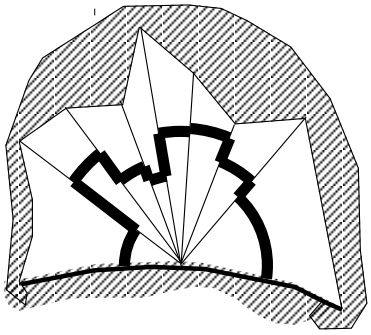

Figure 4.5: Shape of PDF for weighted solid angle sampling.

It is difficult to obtain a point y for all the possible shapes of the patch j. But in the case of planar triangular patches we can resort to the technique described in [Arvo95] to obtain points y with exactly this distribution. We do this by selecting a point A with uniform distribution in $[0,1]^2$, then obtaining B as the image of A under the parametrization function for spherical triangles. Finally, we get C as the first visible point (in the planar triangle) from x in the direction B. Point C is then the sample point y used in the computation This can also be used in the case of polygonal patches which can be decomposed in planar triangles. Figure 4.6 helps to visualize the elements involved in this sampling process.

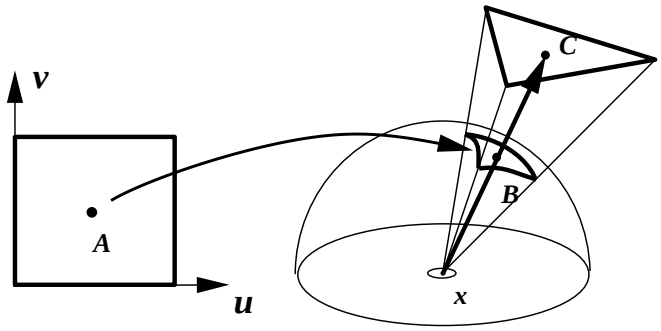

Figure 4.6: Algorithm for weighted solid angle sampling.

4.5.3 Weighted sampling of projected solid angle.

In this case, the value $Q_{xj}(y)$ is proportional to the value $G(x,y)$. Including the necessary normalization constants, we get:

$$Q_{xj}(y) \;=\; \frac{1}{F_j(x)} \frac{cos(x, w_{xy})\, cos(y, w_{yx})}{|x-y|^2} \tag{4.28}$$

where

$$F_j(x) \;=\; \int_{S_j} \frac{cos(x, w_{xy})\, cos(y, w_{yx})}{|x-y|^2}\, dy$$

note that $F_j(x)$ is equal to the point-to-patch form factor from the point y to patch j, multiplied by π. In figure 4.7 we see an approximate picture of the shape of this PDF.

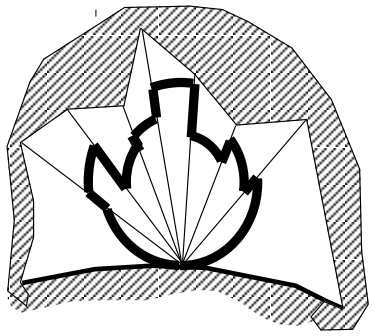

Figure 4.7: Shape of PDF for weighted projected solid angle sampling.

This distribution is the closer one to $G(x,y)B'_a(y)$. In fact, when the error terms Δ_i are all zero, the variance we obtain is also zero. This can be shown by obtaining the expression of the variance. With the above equation, combined with (4.19) and (4.20) we obtain the variance:

$$V(x) \;=\; \frac{1}{n} \left([\, E_g(x) \,+\, \Delta(x) \,] \sum_{i=1}^{m} \frac{E_i^2(x)}{E_i(x) + \Delta_i(x)} \;-\; E_g^2(x) \right) \tag{4.29}$$

when $\Delta_i(x) = 0$, for all i, then $\Delta(x) = 0$ and $V(x) = 0$. This implies that, the closer r_{ij} to $I_j(x)$, then the lower the value we obtain for $V(x)$.

Unfortunately, there is no a known algorithm for selecting points with this distribution. Such an algorithm should be based on the analytical expression of the form factor between a point and a triangle, and using the same principles exposed in [Arvo95] for sampling the solid angle.

4.6 Implementation and Results.

To test the described sampling distributions, we have implemented a two-step method in our rendering system *girt*. In the first step, we compute the function E' as a discrete approximation to the actual irradiance function. This is achieved using photosimulation, or particle tracking, as described in [Pattanaik92]. We have extended the method detailed there in order to compute the amount of irradiance at each patch coming from each other patch.

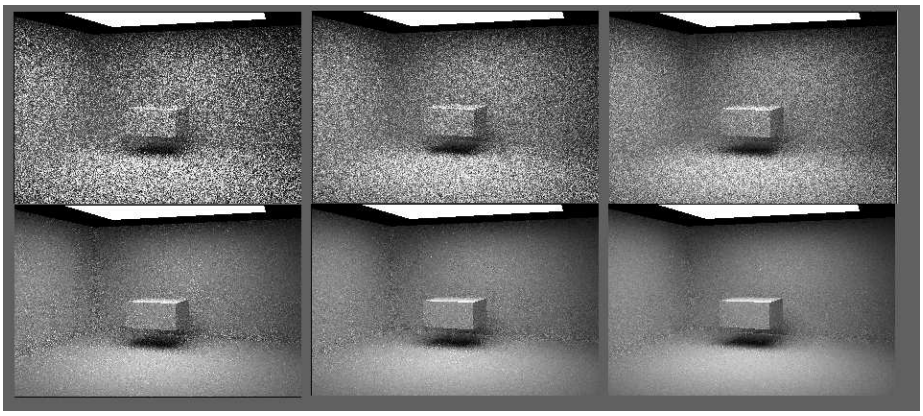

Figure 4.8: Simple scene: 8, 16 y 32 samples per pixel.

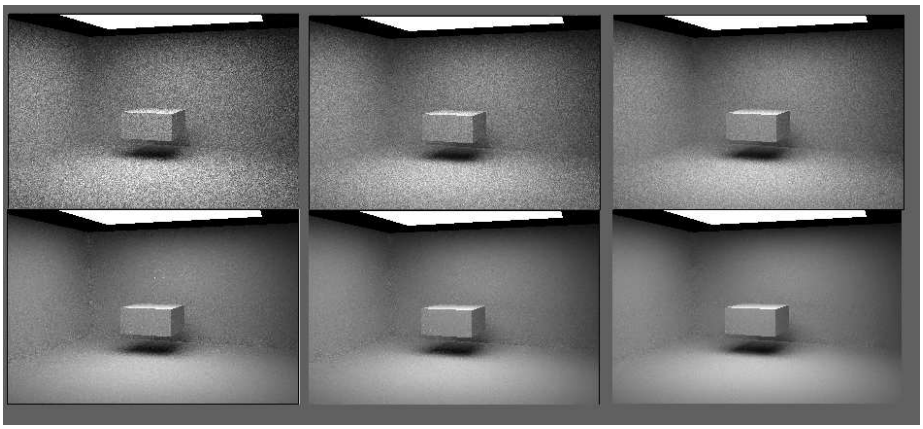

Figure 4.9: Simple scene: 64, 128 y 256 samples per pixel.

After this first step, we perform the second pass. Here we select a view point, and then, for each pixel, we sample the pixel area by using ray-casting. At each sample position we compute the first visible point in the scene, say x. After this

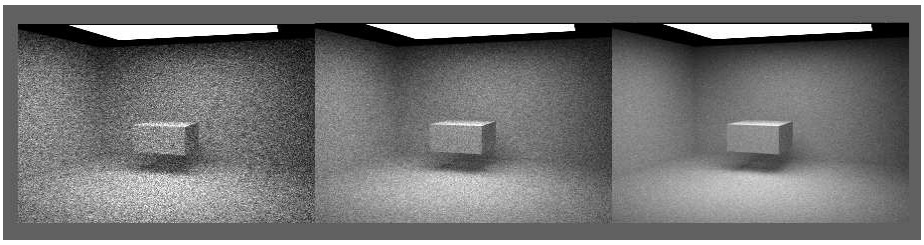

Figure 4.10: Simple scene: path tracing with 64, 256 y 1024 paths per pixel.

we perform final gather over x by selecting a number points y_i in the scene, with distribution $p_x(y)$. Then, by using (4.6) we obtain $E'_g(x)$ as an approximate value of $E_g(x)$. Finally, we compute an approximate value of the radiosity at x, which can be easily obtained from the irradiance.

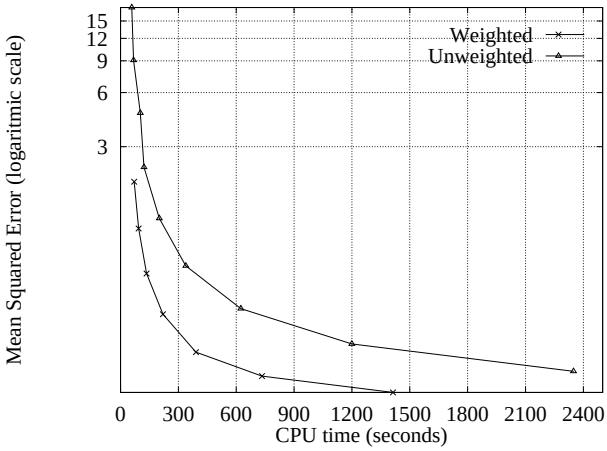

Figure 4.11: Plot of σ^2 versus total CPU time.

