# Newtonian Global Illumination
# Algorithmic Solution, Implementation and Parallelization

Michal Jančok

2007

# Newtonian Global Illumination
# Algorithmic Solution, Implementation and Parallelization

Diplomová práca

Autor: Michal Jančok

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky
Katedra informatiky

Diplomový vedúci: Dr. Tomáš Plachetka

Bratislava, 2007

Názov práce

Newtonian Global Illumination, Algorithmic Solution, Implementation and Parallelization

Cieľ práce

To design and implement an algorithm that solves the Global Illumination Problem based on newtonian optics as defined by the rendering equation. To develop a parallel version of the algorithm above.

Týmto prehlasujem, že som diplomovú prácu vypracoval samostatne za pomoci konzultácií a s použitím uvedenej literatúry.

Bratislava, 2007 .....

Michal Jančok

Chcel by som sa poďakovať svojmu diplomovému vedúcemu Dr. Tomášovi Plachetkovi za jeho cenné rady a pripomineky, ktoré mi veľmi pomohli pri písaní tejto práce.

# ABSTRAKT

Po porovnaní reálneho osvetlenia s osvetlením vypočítaným súčasnými metódami globálnej iluminácie sa objavuje viacero otázok ohľadom správnosti týchto metód. Pri snahe o optimalizáciu za účelom získania krajších obrázkov často obetujeme fyzikálnu a matematickú správnosť týchto metód.

Správnosť metódy, pri ktorej je nutné nastavovať niekoľko desiatok parametrov, je diskutabilná. Hlavne ak je potrebné, aby používateľ nastavoval príslušné parametre pre každú scénu zvlášť, kým nezíska dostatoćne *reálne* výsledky. V dôsledku toho získavame viacero verzií zobrazenia reality a je tažké rozhodnúť, ktoré je to správne, t.j. najbližšie k realite.

Obrázky, ktoré nie je možné rozoznať od fotografií nazývame fotorealistické. Kvalita metód globálnej iluminácie sa posudzuje často práve podľa toho ako veľmi fotorealistické ňou generované obrázky sú. Rozhoduje o tom človek. Je však vôbec možné porovnávať algoritmy takouto subjektívnou metódou?

V tejto diplomovej práci sa pokúsime vyhnúť týmto a aj iným problémom tým, že sme zvolili fyzikálny prístup k riešeniu globálnej iluminácie. Predstavíme matematickú definíciu a algoritmické riešenie tohto problému. Zhodnotíme súčasné algoritmy na riešenie globálnej iluminácie. Na záver predstavíme návrh, implementáciu a vylepšenia algoritmu, ktorý rieši problém globálnej iluminácie v zmysle newtonovskej optiky.

kľúčové slová: global illumination, monte-carlo methods, rendering

# ABSTRACT

Comparison of real lighting with the lighting computed by current global illumination methods brings up many questions concerning the correctness of these methods. While optimizing the methods and making the output images look nicer, physical and mathematical plausibility is often sacrificed.

Correctness of a method which uses several dozens of parameters is disputable, especially when the user has to set the parameters for every scene until the image looks real to him. As a result, there are more versions of *reality* and it is difficult to judge which one is the most real.

Images which cannot be distinguished from real images (photographs) are called photo-realistic. Global illumination methods are often compared by the quality of the output images and how much *photo-realistic* they are. The judge is a human operator. But can we really compare algorithms by such a subjective measure?

In this thesis we try to avoid these and other problems by taking a physically-based approach to global illumination. We give a mathematical definition and propose algorithmic solutions to the problem. Then we discuss the state-of-the art global illumination algorithms. Finally, we present design, implementation and improvements of an algorithm which solves the global illumination problem in newtonian sense.

keywords: global illumination, monte-carlo methods, rendering

# CONTENTS

# 1. GLOBAL ILLUMINATION

Computer graphics allows us to display various data with accuracy never known before. It has become very affordable to visualize and model almost anything—the complexity of the models is limited only by the capability of computers. One of the central tasks which computer graphics has addressed from its very beginning was simulation of the real world. The simulations range from the microscopic world of atoms in chemistry and particle physics to models of buildings for architects and visual effects for the movie industry. The goal is to create realistic images, so real that they would not be distinguishable from the reality by the human eye. Two approaches have been taken to solve this task. The first one can be seen mainly in modern computer games. The techniques used there have usually nothing in common with the reality. They focus on the final picture: how fast it can be created and if it looks *good* enough. In this thesis, we will focus on the second approach which attempts to create images that are based on the physical reality of the scenes we want to visualize. We will discuss the actual goals of this approach, methods, algorithms and how the state-of-art solutions solve this task. Finally we will propose our own algorithm.

## *1.1 Definition of the Global Illumination Problem*

An informal definition of the task is simple. We have a three-dimensional scene consisting of objects and light sources. Usually light sources are also objects that have the ability to emit light. Each object has its set of features such as material and geometrical shape. When light arrives at the object its trajectory, intensity, color and other properties change. The task at hand is to calculate what an observer positioned somewhere in the scene sees.

Computer graphics began with simple models, where from the algorithmic point of view every object was considered a light source. These models only allowed for computation of direct illumination. In other words, they only decided whether there is no obstacle along the path from point of emission toward the sensor (Figure 1.1).

As the hardware evolved, simple interactions between a light source, an object and a sensor could be calculated. This approach is called Local Illumination as it only deals with local changes of light at an object and does not take into account reflection and refraction of light by objects in the scene (Figure 1.2).

An approach that takes into account all possible light-object interactions is called Global

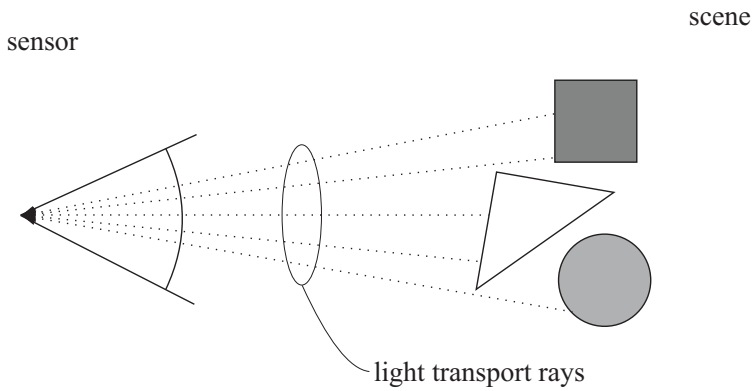


*Fig. 1.1:* Direct Illumination

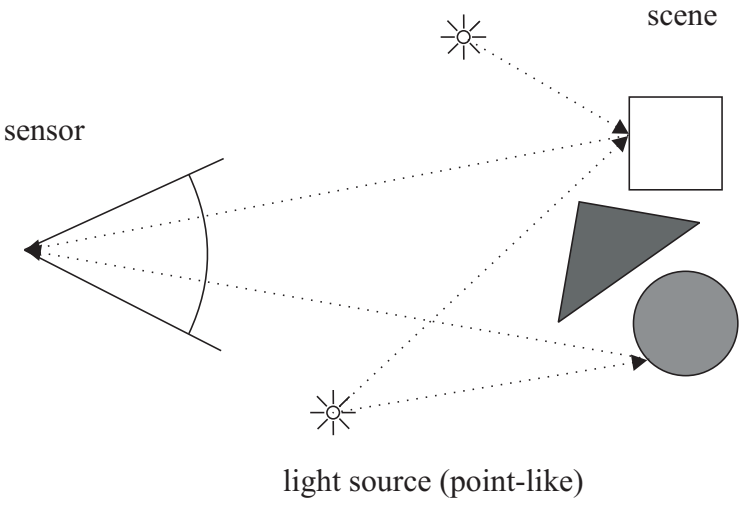


*Fig. 1.2:* Local Illumination

Illumination (Figure 1.3).

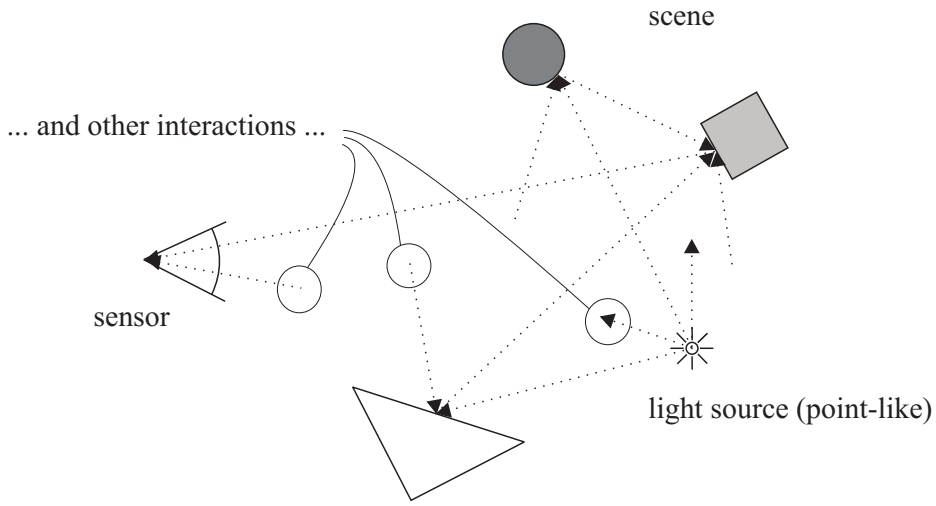


*Fig. 1.3:* Global Illumination

Extensive work has been done to solve Global Illumination. As a result large number of algorithms was created. Among the most significant we can name the algorithms of Ray Tracing [Gla89], Radiosity [GCT86], Path Tracing [Kaj86], Bi-Directional Path Tracing [LW93] and Photon Mapping [Jen01]. Each of them takes a slightly different approach toward solving the Global Illumination problem. Currently the most successful algorithm in terms of calculating nice images, able to visualize a large variety of phenomena light-object interaction in a reasonable time, is considered to be the Photon Mapping algorithm. In the last years a lot of effort has been put in improvement of this algorithm. The improvements were in areas such as output image quality: [SSK], [KW00], computing speed: [GT05], [Chr99], [GWS04] and others. We will discuss the details of the most important solutions in a special chapter later.

## *1.2 Solving the Global Illumination Problem*

An algorithm solving the Global Illumination Problem must provide an output generated by considering all possible interactions of the light and the scene according to the laws of physics. Unfortunately, due to the complexity and continuity of the real world this is not yet possible in reasonable time. Therefore simplifications are necessary. The only assumption we make is the assumption of newtonian optics. Newtonian optics is described by a Fredholm integral equation of the second kind. We solve this equation directly and obtain its exact solutions in the limit. Many rendering algorithms make additional simplifications, while unjustly claiming to give solution to the Global Illumination Problem.

A common criterion is needed to assess the success of these algorithms. Therefore a very

convenient test from the field of photorealism was adopted. The successful algorithm must generate an output that is indistinguishable from a photograph. This is decided by a human operator. We can see that this criterion is hardly measurable and is very subjective. We can argue about the choice of this *representative* arbiter and his ability to assess for example an artificial scene of an alien planet of which no other photograph exists. The problem with a human arbiter is the fact that people do not judge the quality on the basis of what reality is but what they *believe* reality is. In our opinion a subjective judgment is unsuitable for judging the quality of algorithms solving the Global Illumination problem unless the task of Global Illumination is to generate pieces of art. Generally art is not he only application of computer graphics.

