Robust Monte Carlo Methods for Photorealistic Rendering of Volumetric Effects

Mark Pauly

Oktober 1999

Universität Kaiserslautern
Fachbereich Informatik
AG Numerische Algorithmen
Prof. Dr. Stefan Heinrich

Aufgabenstellung und Betreuung:
Dr. Alexander Keller

Erklärung

Hiermit versichere ich, die vorliegende Arbeit selbständig und
nur mit den angegebenen Hilfsmitteln angefertigt zu haben.

Kaiserslautern, Oktober 1999

Acknowledgements

I would like to thank the people who supported me during my work on this thesis: Alexander Keller for his assistance and for encouraging me to pursue my own ideas. Thomas Kollig, whose expertise in Metropolis light transport was invaluable for the completion of this thesis. Ilja Friedel, Ingo Wald and Johannes Timmer for many fruitful discussions on computer graphics and general politics. David Ebert for his cloud data that was very useful for testing our algorithms.

Sincere thanks also to Anton Vukovic for providing food, companionship and a mattress in times of need.

Contents

Chapter 1

Introduction

Photorealistic rendering is concerned with the synthesis of images that are indistinguishable for the human visual system from a photographic reproduction of the actual environment. Many global illumination algorithms have been developed to meet this goal, yet the majority of these methods focuses on scenes without participating media, disregarding volumetric effects due to clouds, fog, smoke or fire. These can greatly enhance the realism of a rendered image, however, and in many application are the decisive factor of the simulation. Visibility analysis for traffic or building design, flight simulation, fire research and high-quality special effects in animation systems all rely on a realistic depiction of volumetric phenomena [Rus94].
The objective of this thesis is to develop, implement and analyse global illumination algorithms for scenes with radiatively participating media. We concentrate on algorithms that are

- *physically-based*
  In order to cover a wide range of lighting effects, the simulation should be based on the fundamental equations governing the distribution of light. This ensures that we can accurately predict the visual appearance of a specified scene.

- *general purpose*
  Many existing approaches are limited to certain classes of geometry, e.g. polygonal surfaces, or reflection models, e.g. perfectly diffuse reflectors. We are interested in algorithms that are not restricted in this sense, as we want to support the full complexity of real environments.

- *robust*
  Most global illumination algorithms are specifically designed for a certain subclass of inputs, e.g. they perform well for scenes with strong direct illumination, but need an excessive amount of resources to handle scenes where indirect ligthing is predominant. In contrast, a robust algorithm can efficiently handle a broad range of input scenes and lighting situations, thus being suitable for many application domains.

Generating an image based on the distribution of radiative energy in a scene is essentially an integration problem, where the integrand is usually a high-dimensional, discontinuous function. To comply with the above requirements we set our focus on Monte Carlo algorithms, currently the most versatile instrument for numerical integration. We extend two existing approaches, *bidirectional path tracing* (BDPT) and *Metropolis light transport* (MLT) [Vea97] both in their

theoretical framework and algorithmic implementation to incorporate participating media.

1.1 Thesis organization

Chapter 2 provides the theoretical background. It derives the fundamental equation governing the distribution of light and presents a theoretical framework that allows a broad range of numerical integration methods to be applied for its evaluation. **Chapter 3** is concerned with different aspects of sampling. It describes how participating media are modeled and light transport paths are generated. **Chapter 4** introduces the BDPT and MLT algorithms for participating media. It gives a conceptual overview of both methods and discusses some implementation details. **Chapter 5** presents results that were obtained with both algorithms and provides a comparative analysis of BDPT and MLT. **Chapter 6** concludes this thesis with a summary and some suggestions for future research.

Chapter 2

Light Transport for Global Illumination

The goal of any global illumination algorithm is to produce an image of a given scene, based on the distribution of radiative energy in the specified environment. Section 2.2 presents a derivation of the *light transport equation*, which describes this distribution in equilibrium. We will show how the light transport equation can be transformed from an integro-differential equation to an integral expression by applying the *path integral formulation*. This formalism has been introduced in [Vea97] for scenes in vacuum. Section 2.3 presents a generalization of the method that incorporates participating media.