We have implemented three different methods to achieve final gather:

- **Projected solid angle sampling**, as described in subsection 4.3.3 This method has the disadvantage that small sources illuminating the target point are very poorly sampled. The image appears too noisy, because this sources have a small chance to be sampled, although they can greatly contribute to the irradiance at the target point.
- **Weighed sampling of parametric area**, as described in subsection 4.5.1 In this case, each sources receives (in the mean), a number of samples which is proportional to its contribution to irradiance at the target. Unfortunatelly, this method shows high variance when the target point is very near the source

patch. This is due to high variations in the differential form factor from the target point to the sampled point.

- **Weighed sampling of solid angle**, as described in subsection 4.5.2. In this case, weighed sampling ensures that important sources are correctly sampled. Even in the case that the target point is near the selected patch, the variance is smaller than in previous method.

The second method is not comparable to the the third, because the noise is vary large when the target point is near the sampled source point. This noise is smaller when using the third method.

	Unweighted				*Weighted*			
ns	*rays*	*CPU*	σ^2	σ	*rays*	*CPU*	σ^2	σ
1	384	58.2	17.818	4.221	384	70.6	1.915	1.384
2	441	67.3	9.078	3.013	441	93.6	1.052	1.026
4	556	102.7	4.617	2.148	555	134.8	0.591	0.769
8	784	121.4	2.307	1.518	783	219.9	0.351	0.593
16	1241	201.0	1.198	1.094	1239	390.8	0.216	0.465
32	2155	338.3	0.652	0.807	2152	733.6	0.159	0.399
64	3983	624.0	0.377	0.614	3976	1413.0	0.129	0.360
128	7639	1199.2	0.240	0.490	7625	2782.0	0.114	0.338
256	14952	2348.9	0.169	0.411	14924	5509.4	0.107	0.328

Figure 4.12: Results from numerical comparisons.

With respect to the other two methods, we have run simulations on a very simple scene with two walls, the floor, a large light source on the ceiling, and a cube. For every target point, the light source covers a large solid angle, so projected solid angle sampling shows less variance than in scenes with small light sources, which are unlikely to be sampled. 100.000 particles where shot from the light sources. The scene is meshed using 100 patches.

Figure 4.12 shows the results we obtained for the described scene. The first column gives the number of samples, and covers all the powers of 2 from 1 to 256. After this we give CPU time in seconds and the number of rays in thousands. Both previous values are for the whole simulation, including particle tracing. The errors are measured with respect to a solution obtained with pure MC path tracing (without explicit light source sampling), and using 1024 paths per pixel. We show the average squared error per pixel (σ^2), and the root of that value (σ). Values are for the whole simulation, including particle tracing. Note that CPU times are higher in the case of weighed sampling for the same number of samples. This is due to the cost involved in sampling spherical triangles[3] A plot of σ^2 (with CPU time in

[3]The current implementation is not optimized in the sense that a new spherical triangle is built for each sample. We would have obtained less CPU time in the case of caching and reusing the spherical triangles whenever possible, as described in [Arvo95].

the X axis) for the two methods is shown in figure 4.11. As can be seen, weighed sampling of solid angle performs better than projected solid angle sampling.

Besides this numerical errors, the images obtained by projected solid angle sampling show too much more noise than those obtained with weighed solid angle sampling, even for similar CPU times. This can be seen in figures 4.8 and 4.9. Each of those figures show two rows of images. The first row was obtained with projected solid angle sampling, the second with weighed solid angle sampling. At each column, the number of samples taken is the same for both images. These images where used for the numerical comparisons.

Figure 4.10 shows three images obtained with MC Path Tracing, by using 64,256 and 1024 paths per pixel. The last of them is the one used as reference for all the others.

Finally, figure 4.13 shows a room with 540 objects, meshed in 800 patches, and rendered with weighed solid angle sampling. For this image, 4 samples where taken for each pixel, and 50 samples where taken for final gather.

4.7 Conclusions.

The use of *a priori* information about a function can help in the design of sampling distributions for Monte-Carlo integration of that function. This is the basic idea presented in this chapter. In the context of radiosity computations, we have shown how the first pass (in two-pass algorithms) can be used to gather information about the distribution of radiance transfers. This information is used in the second pass in order to reduce the time needed for sampling, by assigning a higher probability to relevant sources or reflectors.

By using the above idea, we have designed a final-gather algorithm with reduced variance. We have compared the results obtained for a simple scene, by using the proposed method, standard path tracing and final-gather without *a priori* information. These test include both numerical comparisons of the error and visual comparison of image quality. The underlying system used for implementation was the same for all the cases. The proposed method resulted the most efficient of all, because its reduced variance.

Figure 4.13: Room with 800 patches, and 4×50 samples for final gather.

Chapter 5

A two-pass Monte Carlo algorithm for Global Illumination.

5.1 Introduction

In this paper we propose a two-pass Monte-Carlo algorithm for glossy environments. In the first pass, a low resolution approximation to the radiance function is obtained by using a particle tracing algorithm. In the second pass, an image is produced after a final-gather (or local-pass) step at the pixel level, as has been done for finite-elements global illumination algorithms. This final gather step is improved by importance sampling with reduced variance. Importance sampling is based on the information about energy exchange between surface elements which is stored in an hierarchical structure of links.

With respect to previous finite-element methods, the use of stochastic sampling avoids explicit sampling of every link during final-gather. As in previous chapter, the number of samples taken is not related to the number of links in a receiver patch, but it is related to the variance. This variance can be greatly reduced when visual importance is taken into account in the links structure, specially for glossy environments.

The new proposed algorithm extends the one presented in previous chapter to glossy environments while reducing the time and storage complexity shown in that work. In that paper, improvement of the final gather step was obtained by explicit representation of energy transfer between every pair of patches. This is too inefficient for medium and high complexity scenes. However, the involved complexity can be reduced by a view-dependent first pass (which yields the amount of each energy transfer weighted by its importance to the observer), and also by the use of an hierarchical clustering approach for the representation of those transfers.

Computing radiance increases the dimension of the problem (with respect to

computing radiosity), because radiance is defined on the ray space while radiosity is defined just on the two-dimensional space of surface points.

The classic radiosity equation and the radiance equation are both second order integral equations with a two variable kernel, and a one variable, unknown, function. This unknown function can be represented, in both cases, as a low resolution function stored by using a set of coefficients for a given basis. These coefficients can be approximately computed by particle tracing.

5.1.1 Previous work.

As has been stated in previous chapter, the problem of refining a low resolution approximation to the radiosity function has been largely studied in the field of realistic visualization. To all the papers referenced in the previous chapter (which deal with diffuse environments) we can add the paper by Jensen [Jensen95] for glossy environments. In that paper, it was described a two-pass algorithm, including a particle-tracing first pass and a rendering second pass. After the first pass, the set of particle impact points and incoming directions is stored. In the second pass, that set is used to guide the sampling process in order to obtain the final image.

The sampling process which takes place in the second pass is a modified path-tracing algorithm. The selection of a direction on the hemisphere for secondary, reflected rays is done by using a discrete *pdf* which has been built by using the impact position and directions around the sample point.

This has the disadvantage that the creation of the *pdf* may slow down the algorithm. Furthermore, important sources covering a small solid angle can still be missed, because of the limited resolution of the discrete *pdf*. In our rendering system, we have tried to solve this by using the links structure, because those links allow us to direct the sample to appropriate sources.

5.1.2 The proposed algorithm.

If a structure of links is considered between all the possible pairs of ray space elements then a scalar value for any given pair of those elements should be stored. This would imply an overwhelming cost in terms of storage and time, because the number of basis used to approximate radiance is an order of magnitude higher than the number of basis (patches) used to model the radiosity function.

In order to overcome this limitation, we consider the nature of the kernel K involved in the definition of radiance, as defined in equation (1.35). The presence of a delta function in it implies that for almost all of the pairs of rays there is no direct energy transfers between them. In fact, we have just to account for energy being transfered between trios of surface points, by using a three point transport formulation, as done in ([Aupperle93]). The resulting complexity is then in the order of n^3, where n is the number of surface patches, which is still very high for medium or high complexity environments.

The complexity can be further reduced by using a view dependent first pass. We do not need to represent the power being transfered between all the possible trios of surface points, because the length of the paths in the second pass is just two. Then,

the first point is always the observer position, and thus is fixed for the simulation. The representation of the power transfers is a two-variable function, which gives, for all pairs of points x, y the power going from x to y and then bouncing towards the observer. This is done at the cost of obtaining view-dependent results, but complexity is kept in the order of n^2, as in the previous chapter.

This is still a very high complexity for environments with a huge number of patches, and is the main drawback of the technique exposed in the previous chapter. The next step is to consider hierarchical clustering techniques. Lets consider any fixed receiver patch. The algorithm described in previous chapter uses the distribution of irradiance on that patch coming from each patch in the environment, and this leads to $O(n^2)$ complexity.

A more efficient algorithm can be considered by discarding transfers of energy whose weight is below a given threshold. Then the maximum number of transfers considered for the receiver is bounded (and this bound is inversely proportional to the threshold). This leads to an hierarchical representation of the power transfers on a patch. This is done by structuring the scene in a cluster-patch hierarchy. For each receiver patch node, a link is kept which holds the energy coming from a given source cluster or patch.

5.2 Clustering

The clustering algorithm we have used is described in [Christensen97]. As a result of the clustering process, a tree is obtained in which each node is associated to a portion of the surfaces in the environment. The area covered by any node is a subset of (or equal to) the one covered by its parent. The root node covers all the surfaces of the objects.