In this thesis we deal with the construction of a Global Illumination algorithm that does not need an arbiter. An algorithm that solves the Global Illumination either solves or does not solve the underlying physical model. The visual quality is then based on this model; the choice of the physical model determines which phenomena we can describe and which we can visualize. Our ambition is to propose a stochastic but unbiased algorithm that is 100% correct in terms of implementing the theory it is based upon and is able to provide (even if only mathematically) the complete solution of the Global Illumination Problem.

As we have shown in the definition of the Global Illumination Problem, the problem itself consists of several parts that can be addressed more or less independently. The parts we will address are

1. *Light Transport Model*
2. *Material and Geometry Model*
3. *Sensory Model*

We will begin with the choice of the Light Transport Model.

# 2. THE LIGHT TRANSPORT MODEL

To measure the quality of any algorithmic solution to the Global Illumination Problem we want to evaluate if the algorithm is a mathematical solution to a clearly defined task. Thus we need a mathematical description of the problem.

## *2.1 Physics of Light*

In order to mathematically describe a natural phenomena such as light we need to understand its physical nature. The nature of light has been studied for centuries and there is probably still a lot to learn and to prove. We will present three most widely accepted theories of light.

### *2.1.1 Classical Optics*

The theory of Classical Optics is based on the works of Sir Isaac Newton. He was one of the first who took a scientific approach to study nature of light. His theory is based on simple experiments and observations of the light phenomena.

> By the Rays of Light I understand its least Parts and those as well Successive in the same lines and Contemporary in several lines. [New21]

Newton thought of the light as a flow of tiny particles—rays, traveling in straight lines at infinite speed. The light can be either compound—composed of particles of different *colors* or homogeneous—composed of particles of the same *color*. When hitting an object the rays either refract or reflect according to the object's material properties, their *color* and the Laws of Reflection and Refraction [Str79]. This simple theory is able to explain most basic phenomena of light such as reflection and refraction. However, Newton was not able to describe what exactly happens when light diffracts or interacts with thin transparent bodies. Therefore a different approach was needed.

### *2.1.2 Wave Optics*

The Wave Theory of light emerged at almost the same time as Classical Optics but did not become very popular at first. It was studied later mainly by Ch. Huygens, T. Young and

A.J. Fresnel who have shown its superiority over classical optics.

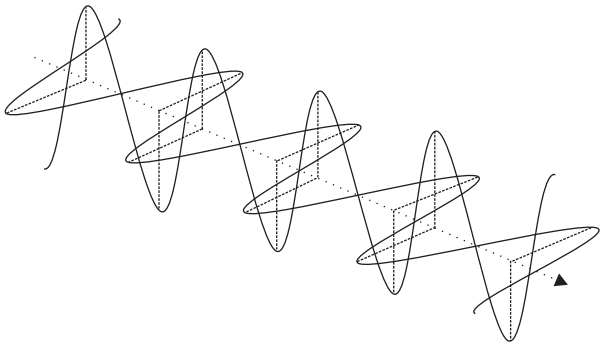

*Fig. 2.1:* Electromagnetic Wave

According to the Wave Theory the light propagates in form of electromagnetic waves (Figure 2.1) emitted by atoms at very short intervals. It travels in lines at finite speeds dependent on the characteristics of the transfer medium. The propagation of light can be described by J.C. Maxwell's equations. As for every electromagnetic wave, light experiences the effects of diffraction and interference under special conditions. Thanks to this theory, the physicists were finally able to explain phenomena such as rainbow-like colors of oil on water surfaces (colors of thin transparent bodies) or interference images when light passes through thin crevices (diffraction).

### *2.1.3 Quantum Electrodynamics*

The QED is considered to be the most complete and accurate theory that explains the phenomena of light. It was created in 1940s by R. Feynman, J. Swinger and S. Tomonaga.

It is a rather complex theory therefore we will not go into much detail. It provides an answer to the question how it is possible that light emitted at point $A$ knows how to travel to get to $B$ over the shortest path possible—a straight line. The theory says that the light actually takes every possible path at every possible speed in every possible time (light can even travel back in time in this theory). The observer at $B$ sees the result when all these paths sum up. By interference the least probable paths cancel each other out and what remains is the most probable path the light could take—a straight line. This theory states that it cannot explain what light is but it can explain how it gets somewhere. It has achieved a remarkable accuracy in its predictions when compared to experimentally measured data. So far it has been able to explain all known phenomena of light.

## 2.2 Newtonian Global Illumination

Having these theories, if we were able to devise an algorithm based on QED we would get the best results in terms of physical reality and plausibility of the output picture. Our algorithm would be able to visualize all phenomena of light-object interaction. Unfortunately, QED-based algorithms would be too complex to provide observable results in reasonable computational time. The same holds for the Wave Theory. In the case of QED we would have to compute the probability of the light energy being at any possible time and space of the scene thus trying to solve an equation roughly similar to

$$L(x,t) = \int_{Space} \int_{Time} L(x',t')dx'dt'$$

A solution based on the Wave Theory would require us to cope with the phase of light and the interference of the light waves. A technique would be needed to detect edges and tiny crevices near the intersection points of light waves and objects and to compute the outcome of such events.

For these reasons the methods used in Global Illumination are based on the Classical Optics theory of light—Newtonian Optics. Although it is rather simple, it is able to describe most of the observable phenomena of light-object interactions. The cases where it fails usually happen under special conditions. Therefore the level of reality it provides is very close to what really happens. We have a very good mathematical model that covers this theory and thanks to its simplicity we are able to solve it. Yet its implementation is not as straightforward as it might seem.

## 2.3 The Rendering Equation

Solving the Global Illumination Problem on a computer requires a mathematical model. Such a model is provided by the rendering equation. The rendering equation is a mathematical model of light transport based on Photometry. Photometry studies physical properties of light that measure the quantitative influence of light radiation on the human eye. Photometry and therefore also the rendering equation does not consider the wave properties of light and assumes that light consists of rays as defined in Classical Optics. Thus we can consider the rendering equation to be a mathematical representation of the Classical Optics and when we want to solve the problem of Newtonian Global Illumination we just need to solve the rendering equation. For the rest of the paper we will consider monochromatic light. We also consider only opaque materials. All these simplifications can be removed by simple changes in the equations and the corresponding algorithms.

We will now deal with the emission of the light energy off a surface. We will need to define the direction of emission by the spherical coordinates.

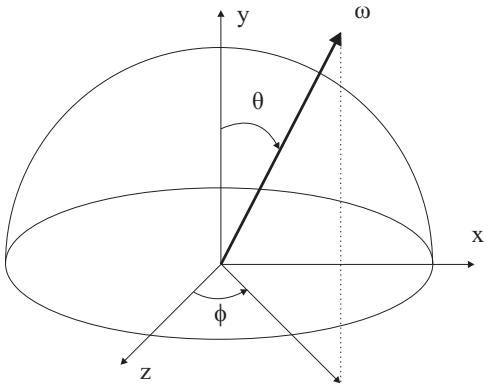


*Fig. 2.2:* Spherical coordinates

A direction $\omega$ is defined by two angles $\theta$, $\phi$. $\theta$ is the angle between the direction and the $y$-axis and $\phi$ is the angle between the projection of the direction into the $x, z$ plane and the $z$-axis (Figure 2.2).

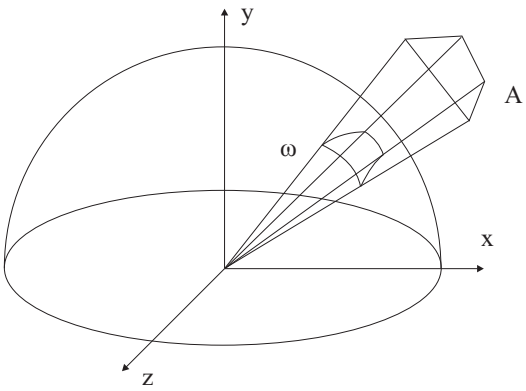


*Fig. 2.3:* Solid angle

A set of directions can be defined by a solid angle. A solid angle is defined by a cone or a pyramid with its size related to the intersection of the respective cone/pyramid and a unit sphere centered at the apex of the solid angle (Figure 2.3).

A differential solid angle can be defined as $d\omega$ where $\omega$ is the direction of the differential set. Therefore we can express the differential solid angle using the $\theta$ and $\phi$ spherical coordinates (Figure 2.4). Let us describe the directions of the differential set as $d\theta$ and $d\phi$. The differential rectangle then has $d\theta$ vertical and $\sin\theta \cdot d\phi$ horizontal sizes and so the size of the solid angle is

$$d\omega = \sin\theta \cdot d\phi d\theta \tag{2.1}$$

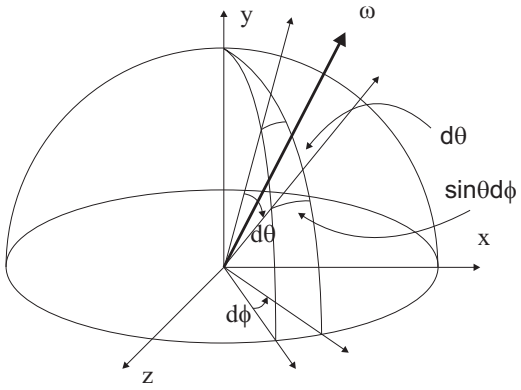


*Fig. 2.4:* Differential solid angle

Finally, the solid angle that corresponds to the differential surface $dA$ as seen from point $p$ is

$$d\omega = \frac{dA \cdot \cos\theta}{r^2} \tag{2.2}$$

where $\theta$ is the angle between the normal vector of $dA$ and the directional vector from $dA$ to $p$ and $r$ is the distance between $dA$ and $p$. Basically, it is the size of the differential rectangle, that we obtain by projecting the infinitesimal area $dA$ along the directional vector from $dA$ to $p$ onto the unit sphere centered in $p$.

The light power (or flux) $\Phi$ is the energy radiated through a boundary. This definition is not very useful as we need to know the boundary. But if we assume that the boundary is infinitesimally small we can take into account only one surface point and one direction. The measure we get this way is called radiance (or intensity).