2.1 Domains and Measures

A scene consists of a collection of objects, described by a number of boundary surfaces; the space between objects can be filled with participating media. More precisely:

- $\mathcal{V} \subset \mathbb{R}^3$ is a finite volume,
- $\partial\mathcal{V}$ is the boundary of $\mathcal{V}$, i.e the finite set of surfaces in $\mathbb{R}^3$ that specify objects,
- $\mathcal{V}^0 := \mathcal{V} \setminus \partial\mathcal{V}$ denotes all points in $\mathcal{V}$ that do not lie on a surface, i.e. those points where volumetric interactions can occur.

A direction is specified by $\omega \in S^2$, where S^2 is the unit sphere in $\mathbb{R}^3$. ω_{xy} is the direction vector that points from x to y and is defined through $\omega_{xy} := \frac{y-x}{\|y-x\|}$, for $x, y \in \mathbb{R}^3$ and $\|\cdot\|$ the L_2-Norm in $\mathbb{R}^3$.
Let $D_2 \subseteq \partial\mathcal{V}$, $D_3 \subseteq \mathcal{V}$, $\Omega \subseteq S^2$. The following measures will be used in this thesis:

- $A(D_2)$ is the *area measure* (Lebesgue measure) on $\partial\mathcal{V}$.
- $V(D_3)$ is the *volume measure* (Lebesgue measure) on $\mathcal{V}$.
- $A^{\perp}_{\omega}(D_2) := \int_{D_2} \cos\Theta dA(x)$ is the *projected area measure*, where Θ is the angle between ω and the surface normal at x. This leads to $dA^{\perp}_{\omega}(x) = \cos\Theta dA(x)$.
  For $x \in \mathcal{V}^0$, $dA^{\perp}_{\omega}(x)$ is defined as the differential area measure on the plane through x that is perpendicular to ω. This entails that $dV(x) = dA^{\perp}_{\omega}(x)ds$ (see figure 2.1).

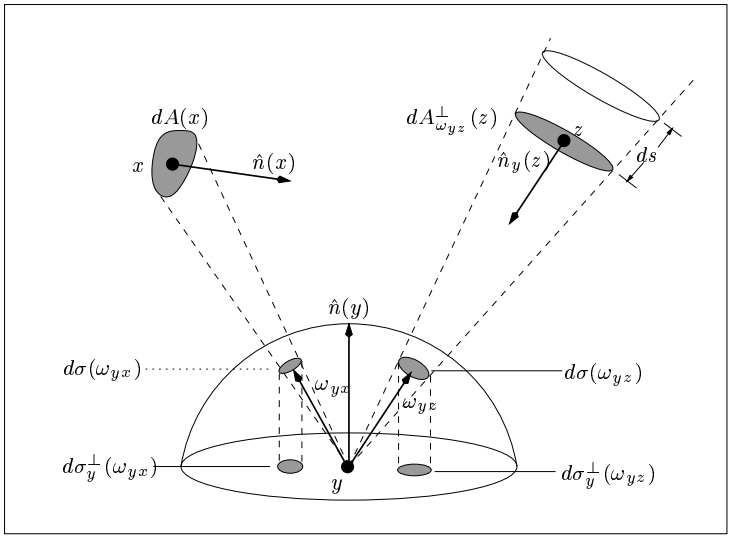


Figure 2.1: Relations of different measures.

- $\sigma(\Omega)$ is the *solid angle* of Ω, i.e. the ratio of the surface area of Ω on S^2 to the area of the unit square.
- $\sigma^{\perp}_x(\Omega) := \int_\Omega \cos\Theta d\sigma(\omega)$ is the *projected solid angle*, hence $d\sigma^{\perp}_x(\omega) = \cos\Theta d\sigma(\omega)$.