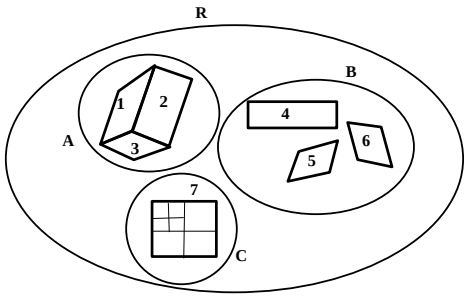

Figure 5.1: A set of clusters and objects.

We classify the nodes in three types: cluster nodes, object nodes, and patch nodes. A cluster node covers a set of objects, an object node covers a single object, and a patch node covers a portion of the surface of an object. Direct descendants of a cluster node can be either other cluster nodes or object nodes. Object nodes have patch nodes as descendants (one for each surface the object is made of). Finally,

patch nodes are either leaf nodes or point to other leaf nodes.

The algorithm used for clustering is as follows: first, the root node is created and associated to all the objects in the scene. When a cluster node is created, each geometric object in that node is tested in turn (the list of objects for the cluster is given as a parameter to the constructor method). If its size exceeds half of the bounding box size, it is attached as a direct descendant of the cluster. Otherwise, we consider the bounding box of all the objects in the node, and its eight octants. The center of the object bounding box is processed and the octant it belongs to is obtained. Then, we include the object in that octant. When a cluster node includes a single geometric object, a object node is created and this is the only son of the cluster.

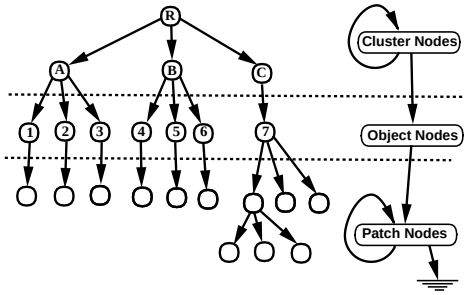

Figure 5.2: The corresponding tree for figure 5.1.

A object node is always composed of a set of patch nodes. These patch nodes are always split in four sub-patch nodes, each covering a quadrant of the original one, except when patch area is smaller than a threshold. In this later case, the patch node is a leaf node, and no further processing is required.

For all the nodes in the hierarchy, their axis-aligned bounding boxes are stored, the list of pointers to their children (except for leaf nodes, of course).

Figure 5.1 shows an example of a object-patch hierarchy. Each ellipse includes a cluster, which is mapped to a node in the tree. The associated tree for this object set is shown in figure 5.2. In this tree, three levels are shown (cluster nodes, object nodes and patch nodes).

5.3 First Pass

The first pass of the proposed method is just a particle tracing algorithm, extended in order to process the links between elements in the cluster-patch hierarchy. In the next two subsections we explain the details about particle tracing, and the algorithm used to update the links.

5.3.1 Particle tracing

The particle tracing process implemented is, in essence, as described in [Pattanaik92]. Particles are created at light sources, and traced through the environment. All light sources do produce energy with a diffuse distribution, and with a constant intensity inside each source area. Then, each source is selected with probability proportional to its total power emitted.

After the source is selected a random point inside it is obtained, with a uniform distribution with respect to the area measure. Then the initial outgoing direction for the particle is computed. This mean that the probability measure function for the first ray of a particle is equal to $L_e(\mathbf{r})d\mu(\mathbf{r})/\phi_e$ where ϕ_e is the total power emitted by all the sources in the environment.

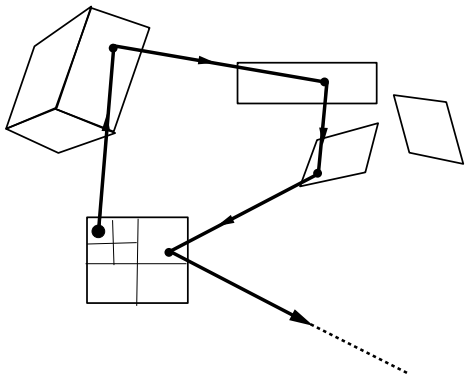

Figure 5.3: A particle path.

After a particle is created, a stochastic simulation of the path it travels takes place. The simulation ends whenever the particle is absorbed on a bounce, [Pattanaik92], or leaves the scene (which is not necessarily closed). The selection of the new particle direction after a bounce is done by importance sampling of the BRDF. Note that the particle path hits, at each bounce, a leaf node in the cluster-patch hierarchy. Thus, we can view the path as a sequence of leaf nodes in the hierarchy.

In figure 5.3 we can see an example of the path traversed by a particle, through a set of objects. This path is mapped to a sequence of nodes in the tree, as can be seen in figure 5.4.

5.3.2 Particle state data.

The state information for a particle includes all the necessary data for computing the radiance function. This information is updated after each bounce. It includes:

- The point where the bounce happened (or the point where the particle was created on a light source)
- The direction vector of the particle after the bounce.

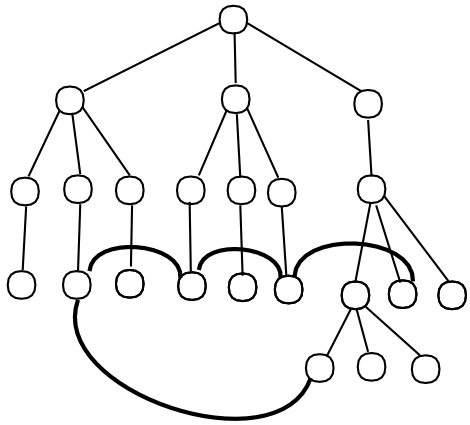

Figure 5.4: Leaf nodes visited by path shown in figure 5.3.

- The weight of the particle, that is, the amount of power it carries.

When the environment is monochrome, a scalar value for the weight is enough, but in fact all the environments show surfaces with a reflectance and emissivity which depend on wavelength. This can be solved by using a number of simulations, each of them for a fixed wavelength, and then combining the resulting images, as usually is done in finite elements algorithms. Another option is to assign a wavelength to each particle, by using a random distribution proportional to the spectral distribution of particle source emissivity as described in [Pattanaik92]. This can be called single wavelength particles. Finally, a third method is to assign an RGB triple to each particle, as done in [Jensen95].

This later option can be generalized to account for an arbitrary spectral distribution. This means that a particle weight is a function of wavelength w, such that $w(\lambda)$ is the actual weight of the particle at wavelength λ. When the particle hits a surface, the directional-hemispherical reflectivity for the surface point and particle incoming direction is represented also by a function of wavelength $\rho(\lambda)$ (which depends on the BRDF).

This representation is introduced in order to differentiate between the model used to store energy distributions and the particle tracing algorithm, as required by our rendering architecture [Urena97]. Single wavelength particles can be viewed as a particular case of this scheme, in which the energy distribution is a delta function with a spike in the particle wavelength.

5.3.3 Survival probabilities.

In the case of simple scalar energy values at a fixed wavelengths, the reflectivity is also a scalar value between 0 and 1, and thus it can be used as the survival probability when simulating particle bounces. Unfortunately, when the energy has various components, the reflectivity can be different for each component (the reflectivity is

a function of wavelength we call this function ρ), although it is still necessary to use a given survival probability. This is done by using the luminance of the particles.

Lets consider the luminous efficiency curve[1] c, then the inner product $\langle w|c\rangle$ is[2] the luminance of the energy distribution w. When a particle hits a surface, we consider the luminance of the distribution[3] $w\rho$, which is $\langle w\rho|c\rangle$. This luminance is necessarily smaller than the particle luminance $\langle w|c\rangle$, because ρ is bounded by 1. Then, the survival probability we use is $r = \langle w\rho|c\rangle \,/\, \langle w|c\rangle$, that is, the ratio of luminances between both distributions.

This scheme implies that the luminance of a particle is kept constant along the path. Other methods, such as *Russian Roulette* can also be implemented by using this survival probability. When implementing this, it is necessary that the interface for energy distribution objects allows to obtain their luminance, and also includes a pointwise product operator. Thus, the interface is independent of the underlying color model used. In fact, this color model must be based in some kind of projection of the curves in a discrete basis, thus involving some error in the approximation.

5.3.4 Radiance function approximation.

When a particle leaves a surface point after a bounce (reflection) on it, it is necessary to update the estimators used to approximate the radiance function. Any basis function set defined in the space of point and directions can be used. In our implementation we have selected the simplest approach, which is constant, non overlapping basis functions.

As noted earlier, all the scene objects are meshed in patches. For each patch an array of energy distributions is allocated, and initialized to zero. Each entry in the array holds an estimation of the average radiance through a set of directions in the unit radius hemisphere. Let us assume that patch number i covers a region $S_i \subseteq S$, and that there exists n patches.

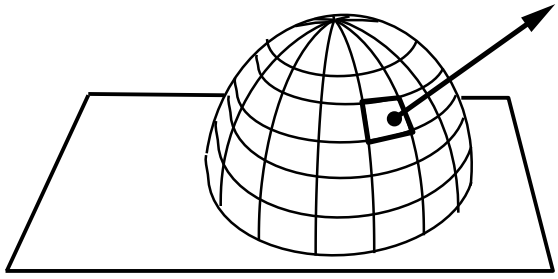

Figure 5.5: Hemisphere meshed in beams.