The radiance $L(x, \omega)$ is the differential light flux $\Phi(\vec{x}, dA, \omega, d\omega)$ that leaves a differential area $dA$ around $x$ in a differential solid angle $d\omega$ around $\omega$ per the projected area $dA$ and the size of $d\omega$. If the angle between the surface normal of $dA$ and the direction of interest is $\theta$, then the projected area is $dA \cdot \cos\theta$ and therefore the radiance is

$$L(x, \omega) = \frac{d\Phi(x, dA, \omega, d\omega)}{dA \cdot d\omega \cdot \cos\theta} \tag{2.3}$$

Now we are able to describe the light energy being emitted and being absorbed by surfaces. We are ready to define the rendering equation. We begin with a simple case of two differential small surfaces in 3D. Surface $dA$ emits light energy and the surface $dA'$ absorbs it (Figure 2.5). By the definition of radiance (2.3), if $dA'$ is visible from $dA$ in solid angle $d\omega$ and the radiance of $dA$ is $L$ in this direction the flux emitted by $dA$ and absorbed by $dA'$ is

$$d\Phi = L \cdot dA \cdot d\omega \cdot \cos\theta \tag{2.4}$$

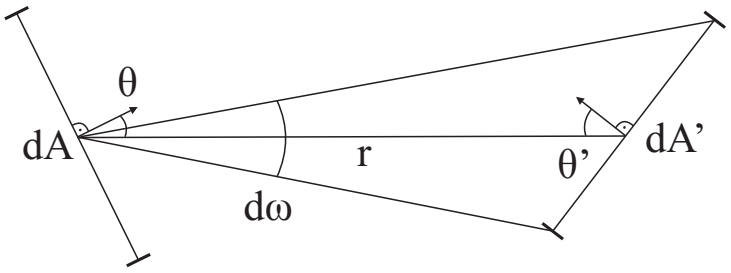


*Fig. 2.5:* Energy exchanged between two infinitesimal surfaces

By (2.2), we can express the solid angle from the projected area of $dA'$ and we get

$$d\Phi = L \cdot \frac{dA \cdot \cos\theta \cdot dA' \cdot \cos\theta'}{r^2} \tag{2.5}$$

We can use the equation (2.2) again now for the emitting surface

$$d\Phi = L \cdot \frac{dA \cdot \cos\theta \cdot dA' \cdot \cos\theta'}{r^2} = L \cdot dA' \cdot d\omega' \cdot \cos\theta' \tag{2.6}$$

So as we can see the same equation holds for the surface that emits the energy as for the surface that receives it.

Upon hitting the surface the light will interact with it. The light can be reflected, refracted or absorbed (refraction can be regarded as a special case of reflection). It can also be reflected not according directly to the law of perfect specular reflection, which holds for mirror-like surfaces, as the area $dA$ around $x$ may contain infinitesimal irregularities (we will discuss this in more detail later). So when the light comes from a direction $\omega'$ to a point $x$, the probability it is reflected from the point $x$ into a solid angle $d\omega$ around direction $\omega$ can be expressed as a probability density function $PR \cdot d\omega$. So the reflection for light energy of power $\Phi^{in}(x, dA, \omega', d\omega')$ (that means coming to an area around point $x$ from solid angle $d\omega'$ around direction $\omega'$) is

$$(PR \cdot d\omega) \cdot \Phi^{in}(x, dA, \omega, d\omega') \tag{2.7}$$

For the total reflected energy all incoming directions $\omega'$ must be taken into account therefore we integrate the energy contributions over the hemisphere

$$\int_{\Omega} \Phi^{in}(x, dA, \omega', d\omega') \cdot PR \cdot d\omega \tag{2.8}$$

Sometimes the surface itself emits energy (when it is a light source)

$$\Phi^e(x,\omega) = L^e(x,\omega) \cdot dA \cdot \cos\theta \cdot d\omega \tag{2.9}$$

which must be added to the total outgoing energy

$$\Phi^{out}(x,\omega) = \Phi^e(x,\omega) + \int_\Omega (PR \cdot d\omega) \cdot \Phi^{in}(x, dA, \omega', d\omega') \tag{2.10}$$

We can express both fluxes by radiance using the equations (2.4) and (2.6)

$$\begin{aligned} \Phi^{in}(x,\omega',d\omega') &= L^{in}(x,\omega') \cdot dA \cdot \cos\theta' \cdot d\omega' \\ \Phi^{out}(x,\omega,d\omega) &= L(x,\omega) \cdot dA \cdot \cos\theta \cdot d\omega \end{aligned} \tag{2.11}$$

Removing the term $dA \cdot d\omega \cdot \cos\theta$ and substituting in equation (2.10) we obtain

$$L(x,\omega) = L^e(x,\omega) + \int_\Omega L^{in}(x,\omega') \cdot \cos\theta' \cdot \frac{PR}{\cos\theta} d\omega' \tag{2.12}$$

The term $\frac{PR}{\cos\theta}$ is known as Bi-Directional Distribution Function and defines the optical properties of the material. We can also rewrite the term $L^{in}(x,\omega')$ to $L(y,\omega')$ where $y$ is a point seen from $x$ along direction $-\omega'$. It is often expressed as a result of a visibility function $y = h(x,-\omega')$ which returns the closest surface point from $x$ in the direction $-\omega'$.

The rendering equation is the equation

$$L(x,\omega) = L^e(x,\omega) + \int_\Omega L(y,\omega') \cdot f_r(\omega', x, \omega) \cdot \cos\theta' d\omega' \tag{2.13}$$

## 2.4 Solving the Rendering Equation

We now have the mathematical basis for our algorithm. What remains is to find a way to solve the rendering equation.

### 2.4.1 Louville-Neumann Series

The rendering equation belongs to a family of equations known as Fredholm equations of the 2nd kind. These equations have a common form

$$f(t) = \phi(t) - \lambda \int_a^b K(t,s)\phi(s)ds \tag{2.14}$$

which can be rewritten to

$$\phi(t) = f(t) + \lambda \int_a^b K(t,s)\phi(s)ds \tag{2.15}$$

Solution to this kind of equations is a Louville-Neumann series

$$\phi(x) = \sum_{n=0}^{\infty} \lambda^n \phi_n(x) \tag{2.16}$$

The nth kernel is defined as

$$K_n(x,z) = \int\int\int \cdots \int K(x,y_1)K(y_1,y_2)\cdots K(y_n,z)dxdy_1\cdots dy_n \tag{2.17}$$

For $\phi_n$ holds

$$\phi_n(x) = \int K_n(x,z)f(z)dz \tag{2.18}$$

Now we define the resolvent $K$

$$K(x,z,\lambda) = \sum_{n=0}^{\infty} \lambda^n K_{n+1}(x,z) \tag{2.19}$$

And finally the solution in form of Louville-Neumann series

$$\phi(x) = \int K(x,z,\lambda)f(z)dz \tag{2.20}$$

For the rendering equation (2.13), the solution is

$$
\begin{aligned}
L \quad & (x,\omega) = L^e(x,\omega) + \\
+ \quad & \int L^e(y,\omega') f(\omega',x,\omega) \cos\theta' d\omega' + \\
+ \quad & \int\int L^e(y',\omega'') f(\omega',x,\omega) f(\omega'',y,\omega') \cos\theta' \cos\theta'' d\omega' \omega'' + \\
+ \quad & \int\int\int L^e(y'',\omega''') f(\omega',x,\omega) f(\omega'',y,\omega') f(\omega''',y',\omega'') \cos\theta' \cos\theta'' \cos\theta''' d\omega' d\omega'' d\omega''' + \\
+ \quad & \cdots
\end{aligned} \tag{2.21}
$$

which is an infinite series of integrals. Individual terms correspond to light paths starting at a light source, traveling to $x$ and leaving the object in direction $\omega$. The first term represents the direct illumination—light travels straight to $x$. The second term describes the paths arriving at $x$ after one reflection, the third term represents two reflections and so on, ending with light paths arriving at $x$ after an infinite number of reflections. To find a solution to each of these integrals we will apply the method of Monte-Carlo Integration. But before we can do this, we first need some basic theory from the field of probability.

### 2.4.2 Monte-Carlo Integration

A *random experiment* is a procedure that depends on a random event. The result of a random experiment is a set of elements of an event space $\Omega$. The event space contains *elementary events* from which all random events can be composed.

An *event algebra* $(\mathcal{A}, \Omega)$ consists of an event space and an $\sigma$-Algebra $\mathcal{A}$ over $\Omega$ and contains all possible results of a random experiment. For any $\sigma$-Algebra holds

1. $\Omega \in \mathcal{A}$
2. $\forall B \in \mathcal{A} : \bar{B} \in \mathcal{A}$
3. $B_i \in \mathcal{A}, i \in \mathbb{N} \Rightarrow \bigcup_{i=1}^{\infty} B_i \in \mathcal{A}$

A function $P : \mathcal{A} \to \mathbb{R}$ that assigns each event $A \in \mathcal{A}$ a probability $P(A)$ is a *probability mass function* if

1. $\forall B \in \mathcal{A} : P(B) \geq 0$
2. $P(\Omega) = 1$
3. $A_1 \cdots A_n, P(A_i \cap A_j) = 0, \forall i,j \in \mathbb{N} \Rightarrow P(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$

holds for $P$. The triple $(\mathcal{A}, \Omega, P)$ is a *probability space*. Events $(A_i)_{i=1\dots n}$ are *independent* if $P(\bigcup_{i=1}^{n} A_i) = \sum_{i=1}^{n} P(A_i)$. A *random variable* is a mapping

$$X : (\mathcal{A}_1, \Omega_1, P_1) \to (\mathcal{A}_2, \Omega_2)$$

from a probability space $(\mathcal{A}_1, \Omega_1, P_1)$ into an event algebra $(\mathcal{A}_2, \Omega_2)$ where

$$\forall A \in \mathcal{A}_2 \; \exists X^{-1}(A) \in \mathcal{A}_1$$

That means that $\mathcal{A}_1$ contains the counter image of $A$ in the mapping $X$. Therefore $P^X : \mathcal{A}_2 \to \mathbb{R}$ defined as

$$P^X(B) = P_1(X^{-1}(B)), B \in \mathcal{A}_2 \tag{2.22}$$

is a probability mass function on $(\mathcal{A}_2, \Omega)$.

Two random variables $X, Y$ are called *independent* if $\forall A, B \in \mathcal{A}_2$, $X^{-1}(A)$ and $Y^{-1}(B)$ are in $(\mathcal{A}_1, \Omega_1, P_1)$ independent. A *distribution function* $F_X : \Omega \to \mathbb{R}$ of a random variable is a function

$$F_X(x) = P(X \leq x)$$

Some obvious properties of a distribution function:

1. $\forall x \in \mathcal{R} : F(x) \in \langle 0, 1 \rangle$
2. Distribution functions are strictly increasing
3. $\lim_{x\to\infty} F(x) = 1$
4. $\lim_{x\to-\infty} F(x) = 0$

For *valid* random variables the distribution function $F_X : \mathcal{R} \to \mathcal{R}$ is defined as

$$F_X(x) = \int_{-\infty}^{x} f_X(x)dx \tag{2.23}$$

A random variable X is *valid* if there exists a function $f_X : \mathcal{R} \to \mathcal{R}$ for which holds

$$P(a \leq X \leq b) = \int_{a}^{b} f_X(t)dt \tag{2.24}$$

$f_X$ is called a *probability density function.* The (Figure 2.6) gives an example of a typical $f_X$ and the respective $F_X$.