According to figure 2.1, area measure and solid angle measure are related through

$$d\sigma^{\perp}_y(\omega) = \frac{\cos\Theta'}{\|x-y\|^2} dA^{\perp}_{\omega}(x),$$

where $x, y \in \mathcal{V}$ and Θ' is the angle between ω and the normal at y for $y \in \partial\mathcal{V}$ and $\Theta' = 0$ for $y \in \mathcal{V}^0$.

2.2 Light Transport Equation

The following derivation of the light transport equation is based on the theory of *radiative transfer* [Cha60], which is concerned with statistical distributions of photons. For the purposes of photorealistic rendering, the underlying model can be simplified by the following assumptions [Arv93]:

- The number of photons is large and photons are infinitely small, i.e. the distribution of photons can be treated as a continuum.
- Photons do not interact with each other, i.e. interference is not accounted for.
- Polarization and diffraction are neglected.
- Collisions of photons with surfaces or particles of the medium are elastic.
- Photons are not influenced by external forces such as gravitation.

These assumptions entail that a photon is completely described by its *state* (x, ω, λ), where x is the position, ω the direction and λ the wavelength of the photon. This leads to a 6-dimensional *phase space*

$$\Psi = \mathbb{R}^3 \times S^2 \times \mathbb{R}^+$$

with an associated *phase space density* function

$$n : \Psi \times \mathbb{R}^+ \to \mathbb{R}_0^+,$$

such that $n(x, \omega, \lambda, t)$ is the density of photons with state (x, ω, λ) at time t. From now on the wavelength will be kept fixed, so that the λ-parameter can be dropped. The *phase space flux* ϕ is the rate at which photons cross a differential surface dA with directions in the differential solid angle $d\omega$ around the normal of dA in time dt:

$$\phi(x, \omega, t) := vn(x, \omega, t), \qquad \left[\frac{1}{m^2 \cdot sr \cdot s}\right]$$

where $v = ds/dt$ is the velocity of the photons. Since the speed of light substantially exceeds that of any macroscopic motion in the scene environment, it can be assumed that the phase space flux reaches equilibrium instantly. Therefore it is sufficient to only consider the stationary distribution of ϕ, which leads to a flux function $\phi \equiv \phi(x, \omega)$ independent of time.

In global illumination, *radiance* L is the fundamental quantity of interest. Measured in $\left[\frac{W}{m^2 \cdot sr}\right]$, $L(x, \omega)$ denotes the power passing perpendicularly through a differential surface around x in a differential solid angle around ω. Radiance can be related to phase space flux by

$$L(x, \omega) = h\nu\phi(x, \omega), \qquad \left[\frac{W}{m^2 \cdot sr}\right]$$

where $h = 6.626 \cdot 10^{-34}$Js is Planck's constant and ν is the frequency of the photons.
We will distinguish between *incident* and *exitant* radiance (see [Vea97]). Incident radiance L_i describes the radiance that arrives at a point x from direction ω, whereas exitant radiance L_o measures the radiance leaving a point x in direction ω. This distinction is mainly relevant on surfaces as in the volume $L_i(x, \omega) = L_o(x, \omega)$ holds[1].

To derive the light transport equation, we consider the radiance distribution for an arbitrarily chosen, fixed $V \times \Omega$ with $V \subseteq \mathcal{V}^0$ and $\Omega \subseteq S^2$. A number of physical processes affect the radiance distribution, either causing an increase or a decrease in radiative energy in $V \times \Omega$. These effects are depicted in figure 2.2 and quantitatively defined below:

The increase in radiative energy is due to

- *Emission:*

  $$E := \int_\Omega \int_V L_e^V(x, \omega)\, dV(x)\, d\sigma(\omega), \qquad [W]$$

  where the *volume emittance function* L_e^V, measured in $\left[\frac{W}{m^3 \cdot sr}\right]$, defines radiation sources in $\mathcal{V}^0$.