[1] that is, $c(\lambda)$ is proportional to the human eye sensibility to a unit of energy at wavelength λ

[2] this inner product $\langle w|c\rangle$ is defined by an integral over the one-dimensional set of all wavelengths, by using the standard length measure on the real line, that is, $\langle w|c\rangle = \int_R w(\lambda)c(\lambda)d\lambda$.

[3] The expression $w\rho$ stands for point-wise function product in the real line, that is, $(w\rho)(\lambda) = w(\lambda)\rho(\lambda)$.

The hemisphere is meshed in a number of regions. These regions are defined by a set of parallels and meridians. Each region covers a subset b_{jk} of the hemisphere. The set of rays in region jk, with origin at patch i will be called D_{ijk}, with $D_{ijk} = S_i \times b_{jk}$. The number of parallels is n_p and the number of meridians is n_m. Thus, the set of basis functions is defined as:

$$\mathbf{B} \;=\; \{f_{ijk} \mid \;\; 1 \leq i \leq n \;,\; 1 \leq j \leq n_m \;,\; 1 \leq k \leq n_p\} \tag{5.1}$$

where each function f_{ijk} is the indicator function for region D_{ijk}, that is:

$$f_{ijk}(\mathbf{r}) = \begin{cases} \dfrac{1}{\mu(D_{ijk})} & \text{when} \;\; \mathbf{r} \in D_{ijk} \\ \\ 0 & \text{otherwise} \end{cases}$$

It can be easily proved that the above set $\mathbf{B}$ of basis functions is orthonormal. The average reflected radiance c_{ijk} on the region jk of patch i can be written as follows:

$$c_{ijk} \;\;=\;\; \frac{1}{\mu(D_{ijk})} \int_{D_{ijk}} L_r(\mathbf{r})\, d\mu(\mathbf{r}) \;\;=\;\; \langle f_{ijk} \mid L_r \rangle \tag{5.2}$$

where the above inner product is defined with respect to the μ measure. Then, the function L_r is approximated by its projection on $\mathbf{B}$, which is the following function

$$(\mathcal{P}_{\mathbf{B}} L_r)(\mathbf{r}) \;\;=\;\; \sum_{ijk} c_{ijk}\, f_{ijk}(\mathbf{r})$$

The coefficients c_{ijk} can be approximated by using particle tracing. Whenever a ray bounces on a surface point at patch i, and survives, the estimators for radiance are updated. This is done by finding, from the ray direction, the beam jk in which it is included. Then the power distribution $w f_{ijk}(\mathbf{r})$ is added to the current value of the estimator c_{ijk}, where w is the current weight of the particle. Each c_{ijk} is a power distribution, initialized to zero at the beginning of the simulation. At the end of it, this distribution is taken as an estimator of $\langle L | f_{ijk} \rangle$.

5.3.5 Link updating

Whenever a particle travels from a source path to a receiver patch, the information stored in the link structure is updated. This is done because the particle transition models a quantum of energy which travels from source to receiver. The first task is to find the link the particle contributes to. As can be seen in figure 5.6, we consider the pair of leaf nodes visited. Then a search is made in order to determine whether a link exists from receiver to source. If it is not found, the process is (recursively) repeated for the parent of the source, until we reach the root node. Necessarily, the source or one of its ancestors includes a link from the receiver.

When the link has been found the data stored in it is updated. This data is composed of two values: an integer one used to count the total number of particles traversing the link, and an scalar value which is called the link weight, and represents

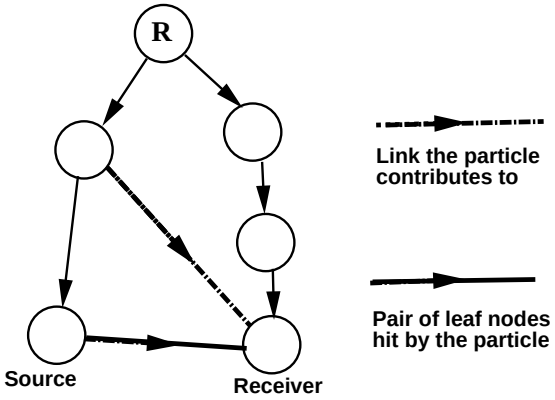

Figure 5.6: Finding the link a particle traverses.

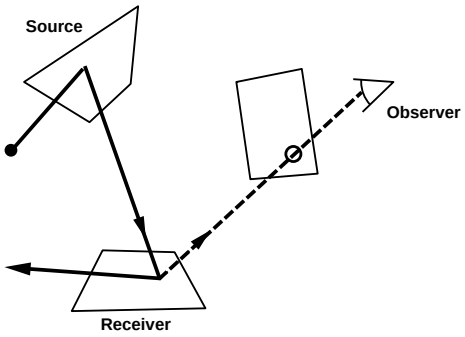

Figure 5.7: Geometry for a particle transition between two surfaces.

the luminance of the fraction of power going through the link and reflected in the observer direction (see figure 5.7). Thus, this data is updated by incrementing the particle count. With respect to the link weight, the following scalar value d is added to it:

$$d \;=\; \langle wr \mid c \rangle \; \frac{n_t}{n_t - n_0} \tag{5.3}$$

where w is the current particle weight, r is a reflectivity distribution obtained by evaluating the BRDF in the intersection point at the receiver, (thus wr is the distribution of power reflected in the observer direction), c is the luminous efficiency curve, n_t is the total number of particles shot during the whole simulation, and n_0 is the number of particles already shot when the link was created, with $n_0 < n_t$. That is, d is the luminance of wr, corrected by the factor $n_t/(n_t - n_0)$, which is necessary because $n_t - n_0$ is the total number of particles shot during the link life.

5.3.6 Link refinement

Link refinement is based on adaptive splitting of links by using the count of particles traversing each of them.

Each link is said to go *from* a receiver node *to* a source node, because it is stored in list at the receiver and includes a pointer to the source. This is done so because links are queried from the receiver at the second pass, that is, once we know the receiver, we must know the set of links *leaving* it and their corresponding source nodes.

Before particle tracing starts, a links is created from every leaf node to the root node. Once a given number of particles n_{min} has gone through the link, it is subdivided. If this links goes from leaf node a to node b, then a set of new links is created. Each of them goes from a to a child of b. When a link is created, its particle count and its weight are initialized to zero. The value n_0 for the new link is the current number of particles shot.

Links which have been split are ignored from that instant, and only non-split links are taken into account for the second pass. With this scheme, it is possible the existence of a link with the value n_0 very close to n_t, that is, the link was created near the end of the simulation. The weights of these links are very bad estimators of the exact weights, because the value $n_t - n_0$ (the number of particles shot during link life) is very small, and the variance can be very high. To avoid this,

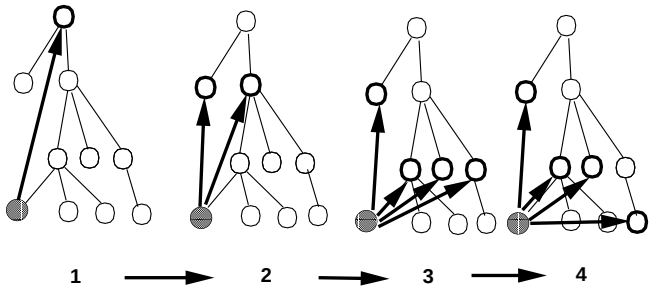

Figure 5.8: Refinement of a link.

we establish a number n_{max} (with $n_{min} < n_{max} < n_t$) such that no link is created with $n_0 > n_{max}$. That is, after the number of particles shot becomes higher than n_{max}, no link is split. With this, the minimum number of particles shot during a link life is $n_t - n_{max}$.

At the end of the simulation there exists, for each leaf node, a set of links pointing to others nodes in the hierarchy. In figure 5.8 you can see how a link (from a leaf node to the root) is refined during the simulation, and the resulting links from that node.

5.4 Second Pass

The second pass is used to obtain an image of the environment as viewed by the observer. This is done by using a final gather step. In this context, final gather is the problem of finding the value $L_r(\mathbf{r})$ for a given ray $\mathbf{r}$, by applying the transport operator $\mathcal{T}$ to a low resolution radiance function L'. This final gather is done by using Monte-Carlo importance sampling, guided by the information stored in the links structure.

First, a ray is shot from the observer towards the scene, passing through the pixel area. The first visible point x on the scene, if any, is obtained. Then, the problem is to obtain the radiance towards the observer from the point x, that is, we face the computation of $L_r(\mathbf{r}) = L_r(x, w)$, where w is the unit direction vector from x to the observer. This is done by sampling a given set of points x_i on the scene surfaces with a given probability density function, and querying the approximate radiance reflected at x towards the observer from those points.