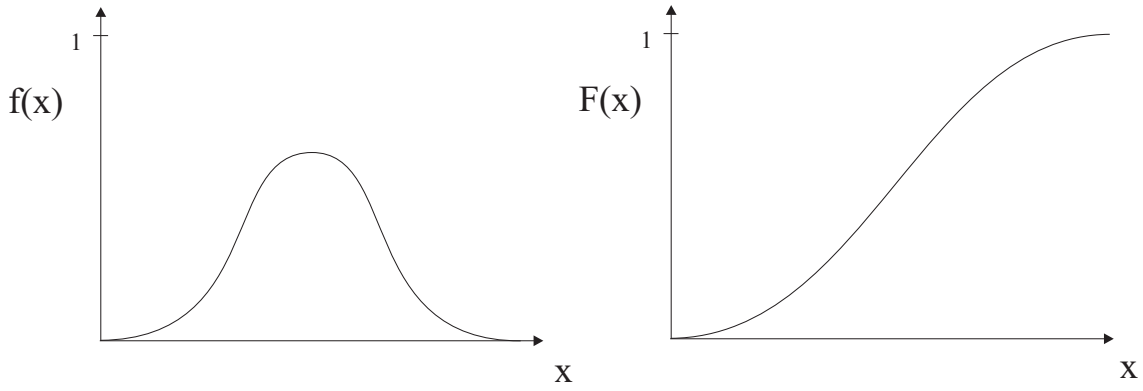


*Fig. 2.6:* Probability density function and the respective distribution function

If $X$ is a valid random variable with probability density function $f$ for which $\int_{-\infty}^{\infty} |x| f(x) dx$ converges. Then

$$E(X) = \int_{-\infty}^{\infty} x f(x) dx$$

is called the *expected value* of random variable X. For $E(X)$ holds

1. $E(x - E(x)) = 0$
2. $E(g \circ X) = \int_{-\infty}^{\infty} g(x) f(x) dx$, if $\int_{-\infty}^{\infty} |g(x)| f(x) dx$ converges
3. $E(aX + b) = aE(X) + b$
4. $E(X + Y) = E(X) + E(Y)$
5. $E(XY) = E(X)E(Y)$, if $X, Y$ are independent random variables

Now we have everything that is necessary to solve an integral equation using stochastic methods. The main idea behind solving the integrals is the *Law of Large Numbers.* For a random variable $X$ holds

$$P\left[E(x) = \lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} x_i\right] = 1$$

where $(x_i)_{i=1\cdots n}$ being a series of random numbers with the same probability density function as $X$. The Law of Large Numbers states that the expected value of a random variable $X$ can be approximated arithmetically by using a series of random numbers generated with the same probability density as the random variable $X$

$$E(X) \approx \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2.25}$$

We will also utilize the fact that when we use a function on a random variable the result is also a random variable. If $X$ has a probability distribution function $f$, for the expected value of $g(x)$ holds

$$E(g(X)) = \int_{y\in\Omega} g(y)f(y)dy$$

Using (2.25)

$$E(g(X)) \approx \frac{1}{n}\sum_{i=1}^{n} g(x_i) \tag{2.26}$$

If we want to determine the value of the integral of function $h$ over $\Omega \in \mathcal{R}$ we can imagine that we can write $h$ as a product of some functions $f$ and $g$

$$h(x) = g(x)f(x) \tag{2.27}$$

where $f$ is a probability density function of a random variable $X : \Lambda \to \Omega$. According to (2.23)

$$\int_{y\in\Omega} f(y)dy = 1$$

Then by (2.26) and (2.27)

$$\int_{y\in\Omega} h(y)dy = \int_{y\in\Omega} g(y)f(y)dy = E(g(x)) \approx \frac{1}{n}\sum_{i=1}^{n} g(x_i) = \frac{1}{n}\sum_{i=1}^{n}\frac{h(x_i)}{f(x_i)}$$

Thus for our integral the approximation will be

$$\int_{y\in\Omega} h(y)dy \approx \frac{1}{n}\sum_{i=1}^{n}\frac{h(x_i)}{f(x_i)} \tag{2.28}$$

Please notice the interesting fact that it is not necessary for $h$ to actually be a product of $g$ and $f$. For the validity of our approximation it is only important that the series $(x_i)_{i=1\cdots n}$ of random numbers has the same probability density as the random variable $X$.

Now we have a mathematical model to describe the light transport in the scene. This model does not state what happens with the light and its properties when it hits an object—it only deals with changes in its directional vector. Neither it tells us how we obtain this intersection point. This will be covered in the following section.

# 3. THE MATERIAL AND GEOMETRY MODEL

Having defined the light transport we now need to describe the material and geometrical properties of the scene somehow. Just like in the case of light transport we will take a look at how it is done by the nature.

The choice of Classical Optics as the theory behind our light transport model has already set the boundaries for the physical behavior of the material and geometry. For Classical Optics the light-object interactions are in fact just intersections of lines and planes. Given any surface we can imagine it as a terrain composed of flat areas. For materials such as mirror, the surface is almost completely flat, with few irregularities, but a surface of a piece of paper is very irregular. When a newtonian ray of light hits an object, at the highest level of details it interacts with a tiny plane and is reflected according to the Law of Reflection. The energy, *color* and other properties of the ray are changed—according to the properties of the material.

If we try to represent this model using computers we encounter several difficulties. All arise from the fact that natural objects are very complex. A data structure required to contain an object's geometry of infinitesimal flat areas would be cumbersome to use and extremely memory consuming. Therefore in computer graphics we tend to use a probabilistic approximation of the material and geometry.

When we observe a bundle of rays interacting an opaque material, we notice that the energy of the bundle is distributed in a hemisphere positioned over the point of interaction. If we measured this energy distribution, stored it in form of a continuous function and used it in our algorithms we would get a very good approximation of the material's behavior. Taking a single ray instead of a bundle, this function becomes a probability density function $f_r : \Omega \times S \times \Omega \to \mathbb{R}$ telling us the probability that an incoming ray from a direction $\omega$ hitting the surface $dA$ at $x$ will be reflected along $\omega'$. This has already been discussed in Chapter 3 where the rendering equation has been presented.

By (2.13)

$$\frac{dL^{out}(x,\omega)}{d\omega'} = f_r(\omega', x, \omega) L^{in}(y, \omega') \cos\theta' \tag{3.1}$$

and therefore for $f_r$

$$f_r(\omega', x, \omega) = \frac{dL^{out}(x, \omega)}{L^{in}(y, \omega') \cos\theta' d\omega'} \tag{3.2}$$

The function $f_r$ is called *Bi-Directional Reflection Distribution Function* and defines partially both geometrical and material properties of the object. Geometrical because it is an approximation of the infinitesimal features of the geometry of the object and material because it also covers the absorption of light energy that interacts with the object.

The problem that remains is how to find such a function for a certain material. This task is nontrivial and so far no fully satisfactory technique has been found for creation of such functions. We can at least restrict the set of all possible functions and consider only those that are *physically plausible*. A physically plausible BRDF does not violate the following conditions.

1. *Reciprocity*

   As the reflection of light is reciprocal, we can exchange the incoming and outgoing directions and the BRDF should yield the same value for both cases

   $$f_r(\theta_i, x, \theta_o) = f_r(\theta_o, x, \theta_i)$$

2. *Conservation of energy*

   The fraction of energy that that arrives from direction $\omega'$ being reflected in the whole hemisphere over $\vec{x}$ must be smaller than 1

   $$\rho(x, \theta_i) = \int_{\Omega} f_r(\theta_i, x, \theta_o) \cos\theta_o d\omega_o \leq 1, \forall x, \theta_i$$

   In other words, the total reflected outgoing energy must be less or equal to the incoming energy being reflected.

To model the macro-geometry of the scene several techniques exist. The objects can be represented by mathematical functions using approaches as spline surfaces, bezier surfaces or coons patches. Another approach is based on *primitives* that can be easily described mathematically (cube, sphere, torus, etc.) and logical operations (union, subtraction, etc.) on these primitives. Finally there is the triangle mesh representation where every object is modeled using triangles. It is clear that not every geometry can be modeled using a finite set of arbitrarily chosen shapes, but we do not address this problem in this thesis.

# 4. THE SENSORY MODEL

The final step of every visualization process is the *seeing*. To see is to display an image of the scene by stimulating a sensory model (usually a grid of sensors) using the information provided by a Global Illumination Algorithm. Seeing is not directly related to the Global Illumination Problem because the illumination does not depend on whether it is seen or not. We believe that the process of visualization should be completely independent from any human factors—we cannot expect to have the luxury of a human operator every time we do a visualization and analyze it. There are several limitations each sensor model has to deal with.

The sensor grid is relatively small compared to the observed reality and the finite number of the sensors in the grid causes a loss of information. Provided that several samples describing a small detail of an object's surface arrive at a sensor. Because they are too close to each other, the sensor can record their energy but any information about the details of the emitting area is lost—as the sensor assumes they are incoming from a single point. Another problem is the number of samples of the lighting information. While the nature works with $\approx 10^{40}$ samples, we only have $\approx 10^{8}$ samples.

By attempting to cope with these problems, the sensor models must keep in mind that the only information available is the flux that falls onto the grid of sensors. Otherwise it can happen that a wrong sensor model compromises the solution of the rendering equation itself. We call this *The Photographer's Approach* as just like photographer, we do not have any information on how and why an object emits a certain value and type of energy. Now we would like to show how nature enables us to see and how our sensory system works.

Our sensory system consists of two parts, an optical sensitive device—the eye and a processing device—the brain. The eye is quite simple and does not differ too much from the systems used in digital cameras. It contains a lens and a sensor grid called retina, composed of photosensitive cells of two types —rods and cones. Rod cells are highly sensitive to the light intensity which allows them to produce very sharp images, but they cannot detect color. These cells enable us to see when there is not enough light available. Cone cells are responsible for color sensing. These cells require more light samples (photons) than rod cells to function correctly. Therefore at low lighting the images we see appear to have less color. The sensing resolution of the eye can be considered high-end by todays standards. It can generate images of about $20000 \times 10000$ pixels but the quality is not balanced. The highest concentration of cells is in the middle of retina and the amount of the type of cells per area

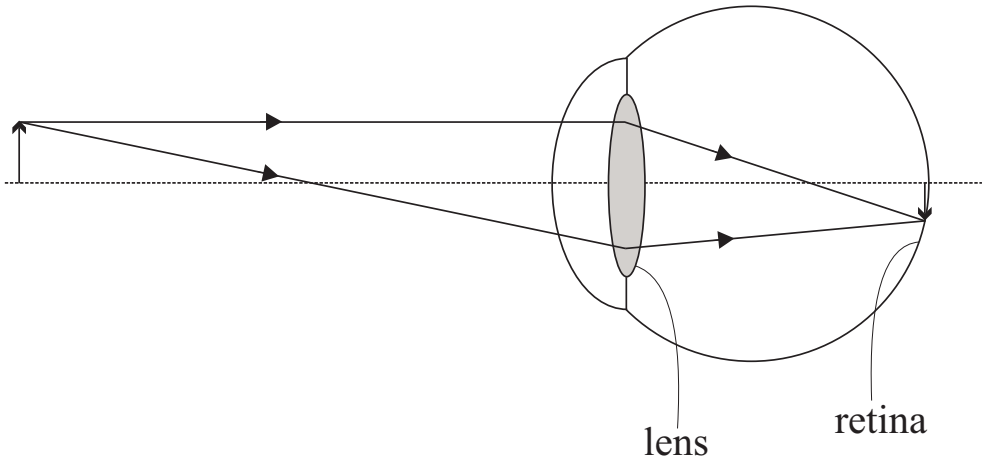


*Fig. 4.1:* The Eye

is also not balanced. The highest concentration of cone cells in the middle of the retina and decreases toward the edges. Therefore the highest quality of the perception with the most vivid colors is in a circular area in the middle of the *output image*. The sensitive area also contains a part where no perception is possible—where the optic nerve connects to the eye. The brain must somehow compensate for this optical imbalance and errors. One peculiar feature of the brain is quite interesting—it never provides an *empty* perception, it instead always tries to complete the image. First it composes the information coming from both our eyes. Then the perception is compared with what we have experienced in the past, what we believe is real and what is important to us. What we *see* (more precisely what we perceive) is the result of this post processing.