5.4.1 Problem statement.

As in the previous chapter, this final gather can be formalized as the computation of the following integral

$$L(\mathbf{r}) \;=\; L_e(\mathbf{r}) \;+\; \int_D K(\mathbf{r},\mathbf{s})\, L'(\mathbf{s})\, d\mu(\mathbf{s}) \tag{5.4}$$

where L' is the low resolution radiance approximation computed in the first pass (which will be called L'), that is, we compute $L = L_e + \mathcal{T}L'$. The value $L_e(\mathbf{r})$ is known for any $\mathbf{r}$, thus the problem is the computation of the integral. We can conceive a probability density function $p_{\mathbf{r}}$ such that:

$$1 \;=\; \int_D p_{\mathbf{r}}(\mathbf{s})\, d\mu(\mathbf{s}) \tag{5.5}$$

$$K(\mathbf{r},\mathbf{s})L'(\mathbf{s}) > 0 \;\;\Rightarrow\;\; p_{\mathbf{r}}(\mathbf{s}) > 0 \tag{5.6}$$

Once a valid p is selected we can conceive Monte-Carlo algorithms for solving (5.4). This is done by selecting a set of n rays in the environment $\mathbf{s}_i = (y_i, w_i)$ with probability density function $p_{\mathbf{r}}$, and computing an estimator of $L(\mathbf{r})$ by using the following random variable $X_{\mathbf{r}}$

$$X_{\mathbf{r}} \;=\; \sum_{i=1}^{n} \frac{K(\mathbf{r},\mathbf{s}_i)\, L'(\mathbf{s}_i)}{p_{\mathbf{r}}(\mathbf{s}_i)} \tag{5.7}$$

it is easy to prove that $E(X_{\mathbf{r}}) = L(\mathbf{r})$, and thus $X_{\mathbf{r}}$ is a unbiased estimator of $L(\mathbf{r})$. Then the goal is to build a sampling distribution which approaches the zero variance sampling distribution. This zero variance sampling distribution is

$$p_{\mathbf{r}}(s) \;=\; \frac{K(\mathbf{r},\mathbf{s})\, L'(\mathbf{s})}{(\mathcal{T}L')(\mathbf{r})} \tag{5.8}$$

This example of $p_{\mathbf{r}}$ obviously obeys (5.5) and (5.6), and has zero variance because is proportional to $K(\mathbf{r},\mathbf{s})\, L'(\mathbf{s})$.

5.4.2 A *pdf* for final-gather.

The goal is to obtain an optimal $p_{\mathbf{r}}$ as near as possible to the ideal one, as stated in previous subsection. This can be done by using the link structure. The set of links leaving any given leaf store an approximate model of the distribution of power landing on that leaf node (weighted by the importance towards the observer) Thus, we can use a density function which is proportional to that distribution, in order to reduce the variance. This means that, if a link exists from a leaf node i to an arbitrary node j with weight r_{ij}, then for all pairs of rays $\mathbf{r}$ (with origin in node i) and $\mathbf{s}$ (with origin in node j) the value $p_{\mathbf{r}}(\mathbf{s})$ is proportional to r_{ij}.

The link from i to j is a leaf link, that is, no others link exists from i to any descendant of j. Thus, links weights are not enough to fully characterize $p_{\mathbf{r}}$, because they give no information about the shape of $p_{\mathbf{r}}$ inside the piece of surface (or set of objects) covered by node j. Once the source link j is selected, we still have to descend the tree until a leaf node is found, and then we have to select a point inside that leaf node.

This can be formally expressed by the following definition of $p_{\mathbf{r}}(\mathbf{s})$

$$p_{\mathbf{r}}(\mathbf{s}) \;=\; \frac{r_{ij}}{r_i}\, t_{\mathbf{rs}}(j,k)\, q_{k\mathbf{r}}(\mathbf{s}) \tag{5.9}$$

where i is the index of the leaf node where $\mathbf{r}$ resides, k is the index of the leaf node where $\mathbf{s}$ resides, j is the first ancestor of k such that a leaf link exists from i to j, r_{ij} is the weight of that link, r_i is the sum of the weights of all links starting at node i (that is, $r_i = \sum_j r_{ij}$), and $q_{k\mathbf{r}}(\mathbf{s})$ is a normalized *pdf* defined for the rays starting at node k, which gives the probability for selecting $\mathbf{s}$ as source. Finally, $t_{\mathbf{rs}}(j,k)$ gives the probability for going down the hierarchy from node k to leaf node j. The function t is iteratively defined as follows

$$t_{\mathbf{rs}}(j,k) \;=\; \begin{cases} 1 & \text{when } \; j = k \\ f_{\mathbf{r}}(j,h)\, t_{\mathbf{rs}}(h,k) & \text{otherwise} \end{cases} \tag{5.10}$$

where h is the direct descendant of j which covers the origin of $\mathbf{s}$ and $f_{\mathbf{r}}(j,h)$ is a value function which gives the probability for selecting h as the son node of j, when $\mathbf{r}$ is the target ray.

Once $\mathbf{r}$ is known, we have to select a set of sample rays, with distribution $p_{\mathbf{r}}$. Each sample ray $\mathbf{s}$ is selected by following the next steps:

1. Obtain the index i of the node where the origin of $\mathbf{r}$ lies.
2. Select a node j with probability r_{ij}.
3. While node with index j is not a leaf node:
   (a) Select a son node (with index h) of node j, with probability $f_{\mathbf{r}}(j,h)$.
   (b) Make j become h.
4. Select a ray $\mathbf{s}$, with origin in node j, by using the probability distribution $q_{j\mathbf{r}}(\mathbf{s})$.

In the following subsection, we detail the functions $f_{\mathbf{r}}$, which governs tree descending, and $q_{k\mathbf{r}}$, which is used to select a ray inside node k.

5.4.3 Tree descending.

The algorithm introduced in the previous section is not complete, in the sense that it is still unspecified which functions $f_{\mathbf{r}}$ and $q_{k\mathbf{r}}$ are used.

Lets assume that j is non-terminal node, with a son node h, and that the target ray is $\mathbf{r}$. Then, the scalar value $f_{\mathbf{r}}(j,h)$ is the conditional probability that the sample point lies in node h, knowing it lies on j. The better strategy is to make $f_{\mathbf{r}}(j,h)$ equal to the following fraction:

$$f_{\mathbf{r}}(j,h) \;=\; \frac{L_h(\mathbf{r})}{L_j(\mathbf{r})} \tag{5.11}$$

where $L_i(\mathbf{r})$ is the radiance at $\mathbf{r}$ due to energy coming from node i. This is impossible to implement because these are unknown values whose computation is expensive because of the nature of the transport function K.

In order to approximate the value in equation (5.11) we use an approximation to the function L_i. This approximation is necessarily easy to compute in order to avoid long execution times, and also enough close to the exact value to prevent very high variance. We have used the following expression:

$$L_i(\mathbf{r}) \;\approx\; L'_i(\mathbf{r}) \;=\; f_r(x,w,w')\,\Phi_i\,cos(n_x,w)\,s(x,i) \tag{5.12}$$

where $\mathbf{r}=(x,w)$, f_r is the BRDF function, w' is the vector $c_i - x$ normalized, Φ_i is the total energy of all the particles leaving node i and the point c_i is defined as follows: when node i is a planar patch, c_i is the center of the patch. In the case that i is a cluster or an object, c_i is the center of its axis-aligned bounding box. Finally, the value $s(x,i)$ is an approximation to the solid angle covered by node i when projected over x.

In the case that i is a planar patch node with area a and normal n_i, we approximate s by using a planar disc with area a and radius r and whose normal points to x. This solid angle can be exactly computed. This value is then multiplied by $cos(n_i,-w')$, in order to (approximately) take into account the reduction in the solid angle covered because of the relative slope of the patch with respect to the point x. Thus, for a planar patch i, s is defined as:

$$s(x,i) \;=\; cos(n_i,-w')\,\frac{r^2}{d^2} \tag{5.13}$$

where $d=|c_i-x|$, note that the fraction above is the solid angle covered by a front facing disc. In the case that node i is a cluster, we set $s(x,i)=r^2/d^2$ where d has the same definition as before, and r is the radius of a sphere whose area equals to the total area of all surfaces in the cluster.

Finally, the value $q_{k\mathbf{r}}(\mathbf{s})$ is the probability for selecting a source ray $\mathbf{s}$ inside leaf patch node k. First, the patch is subdivided in triangles, and one of them is selected according to its covered solid angle. Then we use the Arvo mapping [Arvo95] from the unit rectangle to the projection of the triangle, in order to select a point inside that triangle.

5.5 Results.

A simple scene composed of a rectangular diffuse emitter and a Phong reflector has been used to test the algorithm. The Phong exponent was set to 2. When rendering these images, 200.000 particles where shot from the emitter. The value n_{min} was set to 100, and n_{max} was set to 100.000. The set of images can be seen in figure 5.10. The number of samples for final gather is 1,2, and 4 for the images on the upper row, and 8,16,32 for the images on the lower one. A set of images computed by using raw path-tracing has also been obtained, and can be seen in figure 5.9. The noise for the new algorithm is smaller when compared to path-tracing.

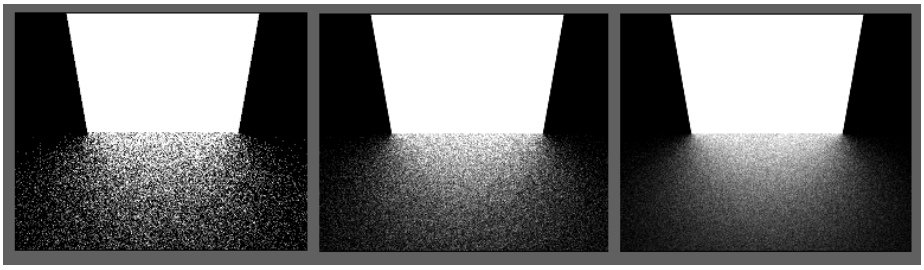

Figure 5.9: Path-tracing with 4,16 y 32 paths por pixel

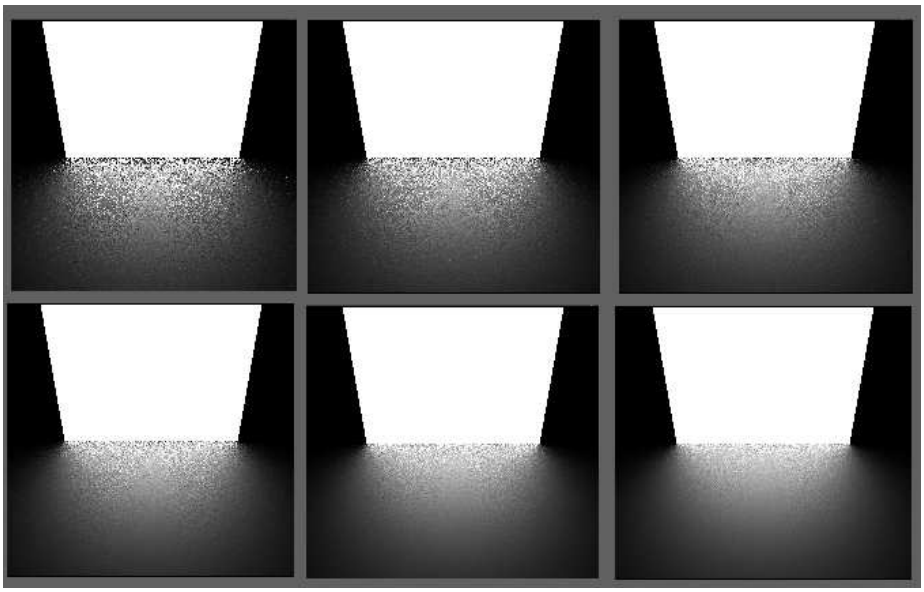

Figure 5.10: New method with 1,2,4,8,16 y 32 samples per pixel.

As can be seen, the variance (noise) in the reflector is higher for points near the emitter and very low for points far away the emitter. This is because, for far away points, the differential form factor from the receiver point to each source node point is almost a constant, while for near points this value shows high variations. Note that in the far region the variance is so small that even by using just one sample per pixel the noise is not noticed. Anyway, this variance is smaller than a similar simulation using BRDF sampling, because the samples are directed to interesting parts of the source.

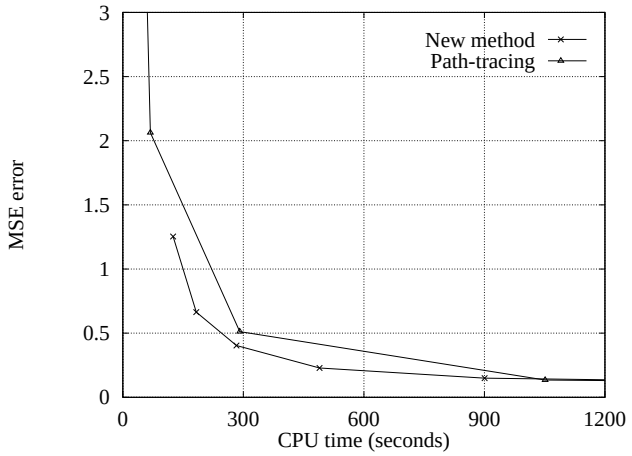

Figure 5.11: Numerical comparison for simple glossy scene.

We have done numerical comparisons with the before mentioned images. We have used as reference a path-tracing image obtained with 1024 paths per pixel. We have computed the difference between each image and the reference image. This difference is the mean of the squared differences at each pixel (MSE error). Path-tracing images have been obtained by using 1,4,16,64 y 256 samples per pixel. The images obtained with the new method has been obtained by using 1,2,4,8,16 y 32 samples for final-gather. A graphical picture of the error can be seen in figure 5.11, where it is compared to total computing time (including the first pass for the new method). As can be seen, the new method outperforms raw path-tracing, even by taking into account the time necessary for the first pass. Both implementation use the same code for intersection computations.

In figure 5.12 we can see a medium complexity scene (including two table-chairs sets) rendered with both path-tracing (left) and the new method (right). Path-tracing took 11 CPU hours, and was done by using 256 paths per pixel. The new method took 7 hours of CPU time to render, and we used 4x50 (equivalent to 200) samples for final gather. We can see how the noise is clearly higher for path-tracing.

By using this scene (the two table set), we have synthesized a set of images for numerical comparisons. The set of path-tracing images has been obtained by using 1,4,9,16,36,64,121 and 256 paths per pixel. The RMSE error (root of mean squared

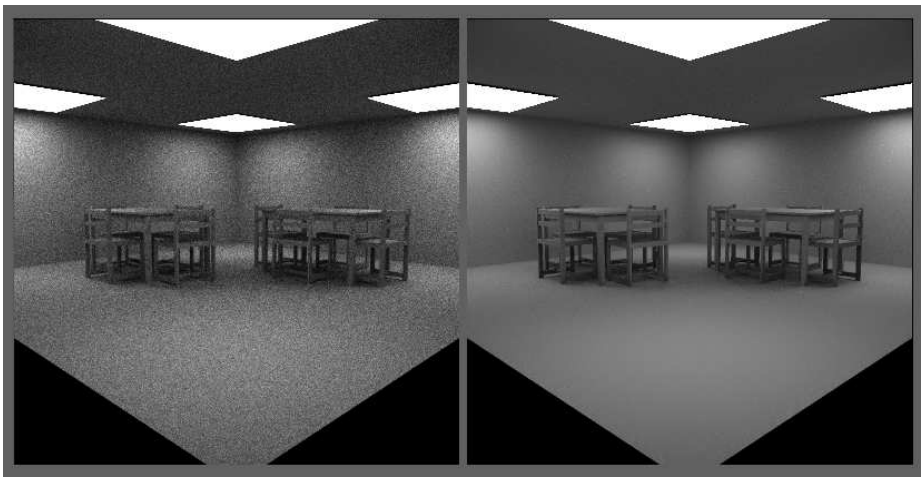

Figure 5.12: Quality comparison for a medium complexity scene.

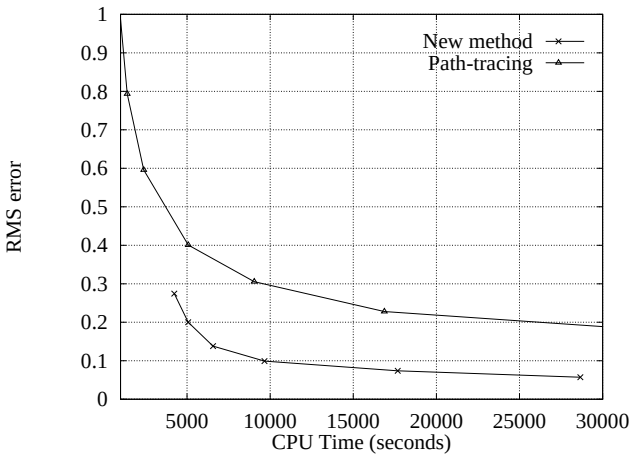

Figure 5.13: Numerical comparisons for a medium complexity scene.

error) was obtained by comparison with a path-tracing image with 1024 paths per pixel. We have also produced images with the new method by using 8,16,32,64,128 and 256 samples per pixel. For this set, we have computed RMSE error with respect an image obtained also by the new method by using 256 samples per pixel. The error is plotted in Y axis, while X axis represents total CPU time in seconds.

In figure 5.14 we observe a complex scene with a number of table and chairs sets. This scene has been rendered by using the new method with 256 samples for final gather, and 4 samples per pixel. The number of particles (from left to right)

Figure 5.14: A complex scene with 3,6 and 12 million particles.

was 3,6 and 12 millions. It can be observed how table and chairs legs are darker in the left image, and brigther in the right one. This is because those small objects did not receive enough particles in the 3 millions particles simulation, but do so in the 12 millions particles simulation. The links weights are better estimated in the right image, thus producing a more exact image with smaller variance.

5.6 Conclusions.

In this chapter we describe an algorithm which extends the one presented in previous one to glossy environments. Here, we use an already existing method for the creation of cluster-patches hierarchies, but applied to Monte-Carlo techniques, instead of finite-elements methods. We propose a new method for the creation of a hierarchy of links by using particle tracing in the first pass. Later, this hierarchy is used for importance sampling in the second pass. Moreover, we show how a recursive algorithm can be used to get samples from the link structure, by a traversal algorithm which runs over the cluster-path tree and selects leaf nodes.

Chapter 6

Conclusions and Relevant Contributions.

In this chapter we include the conclusions and an abstract of the more relevant contributions found in this dissertation.

The main conclusion we obtain from this dissertation is that Monte-Carlo algorithms can be used as an efficient tool for the computation of Global Illumination on arbitrary scenes. This techniques are frequently considered as low-efficiency techniques because of their high variance, and this has limited the research about and use of them. However, for the class of problems that realistic visualization faces, it is possible to design good sampling strategies which improve the efficiency of the sampling process thus reducing the necessary computing time. Although the use of naive algorithms is inefficient, we show how to improve the technique by using importance sampling.

In the first chapter we show an integral radiance equation which has the form of a second order integral equation of the second kind. This is done by deriving the expression for function K which is the kernel of the integral operator involved in that equation. A number of numerical algorithm have been previously designed and applied for that kind of integral operators, and can be also applied to that radiance equation. Previous forms for the radiance equation do not have this form, because the kernel is a three variable function as in [Kajiya86], or because the transport operator is defined as the composition of other two, as in [Arvo95]. In this chapter we also use the previous formulation in order to briefly introduce existing computational methods for integral equations, that is, finite element methods (we restrict to Galerkin method) and Monte-Carlo methods (path-tracing and particle tracing).