# 5. EXISTING SOLUTIONS

Existing solution approaches can be divided into two types—based on two representations of the rendering equation. The *luminance equation* representation (Figure 5.1) describes the energy transfer from the point of view of en emitter.

$$L(x,\omega) = L^e(x,\omega) + \int_\Omega f_r(\omega', x, \omega) \cdot L(y,\omega') \cdot \cos\theta' d\omega' \tag{5.1}$$

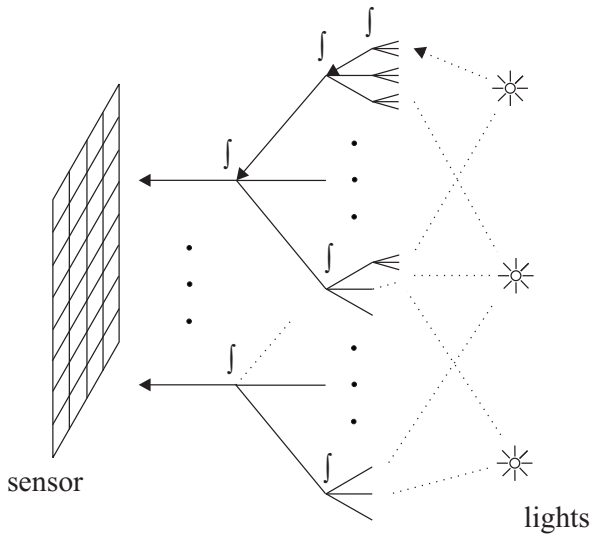


*Fig. 5.1:* Energy transfer according to the luminance equation

It says that the energy emitted in a certain direction is composed of the energy emitted by the object plus the energy reflected along the outgoing direction. In contrary the *potential equation* representation (Figure 5.2) deal with the energy transfer from the point of view of the receiver.

$$W(y,\omega) = W^e(y,\omega) + \int_\Omega f_r(\omega', x, \omega) \cdot W(x,\omega') \cdot \cos\theta d\omega \tag{5.2}$$

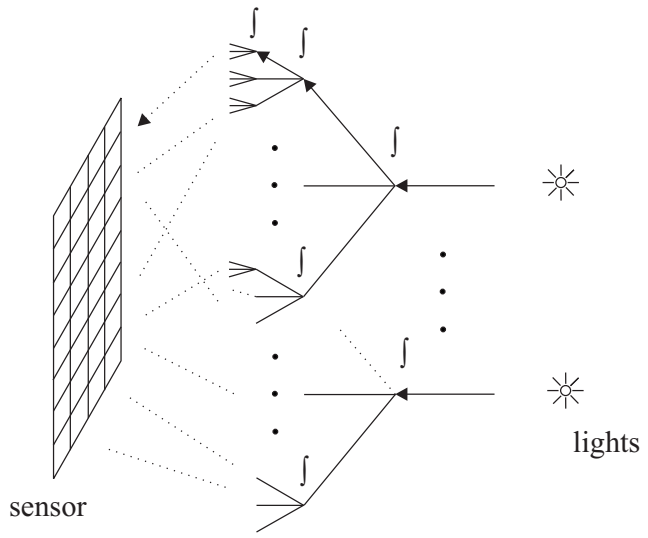


*Fig. 5.2:* Energy transfer according to the potential equation

In other words the amount of energy received is equal to the amount of direct contributions plus the contributions by reflection toward the receiver.

As we have already mentioned, several approaches and respective algorithms exist that claim to give a solution or a partial solution to the Newtonian Global Illumination problem described by the rendering equation. They operate at the basis of various simplifications and approximations of the original problem. But as we will show in some cases they modify the problem itself and therefore the solution they provide is not a solution to the rendering equation any longer. As for the judgment of the algorithms we will not look at the visual *quality* of the output of these algorithms, but only at the mathematical validity of the solutions they provide.

## 5.1 Ray Tracing

Ray Tracing was the first algorithm trying to solve the global illumination problem that actually made some physical sense. It was first published by A. Appel, R.A. Goldstein and R. Nagel at the end of 1960s, but the techniques used in Ray Tracing have already been known for several hundred years. The algorithm claims to solve the luminance equation and can be mathematically described as

$$
\begin{aligned}
L_o(x,\Theta_o) &= L_e(x,\Theta_o) + \\
&+ \sum_{L} \int_{all x_i \in L} v(x,x_l) f_{r,d}(x) L_e(x_l,\Theta'_o) \cos\Theta_l d\omega_l + \\
&+ \int_{\Theta_s \in \Omega_s} f_{r,s}(x,\Theta_s,\Theta_o) L(x_s,\Theta_s) \cos\Theta_s d\omega_s + \rho_d L_a(x) \qquad (5.3)
\end{aligned}
$$

The first term on the right side is the emitted light, the second term refers to the direct contribution of the light sources and the third term is the contribution of the reflected light, evaluated recursively but only for reflections occurring according to the law of reflection (as if every material were an ideal mirror—so called specular reflection). And finally the fourth term is an estimate for the contribution of light reflected throughout the scene by non-specular reflections (ambient light).

Now we will evaluate how this algorithm *solves* the rendering equation. The light transport model corresponds to the rendering equation, but the problem is the material and geometry model. The ambient light (second and higher-order reflections) is approximated by an empirical constant. This approximation causes a systematic error in the computed solution and is hardly a sufficient substitute for all the complex interactions of light-objects in the scene. The specular component by itself should be actually enough to solve the rendering equation provided a special kind of material model were used. According to the classical optics the material of an object can be described by infinitesimal small flat areas and all interactions are perfect mirror-like reflections. Therefore we would need a model that can represent the object composed of these infinitesimal areas. Unfortunately no such model exists that would allow us to model not only the overall shape but also the material of the object on such a small scale. This problem is normally solved by using special functions as described in Chapter 3, but in the case of Ray Tracing we have no such option. The Ray Tracing algorithm only solves the rendering equation for scenes composed of objects with a perfect mirror-like material and therefore does not provide a solution to the Newtonian Global Illumination Problem.

## 5.2 Radiosity

The algorithm of Radiosity is based on a theory that has been used to solve problems in radiative heat transfer since 1950s. It was first published in 1984 by C. Goral, K. E. Torrance, D. P. Greenberg and B. Battaile.

This method makes an assumption that all surfaces in the scene are ideal (Lambertian) diffusers. Energy hitting such a material is scattered evenly in the hemisphere over the point of interest. The equation describing Radiosity is given as

$$B_i = E_i + R_i \int_j B_j \cdot F_{ij} \tag{5.4}$$

where $E_i$ is the energy emitted by the patch, $R_i$ describes its reflectivity and $F_{ij}$ defines the geometrical relations between two patches. It says that the energy radiated by a patch is the sum of the energy emitted plus the reflection of energy incoming from other patches. The algorithm starts with some patches being emitters and the rest having no energy. In a number of iteration steps it tries to distribute the energy over the scene, until the $\delta$ of energy difference between two iteration becomes smaller than a predefined certain value. The radiosity methods make two assumptions which convert the rendering equation into a linear equation system:

1. all reflections are perfectly diffuse (lambertian);
2. the geometry is composed of a finite number of flat areas (patches).

Of course, not every object can be generally represented by perfectly diffuse patches. Materials such as mirror of of generally any material that does not reflect the incoming energy evenly into all directions cannot by visualized using Radiosity. Another problem is the fact that the material model is fully integrated into the algorithm. Without this principle, the algorithm would not work. The Radiosity provides a solution only to a simplification of the Global Illumination Problem. Radiosity can be combined with the Ray Tracing algorithm that is able to visualize specular materials. The results are visually satisfactory, yet still not correct.

## 5.3 Path Tracing

This algorithm was presented by J. T. Kajiya in [Kaj86] as a solution to the rendering equation. It provides a solution to the Global Illumination Problem when run with infinite resources for an infinite time. Similar to the Ray Tracing algorithm it generates light paths starting at the sensor and interacting with the scene. The rays reflect according to probability distribution functions that define the material properties. Using the Monte-Carlo method of solving the rendering equation this algorithm estimates the light energy transfer in the scene. The more paths are generated the closer this estimate comes to the actual light transfer. Note that path tracing starts with the camera, i.e. it does not separate solving the rendering equation from the visualization phase.

This algorithm is able to display any phenomena that can be described by the Classical Optics.

## 5.4 Photon Mapping

Photon mapping is a two-pass algorithm presented by H. W. Jensen in [Jen96] and [Jen01].

The first pass is based on a concept similar to Path Tracing. But instead of gathering the energy along the paths, it starts at the lights and tries to distribute the energy in the scene by shooting samples named *photons*. These are stored in a *photon map*. Like in path tracing large numbers of photons are required to generate a good result. Actually these photons could be visualized directly, but the output does not look nice enough—as there are not enough samples. There is therefore a second pass by a Monte-Carlo Ray Tracing algorithm. Using this algorithm we obtain a number of samples for each pixel, that estimate the radiance. To obtain a sample a ray is traced from the eye through the pixel into the scene. At the point of intersection the photon map is queried for photons lying *near* the point of intersection. Using various methods the nearby photons are used to calculate the radiance at the point of intersection. This process is described in more detail in [Jen01]. What is important is the fact that even the authors confess this process does not always give correct results. Methods exist which compensate for these errors but often they add even more approximations. Each methods brings along a set of parameters which can be set do minimize different types of errors. The result is that when visualizing a scene we have to set a lot of parameters so that the output looks correct, but for another scene the same parameters can generate a wrong output. Another problem is the fact that it is hard to calculate how these methods affect the distribution of energy in the scene and so it is unclear if the algorithm (after the second pass) still is a solution to the rendering equation.

# 6. OUR SOLUTION

It this section, we will present our solution to the Newtonian Global Illumination Problem. Our algorithm is intentionally kept simple. Its task is not to generate *nice* pictures, but to prove that our approach to solve the Newtonian Global Illumination is valid. Possible improvements of the speed of the output image generation and its quality will be discussed later, though they are not very important as they do not affect the correctness of our solution. The algorithm will be described in the same manner we used to describe the Newtonian Global Illumination Problem.