In the second chapter we introduce a formalization of the Global Illumination problem. We state that this problem is the computation of a set of functionals or inner products of the radiance function, which in turn is obtained after applying the global transport operator to the emitted radiance function. The set of inner products is done against a set of basis functions. We call this set the *observer*. This

formulation is used to introduce a number of algorithms which have been previously designed for realistic rendering. These algorithms are classified according to the basic method the used to obtain the discrete image. This allows to highlight both the differences and similarities between them.

In chapter three we focus on existing Monte-Carlo methods for Global Illumination. We obtain a characterization of the variance of all those algorithms, by using a general formulation. This is done by introducing an appropriate probability measure on the space of all chains, and defining a random variable with that measure. A interesting result is that the variance function obeys a second order integral equation. Finally, we apply that result to path tracing for radiance, particle tracing for radiosity, and gathering random walk for radiosity. In all the cases we obtain expressions for the variance, and those expressions involve just the potential function and the unknown radiance function. Related characterizations of the variance where previously known [Mikhailov92], but did not included finite chains ended in absorption. We also introduce ideal estimators with zero variance.

The main contribution of chapter four is the demonstration of importance sampling feasibility for high quality rendering of radiosity environments by using two-pass algorithms. Importance sampling is based on the result obtained in previous chapter about the ideal, zero-variance sampling density. This sampling density can be approached by using information obtained in the first pass. The efficiency of the sampling process is proved when compared to standard path-tracing techniques.

Finally, in last chapter we describe an algorithm which extends the one presented in previous chapter to glossy environments. Here, we use an already existing method for the creation of cluster-patches hierarchies, but applied to Monte-Carlo techniques, instead of finite-elements methods. We propose a new method for the creation of a hierarchy of links by using particle tracing in the first pass. Later, this hierarchy is used for importance sampling in the second pass. Moreover, we show how a recursive algorithm can be used to get samples from the link structure, by a traversal algorithm which runs over the cluster-path tree and select leaf nodes.

As an abstract of the more important contributions, we include here a brief list with each of those:

- A new expression for the rendering equation, which uses a two-variable kernel.
- A classification of a huge set of Global Illumination methods.
- A characterization of the variance for a wide set of Monte Carlo algorithms for Global Illumination.
- A new algorithm with reduced variance for Monte Carlo final gather in diffuse environments.
- A new two-pass Monte Carlo algorithm for Global Illumination, including clustering and importance sampling for final-gather.

Bibliography

[Alpert93] Alpert, B.K.: A Class of Bases in L^2 for the Sparse Representation of Integral Operators. *SIAM Journal of Mathematical Analysis,* 24(1):246-262, January 1993.

[Arvo86] Arvo, J.: Backward Ray-Tracing. *Siggraph'86 Developments in Ray Tracing, Seminar Notes,* Agosto 1986.

[Arvo90] Arvo, J., Kirk, D.: Particle Transport and Image Synthesis *Computer Graphics,* 24(4):63-66, August 1990. ACM Siggraph '90 Conference Proceedings.

[Arvo95] Arvo, J.: Stratified Sampling of Spherical Triangles. *Computer Graphics Proceedings, Annual Conference Series, 1995* pp 437-438

[Arvo95a] Arvo, J.: The Role of Functional Analysis in Global Illumination. *Sixth Eurographics Workshop on Rendering*. Dublin, June, 1995.

[Arvo95b] Arvo, J.: Stratified Sampling of Spherical Triangles. *Computer Graphics Proceedings, Annual Conference Series, 1995* pp 437-438

[Aupperle93] Aupperle, L., Hanrahan, P.: Importance and Discrete Three Point Transport. *Fourth Eurographics Workshop on Rendering*. 1993.

[Baum89] Baum, D.R., Rushmeier, H.E., Winget, J.M.: Improving Radiosity Solutions Through the Use of Analitically Determined Form-Factors. *Computer Graphics,* 23(3):325-334, July 1989. ACM Siggraph '89 Conference Proceedings.

[Beyer94] Beyer, M., Lange, B.: Rayvolution: An Evolutionary Ray Tracing Algorithm. *Proceedings of 5th EG Workshop on Rendering*. Darmstadt, June 1994.

[Beylkin91] Beylkin G., Coifman R., Rokhlin V.: Fast Wavelet Transforms and Numerical Algorithms I. *Communications on Pure and Applied Mathematics,* 44:141-183, 1991.

[Buckalew89] Buckalew, C., Fussel, D.: Illumination Networks: Fast Realistic Rendering with General Reflectance Functions. *Computer Graphics,* 23(3):89-98, 1989. ACM Siggraph'89 Conference Proceedings.

[Christensen93] Christensen, P.H., Salesin, D.H., DeRose, T.D.: A Continuous Adjoint Formulation for Radiance Transport. *Fourth Eurographics Workshop on Rendering*. 1993.

[Christensen94] Christensen, P.H., Stollnitz, E.J., Salesin, D.H., DeRose, T.D.: Wavelet Radiance. In *Proceeding of the Fifth Eurographics Workshop on Rendering*. Darmstadt, Junio 1994.

[Christensen97] Christensen, P.H., Lischinski, D., Stollnitz, E.J., Salesin: Clustering for Glossy Global Illumination. *ACM Transactions on Graphics*, Vol.16, n.1, January,1997, pp 3-33.

[Cohen85] Cohen, M.F. Greenberg, D.P.: A Radiosity Solution for Complex Environments. *Computer Graphics,* 19(3): 31-40, 1985. ACM Siggraph'85 Conference Proceedings.

[Cohen88] Cohen, M.F., Chen, S.E., Wallace, J.R., Greenberg, D.P.: A Progressive Refinement Approach to Fast Radiosity Image Generation. *Computer Graphics,* 22(4):75-84, 1988. ACM Siggraph'88 Conference Proceedings.

[Cohen93] Cohen, M.F. Wallace J.R.: *Radiosity and Realistic Image Synthesis* Academic Press Professional, 1993.

[Cook84] Cook, R.: Distributed Ray-Tracing. *Computer Graphics,* 18(3): pp.137-146, 1984. ACM Siggraph'84 Conference Proceedings.

[Dutre94] Dutré, P., Willems, Y.D.: Importance-driven Monte Carlo Light Tracing *Fifth Eurographics Workshop on Rendering*. Darmstadt, 1994.

[Glassner95] Glassner, A.S.: Principles of Digital Image Synthesis. Morgan Kaufmann Publishers, 1995.

[Goral84] Goral, C.M., Torrance, K.E., Greenberg, D.P., Bennet, B.: Modelling the Interaction of Light Between Diffuse Surfaces. *Computer Graphics,* 18(3): 213-222, 1984. ACM Siggraph'84 Conference Proceedings.

[Gortler93] Gortler, S., Schroder, P., Cohen, M.F., Hanrahan, P.: Wavelet Radiosity. *Computer Graphics,* ACM Siggraph'93 Conference Proceedings.

[Gortler94] Gortler, S., Cohen, M.F., Slusallek, P.: Radiosity and Relaxation Methods (Progressive Refinement is Southwell Relaxation). *IEEE Computer Graphics & Aplications* Noviembre'94.

[Hanrahan91] Hanrahan, P., Salzman, D., Aupperle, L.: A Rapid Hierarchical Radiosity Algorithm. *Computer Graphics,* 25(4), July 1991. ACM Siggraph '91 Conference Proceedings.

[Heckbert90] Heckbert, P.S.: Adaptive Radiosity Textures for Bidirectional Ray Tracing. *Computer Graphics,* 24(4), August 1990. ACM Siggraph '90 Conference Proceedings.

[Immel86] Immel, D.S., Cohen M.F., Greenberg, D.P.: A Radiosity Method for Non-Diffuse Environments. *Computer Graphics,* 20(4):133-142, 1986. ACM Siggraph'86 Conference Proceedings.

[Ito93] Itô, K., Ed.: *Encyclopedic Dictionary of Mathematics (two volumes)*. MIT Press, Cambridge, 1993.

[Jensen95] Jensen H.W.: Importance Driven Path Tracing using the Photon Map. *Proceedings of the 6th EG Workshop on Rendering,* pp 349-368. Dublin, June 12-14, 1995.

[Kajiya86] Kajiya, J.T.: The Rendering Equation. *Computer Graphics,* 20(4):143-150, 1986. ACM Siggraph'86 Conference Proceedings.

[Kalos86] Kalos, M.H., Whitlock, P.A.: Monte Carlo Methods. Volume I: Basics Jhon Wiley & Sons, 1986.

[Kok91] Kok, A.J.F., Jansen F.W.: Source Selection for the direct Lighting Computation in Global Illumination. *Proceedings of the 2nd EG Workshop on Rendering,* Barcelona 1991.

[Lafortune94] Lafortune, E.P., Willems, Y.D.: A Theoretical Framework for Physically Based Rendering. *Computer Graphics Forum 13(2)*, June 1994.

[Lafortune94a] Lafortune, E.P., Willems, Y.: Using the Modified Phong Reflectance Model for Phisically Based Rendering. Report CW 197, November, 1994. Departament of Computing Science, K.U. Leuven.

[Lafortune97] Lafortune E.P., Foo S., Torrance K.E., Greenberg D.P.: Non-Linear Approximation of Reflectance Functions. SIGGRAPH 97 Conference Proceedings, Los Angeles, California, August 1997.