## *6.1 The Algorithm*

### *6.1.1 The General Idea*

The basis of the algorithm is similar to the first pass of the Photon Mapping algorithm [Jen01]. It solves the potential equation (5.2) by generating a large number of samples that originate at the light sources. The potential equation as a form of the rendering equation can be rewritten in form of a Louville-Neumann Series

$$\begin{aligned} W \quad & (y,\omega') = W^e(y,\omega') + \\ + \quad & \int W^e(x,\omega) f(\omega', y, \omega) \cos\theta d\omega + \\ + \quad & \int\int W^e(x',\omega'') f(\omega', y, \omega) f(\omega'', x, \omega) \cos\theta \cos\theta'' d\omega d\omega'' + \\ + \quad & \int\int\int W^e(x'',\omega''') f(\omega', y, \omega) f(\omega'', x, \omega) f(\omega''', x', \omega'') \cos\theta \cos\theta'' \cos\theta''' d\omega d\omega'' d\omega''' + \\ + \quad & \cdots \end{aligned} \tag{6.1}$$

Using the method of Monte-Carlo integration we approximate each of these integrals. The approximation for the first integral in series, according to (2.28) by sampling a direction $\alpha_1$ on a hemisphere over $x$ is

$$\frac{W^e(x,\alpha_1) f(\omega', y, \alpha_1) \cos\theta}{f_1 \alpha_1} \tag{6.2}$$

where $\alpha_1$ is a random value corresponding to the probability distribution $f_1$. We can estimate the second and third integral in the same manner

$$\frac{W^e(x', \alpha_2) f(\omega', y, \alpha_1) f(\alpha_2, x, \alpha_1) \cos\theta \cos\theta''}{f_2(\alpha_1, \alpha_2)} \tag{6.3}$$

$$\frac{W^e(x'', \alpha_3) f(\omega', y, \alpha_1) f(\alpha_2, x, \alpha_1) f(\alpha_3, x', \alpha_2) \cos\theta \cos\theta'' \cos\theta'''}{f_3(\alpha_1, \alpha_2, \alpha_3)} \tag{6.4}$$

and also for the $n$-th integral

$$\frac{W^e(x'', \alpha_n) f(\omega', y, \alpha_1) f(\alpha_2, x, \alpha_1) \cdots f(\alpha_n, x', \alpha_{n-1}) \cos\theta \cos\theta'' \cdots \cos\theta^n}{f_3(\alpha_1, \alpha_2, \cdots, \alpha_n)} \tag{6.5}$$

Together these estimates describe a path starting at a light source and traveling through the scene, interacting with it in $n$ reflections. This path is a very rough estimate for the potential equation. Of course an output image generated from an approximation of the scene by a single path cannot be satisfactory. Therefore a large number of such paths is needed. In case of infinitely many paths we obtain the solution to the potential equation (and thus to the rendering equation). Based on this theory, we will now propose an algorithm which directly solves the potential equation (Algorithm 1).

### 6.1.2 Light Sampling

We will refer to Algorithm 1 as *light sampling*. Before we describe the process itself we will need some basic definitions.

A *scene* consists of *objects* and *light sources*. Each object is defined by its geometry and a bi-directional reflectance distribution function that describes the material properties. A light source is described by its power and an emission function $f_e$. A *sample* $P_i$ is a result of an intersection of a ray of light and an object. A sample contains the following information:

1. intersection point $x$
2. incoming vector $\omega$
3. normal vector $N$
4. original light source $L_i$
5. material properties of the intersected object $f_r$
6. energy modifier $w$

**Algorithm 1** Light Sampling

```
{declaration}
elem {variable representing a light source or a sample}
dataset {a data structure containing calculated samples - paths}
{generating light paths}
Empty(dataset)
while not break do
  elem ⇐ SelectRandomElement(dataset)
  if TypeOf(elem) = "light" then
    ray ⇐ RayFromLightSrc(elem)
  end if
  if TypeOf(elem) ≠ "light" then
    ray ⇐ RayFromSample(elem)
  end if
  intersection ⇐ FindIntersection(ray, scene)
  if (intersection) then
    sample ⇐ GetSampleData(intersection, scene, elem)
    Add(dataset, sample)
  end if
end while
{energy distribution}
for all elem in dataset do
  if TypeOf(elem) ≠ "light" then
    nweight[GetLight(elem)] = nweight[GetLight(elem)] + elem.weight
  end if
end for
for all elem in dataset do
  if TypeOf(elem) ≠ "light" then
    elem.energy = Power(GetLight(elem)) ∗ elem.weight / nweight[GetLight(elem)]
  end if
end for
{visualization}
for all sensor in sensorgrid do
  for all elem in dataset do
    if TypeOf(elem) ≠ "light" then
      if CanSee(sensor, elem) then
        sensor.energy = sensor.energy + GetEnergyContrib(sensor, elem)
      end if
    end if
  end for
end for
```

A *dataset* is a storage structure for samples.

The algorithm begins with an empty dataset. A random choice is made whether we generate a new sample from a light source or from the already calculated samples in the dataset. As the dataset is empty we choose a light source. According to the emission function a random direction $\omega'$ is determined and a ray is generated—starting at the light and oriented toward the sampled direction. If an intersection of the ray with the scene exists, the intersection point $x'$ and the normal vector $N'$ at the point of intersection are calculated. Now we have the necessary information to define a sample (Figure 6.1). As this sample originates directly from a light source $L_i'$, its energy modifier $w'$ is set to 1. The sample $P_i'$ is then stored in the dataset.

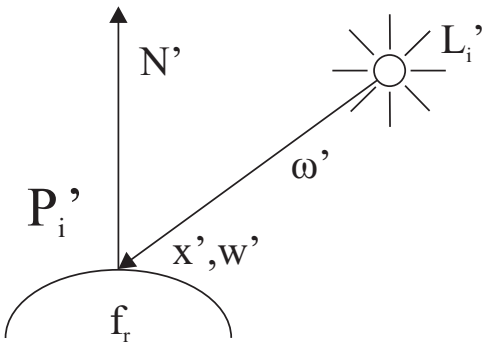


*Fig. 6.1:* Generating a new sample from a light source

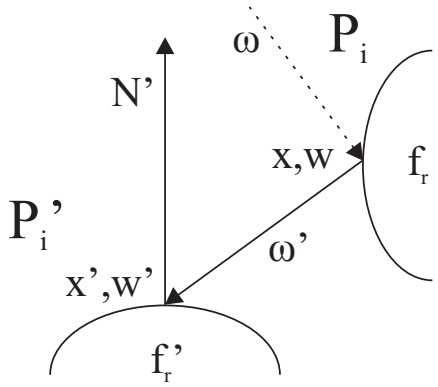


*Fig. 6.2:* Generating a new sample from another sample

In the next step, we have a dataset containing several samples. We make a random choice whether we generate a sample from a light source or from a previously calculated sample in the dataset. Let us assume we chose the dataset. We select a sample $P_i$ from the dataset. We choose a random direction $\omega'$ on a hemisphere over the point of intersection $x$ of the current sample $P_i$. Again a ray defined by the intersection point $x$ and the direction $\omega'$ is traced

through the scene. Just as in the case of a light-source-generated sample we calculate all necessary data for a new sample $P_i'$. The only difference is the energy modifier. By selecting a sample from a dataset we actually decided to extend a path. Therefore our new sample $P_i'$ receives a fraction of the energy of the sample $P_i$. The energy modifier $w'$ is determined using the material properties $f_r$ stored with the sample $P_i$, the incoming vector $\omega$ and the vector $\omega'$ (Figure 6.2)

$$w' = w \cdot f_r(\omega', x, \omega) \cdot g(x, x') \tag{6.6}$$

The energy modifier describes the fraction of energy coming from $\omega$ to $x$ that is being reflected along $\omega'$ toward $x'$. The function $g$ describes geometrical relations between infinitesimal surfaces of $x$ and $x'$.

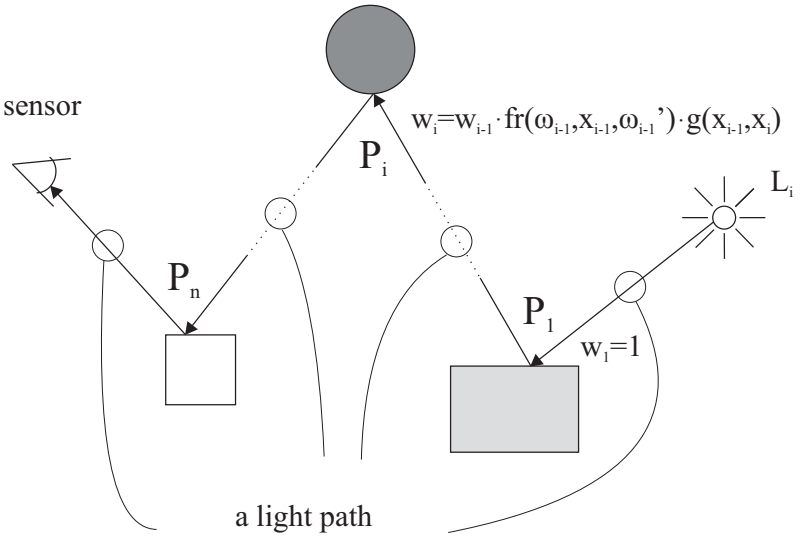


*Fig. 6.3:* Energy transport along a light path

Each sample can be regarded as the final segment of a single path (Figure 6.3). The energy modifier of the sample defines the amount of energy transported along this path. This process should continue infinitely—creating an infinite number of samples, thus generating all possible paths needed to solve the potential equation. Of course this cannot be accomplished in reality. Therefore the process is stopped at some point, usually by user interaction or when the resources are depleted.

### *6.1.3 Scene Definition*

As we have already mentioned the scene consists of light sources and objects. The light sources can be of two types—point-like light sources defined by a single point $x$ which emit light in a sphere around the point $x$ and planar light sources defined by an area $S$ and normal

$N$ that emit light from each point of the surface of $S$ in any direction on a hemisphere over the point of emission oriented according to $N$. The light source does not have to emit light uniformly in all directions. This fact is represented by a emission function $f_e$ which is a probability distribution function similar to bi-directional reflectance distribution functions for materials.

The objects are represented as triangle meshes. This representation allows for fast computing of ray-object interaction [MT97] and is easy to implement. The material properties and micro-geometry is described by the Modified Phong bi-directional reflectance distribution function [LW94]. This material model is proved to be physically plausible and it allows for easy implementation of the importance sampling method.

All these data are stored in VRML.

### 6.1.4 Visualization

Having calculated the data of the energy transfer in the scene, we need to distribute the energy emitted by the light sources. We will consider the emission of light for a infinitesimal unit of time. During this time only a certain amount of energy can be emitted by the light sources

$$E = \sum_{Lights} L_k \tag{6.7}$$

where $E(L_k)$ is the power of the light source $L_k$.

The energy modifier of a sample represents the fraction of a unit amount of energy that is transported along the path with the sample as its final segment. To find the energy distribution we divide the energy according to the energy modifiers. It is also important that we divide the energy of a single light source only among those samples which paths originate at the given light source. Therefore, for the energy of a sample $E(P_i)$ holds

$$W_k = \sum_{Samples} w_i$$

$$E(P_i) = \frac{E(L_k)}{W_k} \cdot w_i \tag{6.8}$$

where $L_k$ is the original light source of the sample, $W_k$ is the sum of the weights of the samples for $L_k$ and $w_i$ is the modifier that determines the fraction of the energy for the sample $P_i$.

Now for the final visualization we must extend each path one more time to get the energy directly to the sensors. The visualization model consists of two parallel grids (Figure 6.4). First one is covered by sensors—each represented by one pixel. The other grid consists of

small squares. Each square corresponds to a certain sensor (pixel) on the sensor-grid. Now the algorithm evaluates for each sensor, square and every sample in database whether the sample is seen from the sensor, while limited by the solid angle defined by the sensor and the square.

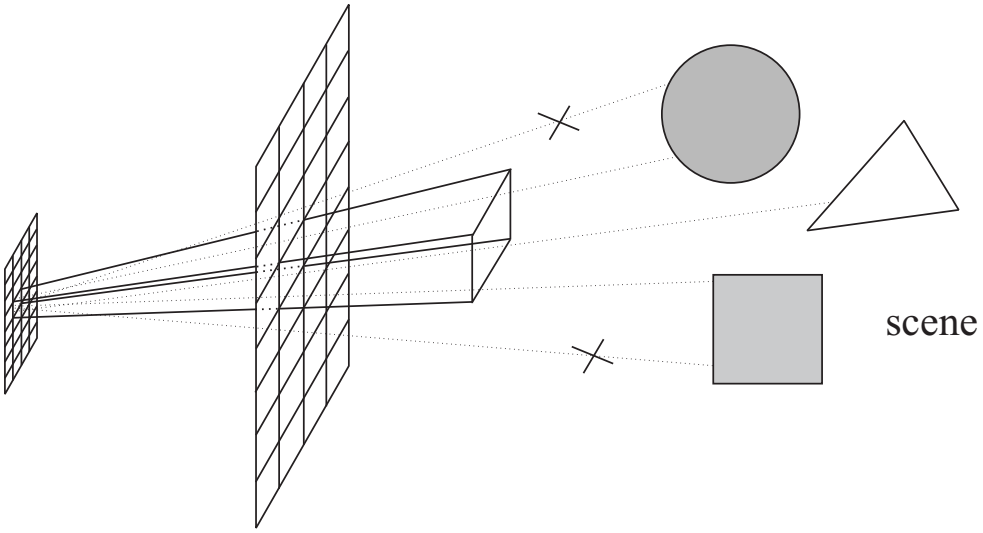


*Fig. 6.4:* The Sensor Model

For every sensor $S_i$, the energy contribution of each *seen* sample is

$$R_j = \sum_{S_i;P_j \in solidangle(S_i,SQ_i)} f(\omega, x, \omega')$$

$$E(S_i, P_j) = E(P_j) \cdot g(S_i, P_j) \cdot \frac{f(\omega, x, \omega')}{R_j} \tag{6.9}$$

It is the fraction of energy transported to $x$ along $\omega$ being reflected to $S_i$ along $\omega'$. Then for the total magnitude of the stimulus of the sensor $S_i$ holds

$$E(S_i) = \sum_{Samples} E(S_i, P_j) \tag{6.10}$$

These values $E(S_i)$ are regarded as weights when applied on a pixel map to obtain the final image.

### 6.1.5 Summary

Due to the infinitesimal nature of the problem, we will never be able to provide an exact solution to the Global Illumination Problem. Our algorithm solves the problem stochastically. It does not attempt to output nice images, it just simulates the classical (newtonian) optics and gives a guarantee that its solution converges to the exact solution.

We will now compare our solution to the Path Tracing algorithm presented by Kajiya in [Kaj86]. While the Path Tracing algorithm is a solution to the luminance equation, our algorithm solves the potential equation form of the rendering equation. As we have already shown, both equations provide the same results, but lead to different approaches in application of the Monte-Carlo method. Our algorithm varies from Path Tracing in following properties:

1. To solve the potential equation our algorithm creates samples beginning at the light sources instead of starting at the sensor.
2. Every path is stored in a persistent data structure. A single path is represented by a sample carrying the information about the fraction of energy transported along the path.
3. The energy of the light sources is distributed among among the path based on the fraction of energy they transport.

These properties lead to following differences between the algorithms.

Due to the fact that the paths start at light sources, the energy distribution process is independent from the visualization. Therefore the same calculated data can be used to render images with any position of the sensor in the scene. The visualization process is a simple extension of the paths so that the energy reaches the sensor—it is possible to use various types of sensors representing various optical measuring devices.

We can use the methods of importance sampling more efficiently, because we can easily chose those paths that carry more energy. In Path Tracing we could only guess whether the path will be of any significance for the generation of the final image.

Because the energy distribution data is persistent, it can be refined when more computational resources are available or a better image output is required.

Instead of calculating the actual energy distribution we calculate how the energy is distributed. This fact allows us to change certain parts of the input without having to recompute the illumination from the beginning. For example, changing intensity of a light source or adding a new light source can be done on-the-fly. Moreover, in our method the computed illumination is explicitly stored and is independent on the camera. This makes our method very suitable for walkthrough animations in which camera is moving.

The algorithm is energy conserving as only the energy emitted by the lights is distributed among the samples.

The algorithm still requires large numbers of samples to produce satisfactory results. Due to the fact that we start at light sources and do not consider the position of the sensor, we also generate samples (paths) that do not contribute much or at all to the final image.

Overall, our approach is superior in many aspects to the original Path Tracing algorithm.

## 6.2 Improvements

Our algorithm calculates the illumination according to the Classical Optics—and this defines the set of phenomena which appear in computed images. We will now discuss efficiency of the algorithm. It has to run for an infinite time to provide the complete solution. In reality, this is not feasible and we need the algorithm to display the calculated data after some reasonable computational time. We naturally expect these data to have certain level of visual quality.

There are several techniques which help to speed up the convergence of the algorithm. Among these techniques we will discuss *importance sampling*, *russian roulette*, *priority sample selection* and *parallelization*.

### 6.2.1 Russian Roulette

With a limited amount of time and resources we cannot calculate all needed data. It is therefore desirable to remove data that are not significant and do not contribute too much to the final image. Still this must be done in such a way, that the correctness of the algorithm is not compromised—every possible sample (light path) must be generated with non-zero probability.

The basis of russian roulette is a function $r : \mathcal{S} \to \mathbb{R}$, $r(P_i) = p, p \in (0, 1)$. The number $p$ is the probability that the sample contributes to the final image. As $p$ is always greater than 0 the condition of being able to calculate every possible path in infinite time is not violated. The function $r$ represents a set of various criteria. These criteria can vary from algorithm to algorithm based on what makes a sample *important*.

In our algorithm we have chosen the energy of the sample $w_i$ as such a criterion. When selected randomly from the dataset, the sample is evaluated whether the algorithm continues and extends the path or the path is ended and the sample discarded from the dataset. In this way the dataset statistically contains only paths which transport the most energy. This technique can also be applied when the memory resources are exhausted. We can refine the result by removing *unimportant* samples and so freeing resources for calculation of samples with higher energy.

### 6.2.2 Importance Sampling

Generally, most materials do not reflect light energy evenly in the hemisphere over the point of intersection. When we sample the BRDF that represents the material uniformly, we generate samples also in directions that are not very important from the point of view of the energy transfer. A similar situation can be observed regarding the scene. Clearly there are directions on the hemisphere where no intersection with an object of the scene occurs. We thus spend time and resources on calculating samples that do not contribute much to the final image. Although these samples are necessary for the correctness of the final image, it is reasonable to generate more samples in directions of higher importance.

This can be achieved by sampling the BRDF with a non-uniform probability distribution function $f$, that comes as close as possible to the sampled BRDF and/or takes into account the position of other object in the scene (Figure 6.5). This technique is called importance sampling.

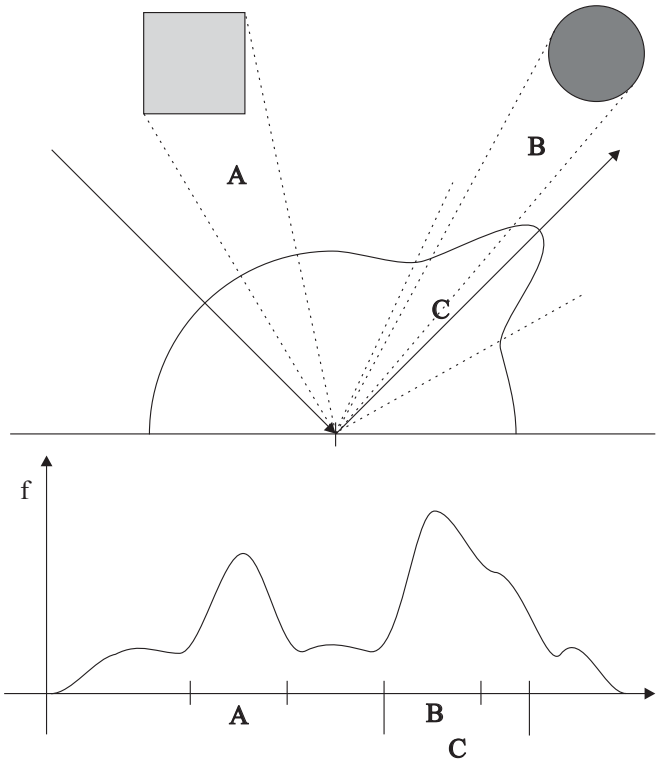


*Fig. 6.5:* Importance Sampling

We have implemented the Modified Phong BRDF with the respective importance sampling—as described in [LW94].

### 6.2.3 Priority Sample Selection

To accelerate the convergence of the output image we need to focus on calculating paths that carry the largest amounts of energy. One way to achieve this is a selection of the samples for path extension according to a non-uniform probability distribution function.

Imagine we have two samples $i$, $k$ with energy modifiers $w_i$ and $w_k$. Ideally we want that sample $i$ is selected $fracw_iw_k$ more often than sample $k$. By simple analogy on natural numbers and using numerical methods, we can create a probability distribution function $f$ that describes the probability of such selection. We order the samples in a queue with high energy modifiers samples at the beginning—this can be achieved easily by keeping the dataset orderer with every insertion of a sample. To search, add and remove samples effectively, we recommend using a search tree defined on top of this queue. When we want to select a sample, we determine its position using the function $f$.

In this way, we will select samples with higher energy more often, which statistically generate high-energy descendants. This increases the number of paths transporting larger amounts of energy.

### 6.2.4 Parallelization

The methods presented earlier allow us to use the computing resources more efficiently by generating those samples among the first which have the greatest contribution to the visual aspects of the scene. Still they do not address the basic problem of generating large amounts of samples in reasonable time. By using parallelization we can utilize the computing power of several computers working on a single task.

Compared to a sequential algorithm, its parallel counterpart running on $n$ processors should in ideal case achieve a speedup of $T = \frac{T_S}{n}$. But due to the need of the individual processes to communicate this is usually less than the ideal value [CDR02]. The level of speedup depends on several factors, such as the individual nature of the problem, how effectively it can be divided in separate subtasks and how much the individual processes to have communicate do solve the task.

Algorithms such as ours are very suitable for parallelization. As we have already mentioned we need to generate large numbers of samples. According to (6.1) the individual samples (light paths) are independent from each other. Therefore a trivial parallelization technique can be easily applied and the achieved speedup comes close to the ideal values:

The master process distributes the scene data among the worker processes. Each worker runs the same light sampling algorithm to generate a dataset containing the calculated samples (paths). On demand the master send a termination signal to the processes. The workers finish the calculation and send their local dataset to the master. Master merges the data and visualizes it.

The same optimization techniques can be used in the processes as in the sequential algorithm—russian roulette, importance sampling and the priority sample selection. To compare the parallel implementation to its sequential counterpart, it is possible to run a similar algorithm sequentially. A single process calculates a number of independent datasets, thus simulating the calculation by several worker processes (Figure 6.6).

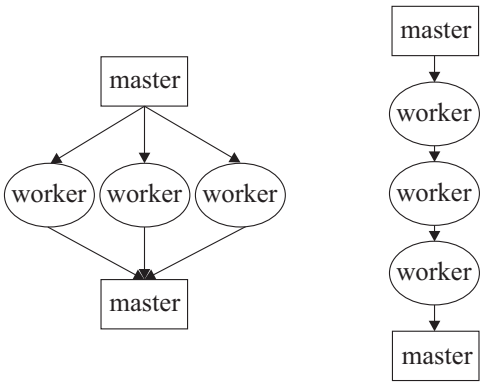


*Fig. 6.6:* Comparison of the data-flow in a parallel implementation and its sequential simulation by a single process

## 6.3 Implementation

The algorithm was implemented on the x86 architecture. However, it can be compiled to run on any system that has an ANSI C++ compliant compiler. The source code can be easily modified to comply with the ANSI C standard. We used the Eclipse environment with CDT plug-in.