[Languenou89] Languénou, E., Bouatouch, K., Tellier, P.: An adaptive Discretization Method for Radiosity. *Computer Graphics Forum 11(3)*, September 1992.

[Lewis93] Lewis, R.: Making Shaders more physically plausible. Proceedings of the Fourth EG Workshop on Rendering. Paris, France, 14-16 June, 1993. Also in *Computer Graphics Forum 13(2)*, pp. 109-120, June, 1994.

[Lischinski92] Lischinski, D., Tampieri, F., Greenberg D.P.: Discontinuity Meshing for Accurate Radiosity. *IEEE Computer Graphics & Applications,* November, 1992.

[Mallat89] Mallat, S.G.: A Theory for Multiresolution Signal Decomposition: The Wavelet Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2(7):674-693, July 1989.

[Malley88] Malley, T.J.: A Shading Method for Computer Generated Images. Master's Thesis, Dept. of Computer Science, University of Utah, June 1988.

[Mikhailov92] Mikhailov, G.A.: *Optimization of Weighted Monte Carlo Methods*. Springer Series in Computational Physics. Springer Verlag, 1992.

[Pattanaik92] Pattanaik, S.N., Mudur, S.P.: Computation of Global Illumination By Monte Carlo Simulation of The Particle Model of Light In *Proceedings of the Third Eurographics Workshop on Rendering*. Consolidation Express, Bristol, 1992.

[Pattanaik93] Pattanaik, S.N., Mudur, S.P.: Efficient Potential Equation Solution for Global Illumination Computation. *Computer & Graphics*. 17(4):387-396, 1993.

[Pattanaik94] Pattanaik, S., Bouatouch, K.: Haar Wavelet: A Solution to Global Illumination with General Surface Properties. In *Proceeding of the Fifth Eurographics Workshop on Rendering*. Darmstadt, Junio 1994.

[Phong75] Phong, B.: Illumination for computer generated images. *Comunications of the ACM*. 18(6):311-317, 1975.

[Reichert92] Reichert, M.C.: *A two pass radiosity method driven by lights and viewer position*. Master's thesis, Program of Computer Graphics, Cornell University, Jan. 1992.

[Rubinstein81] Rubinstein, R.Y.: *Simulation and the Monte Carlo Method*. Jhon Wiley & Sons, 1981.

[Rushmeier88] Rushmeier, H.E.: *Realistic Image Synthesis for Scenes with Radiatively Participating Media*. PhD thesis, Program of Computer Graphics, Cornell University, 1988.

[Rushmeier93] Rushmeier, H., Patterson, C., Veerasamy, A.: Geometric Simplification for Indirect Illumination Calculations. *Proceeding of Graphics Interface 93*, Canadian Information Processing Society, 1993.

[Sbert93] Sbert, M.: An Integral Geometry Based Method for Fast Form Factor Computation. *Computer Graphics Forum*. 12(3):409-420, 1993. Eurographics '93 Conference Proceedings, Barcelona, September 1997.

[Sbert97] Sbert, M.: Error and Complexity of Random-Walk Monte-Carlo Radiosity. *IEEE Transactions on Visualization and Computer Graphics* 3(1), March 1997.

[Sbert97b] Sbert, M.: Optimal Source Selection in Shooting Random Walk Monte Carlo Radiosity. *Computer Graphics Forum*. 16(3), 1997. Eurographics '97 Conference Proceedings, Budapest, September 1997.

[Schroder93] Schroder, P., Gortler, S.J, Cohen, M.F.: Wavelet Projections for Radiosity. *Proceedings of the Fourth Eurographics Workshop on Rendering*, Paris, Junio 1993.

[Schlick94] Schlick, C.: A Survey of Shading and Reflectance Models. *Computer Graphics Forum 13(2)*. pp 121-132. Blackwell, 1994.

[Schroeder94] Schroeder, P., Hanrahan, P.: Wavelets Methods for Radiance Computations. *Proceedings of the Fifth Eurographics Workshop on Rendering*, Darmstadt, Junio 1993.

[Shao88] Shao, M., Peng, Q., Liang, Y.: A New Radiosity Approach by Procedural Refinements for Realistics Image Synthesis. *Computer Graphics,* 22(4): 93-101, 1988. ACM Siggraph'88 Conference Proceedings.

[Shirley90] Shirley P.: A Ray Tracing Method for Illumination Calculation in Diffuse-Specular Scenes. *Proceedings of Graphics Interface 90,* Canadian Information Processing Society, 1990.

[Shirley91] Shirley P.: Time Complexity of Monte-Carlo Radiosity. *Eurographics'91 Conference Proceedings*. pp 459-465, September, 1991.

[Shirley92] Shirley, P.S., Wang, C.: Distribution Ray-Tracing: Theory and Practice. In *Proceedings of the Third Eurographics Workshop On Rendering*. Consolidation Express, Bristol, 1992.

[Shirley96] Shirley, P.S., Wang, C., Zimmerman, K.: Monte Carlo Techniques for Direct Lighting Calculations. *ACM Transactions on Graphics, 15(1)*, January 1996. pp 1-36.

[Sillion89] Sillion, F., Puech, C.: A general Two-Pass Method Integrating Specular and Diffuse Reflection. *Computer Graphics,* 23:(3):335-344, 1989. ACM'Siggraph'89 Conference Proceedings.

[Sillion91] Sillion, F., Arvo, J.R., Westin S.H., Greenberg, D.P.: A global Illumination Solution for General Reflectance Distributions. *Computer Graphics,* 25(4):187-196, Julio 991. ACM Siggraph'91 Conference Proceedings.

[Smits92] Smits, B.E., Arvo, J.R., Salesin, D.H.: An Importance-Driven Radiosity Algorithm. *Computer Graphics,* 25(4), July 1991. ACM Siggraph '91 Conference Proceedings.

[Spanier69] Spanier, J., Gelbard, E.M.: *Monte Carlo Principles and Neutron Transport Problems*. Addison-Wesley 1969.

[Urena92] Ureña, C., Parets, J., Torres, J.C., del Sol, V.: An Object-Oriented approach to Ray Tracing Image Synthesis Algorithm Implementation. *Computer & Graphics 16(4)*. pp. 363-369. Pergamon Press, 1992.

[Urena93] Ureña, C., Torres, J.C.: Un mtodo de Monte Carlo Bidireccional para Sntesis de Imgenes. *Actas del III Congreso Español de Informática Gráfica (CEIG'93)*. pp. 209-222. Granada, 1993.

[Urena97] Ureña, C., Torres, J.C.: A Formalization and Classification of Global Illumination Methods. *Computer & Graphics 21(2)*. Pergamon Press, 1997.

[Urena97a] Ureña, C., Torres, J.C.: Improved Irradiance Computation by Importance Sampling. *Rendering Techniques'97* . Springer Verlag, Wien, 1997.

[Urena97b] Ureña C., Torres, J.C., Revelles, J., Cano, P., del Sol, V., Cabrera, M.: Designing an Object-Oriented Rendering System. *7th Eurographics Workshop on Programming Paradigms in Graphics*, Budapest, Hungary, September, 1997.

[Veach94] Veach, E., Guibas L.J.: Bidirectional Estimators for Light Transport. *Fifth Eurographics Workshop on Rendering*, Darmstadt, 1994.

[Veach95] Veach, E., Guibas L.J.: Optimally Combining Sampling Techniques for Monte Carlo Rendering. *Computer Graphics* Proceedings, Annual Conference Series, 1995. pp. 419-428, 1995. ACM Siggraph'95 Conference Proceedings.

[Veach97] Veach, E.: *Robust Monte Carlo Methods for Light Transport Simulation*. Ph.D. Thesis. Standford University. December, 1997.

[Wallace87] Wallace, J.R., Cohen, M.F., Greenberg, D.P. A Two Pass Solution to the Rendering Equation: A Synthesis of Ray-Tracing and Radiosity Methods. *Computer Graphics,* 21(4):311-320, July 1987. ACM Siggraph'87 Conference Proceedings.

[Wallace89] Wallace, J.R., Elmquist, K.A., Haines, E.A.: A Ray Tracing Algorithm for Progressive Radiosity. *Computer Graphics,* 23(3):315-324, 1989. ACM Siggraph'89 Conference Proceedings.

[Ward88] Ward, G.J., Rubinstein, F.M., Clear, R.D.: A Ray Tracing Solution for Diffuse Interreflection. *Computer Graphics,* 22(4):85-92, August 1988. ACM Siggraph'88 Conference Proceedings.

[Ward91] Ward, G.J.: Adaptive Shadow Testing for Ray-Tracing *Proceedings of the 2nd EG Workshop on Rendering.*, Barcelona, 1991.

[Whitted80] Whitted, T., An Improved Illumination Model for Shaded Display. *Communications of The ACM*, 23(6):343-349, 1980.

[Yu95] Yu, Y., Peng,Q.: Multiresolution B-Spline Radiosity. *Computer Graphics Forum*. 14(3):285-298, 1995. Eurographics'95 Conference Proceedings.

[Zimmerman95] A Two-Pass Solution to the Rendering Equation with a Source Visibility Preprocess. *Proceedings of the 6th EG Workshop on Rendering*. Dublin, 1995.

[Zatz93] Zatz, H.R.: Galerkin Radiosity. A Higher Order Solution Method for Global Illumination. *Computer Graphics Proceedings, Annual Conference Series*, 1993.