Effort has been made to keep the number of parameters as low as possible. In fact these are not needed for a correct approach as there should be nothing to set except the scene definition. The necessary parameters are:

1. a scene with geometry and material definitions (using a modified VRML file)
2. light source definition (in the same VRML file)
3. number of samples to be generated
4. position of the observer

To speed up the convergence and thus improve the image quality we have utilized the techniques of importance sampling and russian roulette.

# 7. CONCLUSIONS

We have shown that is is possible to define the Global Illumination Problem according to a physical theory, to describe it mathematically and find its algorithmic solution. By implementing this solution we have proved that it is possible to use this approach for image generation with satisfactory results. Instead of comparing output images of different solution to decide on the correctness of the output we can say that our algorithm provides a correct solution without even running it. The visual quality depends only on the computational time and available resources and not on *features* that the algorithm does or does not implement. In spite of of its similarity, our concept is superior to e.g. photon mapping [Jen01]. That is because we carefully avoid unnecessary approximation methods that sacrifice mathematical correctness of the solution for image quality. As a result our algorithm can be applied to any scene without special tuning of any parameters. We can expect it to generate images represent the exact behavior of light according to the classical optics theory (and thus according to the rendering equation).

We know that our approach is in principle not new—the theory behind it has already existed for a long time. Nevertheless, we are aware of only a few implementations which strive to simulate the newtonian optics correctly [LW93], [VG97].

We now provide several images (Figure 7.1, Figure 7.2, Figure 7.3) generated by an implementation of the algorithm. They were calculated using 4000000 samples, each image took 20 minutes to compute. The noise can be reduced by choosing a different sensor model and by using post processing techniques.

*Fig. 7.1:* A scene with 3000 triangles

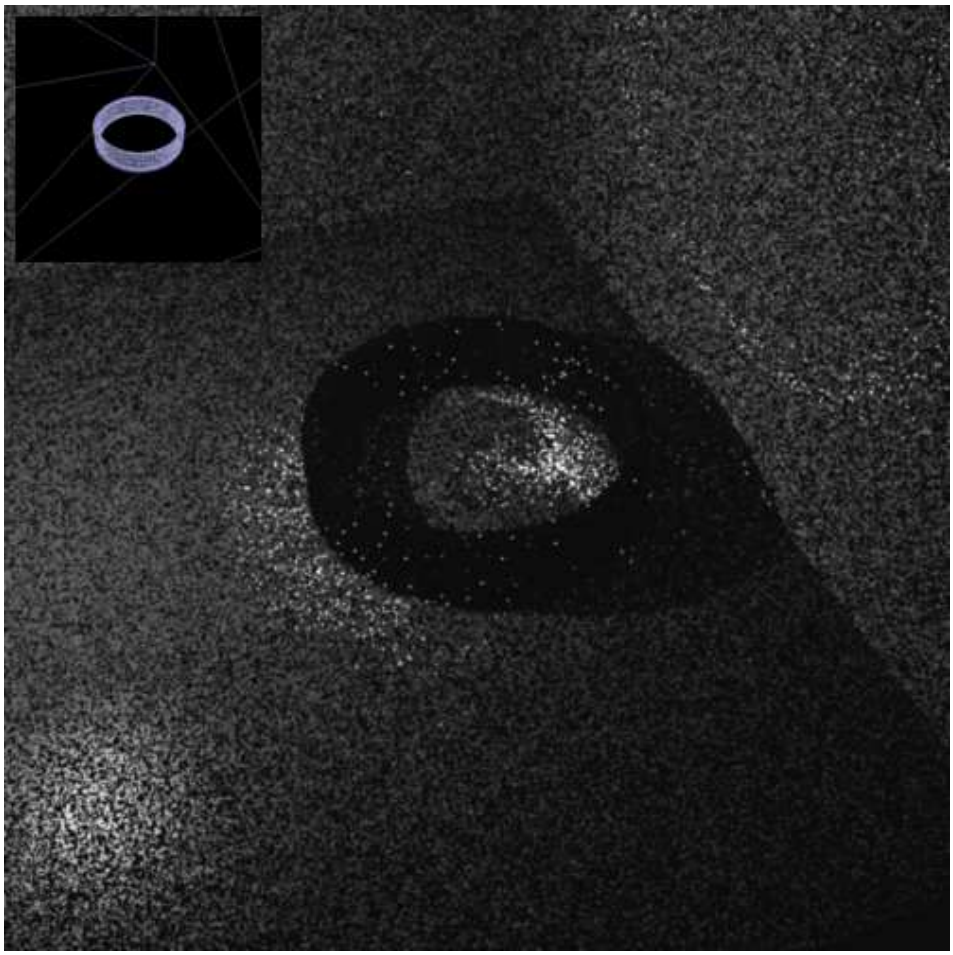

*Fig. 7.2:* A scene with caustics

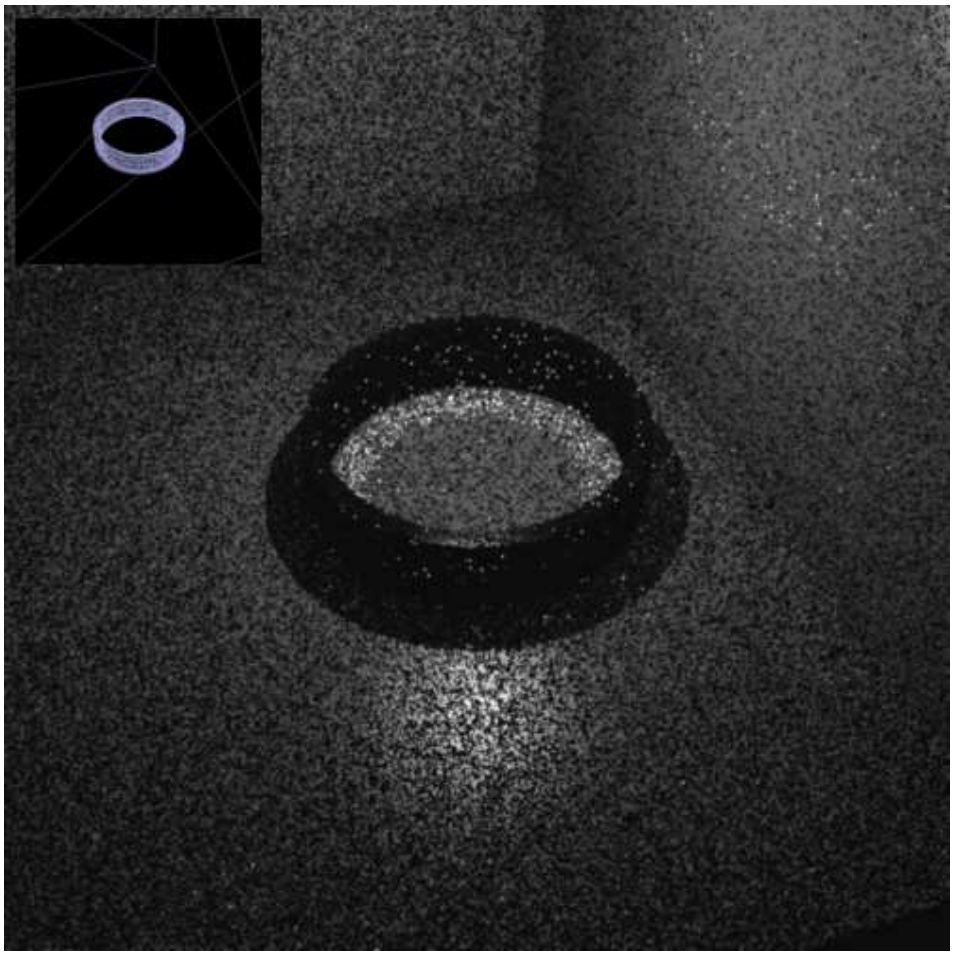

*Fig. 7.3:* A scene with caustics - different position of light source

# BIBLIOGRAPHY


[CDR02] Alan Chalmers, Timothy Davis, and Erik Reinhard, editors. *Practical parallel rendering*. A. K. Peters, Ltd., Natick, MA, USA, 2002.

[Chr99] Per H. Christensen. Faster photon map global illumination. *J. Graph. Tools*, 4(3):1–10, 1999.

[Dut] Philip Dutre. Global illumination compendium.

[Fey88] Richard P. Feynman. *QED: The Strange Theory of Light and Matter*. Princeton University Press, October 1988.

[GCT86] Donald P. Greenberg, Michael F. Cohen, and Kenneth E. Torrance. Radiosity: A method for computing global illumination. *The Visual Computer*, 2(5):291–297, 1986.

[Gla89] Andrew S. Glassner, editor. *An introduction to ray tracing*. Academic Press Ltd., London, UK, UK, 1989.

[GT05] Otavio Good and Zachary Taylor. Optimized photon tracing using spherical harmonic light maps. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Sketches*, page 53, New York, NY, USA, 2005. ACM Press.

[GWS04] Johannes Günther, Ingo Wald, and Philipp Slusallek. Realtime caustics using distributed photon mapping. In Alexander Keller and Henrik Wann Jensen, editors, *Rendering Techniques 2004 : Eurographics Symposium on Rendering*, pages 111–121, Norkoeping, Sweden, June 2004. Eurographics Association, Eurographics.

[Jen96] Henrik Wann Jensen. Global illumination using photon maps. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 21–30, New York, NY, 1996. Springer-Verlag/Wien.

[Jen01] Henrik Wann Jensen. *Realistic image synthesis using photon mapping*. A. K. Peters, Ltd., Natick, MA, USA, 2001.

[Kaj86] J. T. Kajiya. The rendering equation. *Computer Graphics*, 20(4):143–150, August 1986. Proc. of SIGGRAPH'86.

[KW00] Alexander Keller and Ingo Wald. Efficient importance sampling techniques for the photon map. In *Proceedings of Vision Modelling and Visualization 2000*, pages 271–279, Saarbruecken, Germany, 2000.

[LW93] Eric P. Lafortune and Yves D. Willems. Bi-directional Path Tracing. In H. P. Santo, editor, *Proceedings of Third International Conference on Computational Graphics and Visualization Techniques (Compugraphics '93)*, pages 145–153, Alvor, Portugal, 1993.

[LW94] Eric P. Lafortune and Yves D. Willems. Using the modified phong brdf for physically based rendering. Technical Report CW197, Leuven, Belgium, 1994.

[MT97] Tomas Mueller and Ben Trumbore. Fast, minimum storage ray-triangle intersection. *J. Graph. Tools*, 2(1):21–28, 1997.

[Neu01] Attila Neumann. *Constructions of Bidirectional Reflection Distribution Functions.* PhD thesis, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria, 2001.

[New21] I. Newton. *Opticks: Or A Treatise of the Reflections, Refractions, Inflections and Colours of Light.* William and John Innys, third edition, 1721.

[Sal78] T. Salat. *Malá encyklopédia matematiky.* Obzor, 1978.

[SK99] L. Szirmay-Kalos. Monte-carlo methods in global illumination, 1999.

[SSK] Laszlo Szecsi and Laszlo Szirmay-Kalos. Improved indirect photon mapping with weighted importance sampling.

[Str79] A. Strba. *Všeobecná fyzika 3 Optika.* Alfa, 1979.

[VG97] E. Veach and L. Guibas. Metropolis light transport, 1997.

# LIST OF FIGURES