[1] The index will be omitted, whenever this equality is valid.

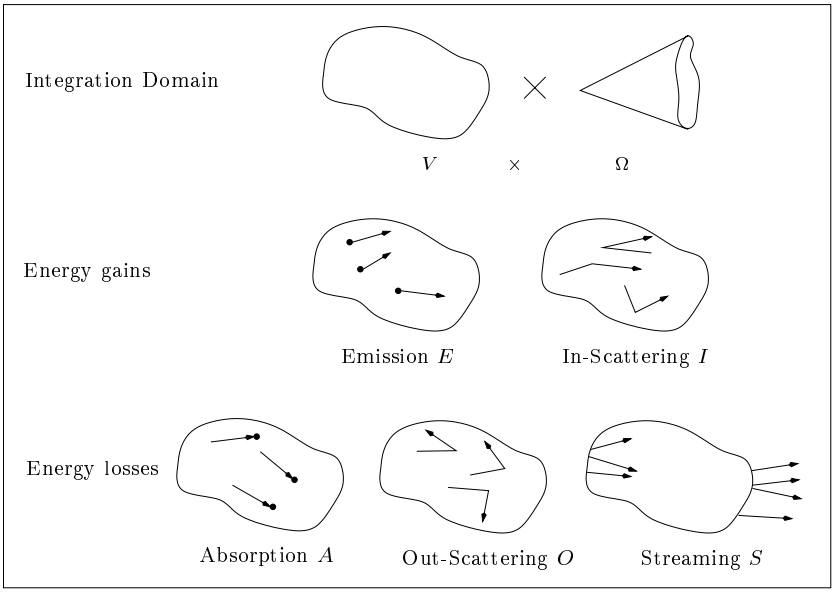


Figure 2.2: Physical processes that are responsible for changes of radiative energy in a certain region $V \times \Omega$ of phase space.

- *In-scattering*:

$$I := \int_\Omega \int_V \sigma_s(x) \int_{S^2} f_p(\omega, x, \omega') L(x, \omega') \, d\sigma(\omega') \, dV(x) \, d\sigma(\omega), \qquad [W]$$

  where $\sigma_s(x)$ is the *scattering coefficient*, measured in $[m^{-1}]$, which denotes the probability of a scattering interaction of a photon per unit distance. f_p is the *phase function*, measured in $[sr^{-1}]$, which describes the directional aspect of scattering in the medium. It determines the fraction of scattered radiance arriving at x from direction ω that is scattered in direction ω'. The phase function is normalized, i.e.

$$\int_\Omega f_p(\omega, x, \omega') d\sigma(\omega') = 1, \quad \forall \omega \in \Omega, x \in \mathcal{V}^0.$$

  and Helmholtz reciprocal, i.e.

$$f_p(\omega, x, \omega') = f_p(\omega', x, \omega), \quad \forall \omega, \omega' \in \Omega, x \in \mathcal{V}^0$$

  The latter relation allows to exchange incoming and outgoing direction. When sampling a direction (see section 3.3.1) we can thus trace rays from as well as towards the light sources.

Losses in radiative energy in $V \times \Omega$ are comprised of

- *Absorption*:

$$A := \int_\Omega \int_V \sigma_a(x) L(x, \omega) \, dV(x) \, d\sigma(\omega), \qquad [W]$$

  with the *absorption coefficient* $\sigma_a(x)$ with unit $[m^{-1}]$,

- *Out-scattering*:

$$O := \int_\Omega \int_V \sigma_s(x) \int_{S^2} f_p(\omega, x, \omega') L(x, \omega)\ d\sigma(\omega')\ dV(x)\ d\sigma(\omega) \qquad [W]$$

and

- *Streaming*

$$S := \int_\Omega \int_{\partial V} L(x, \omega)\ dA^\perp_\omega(x)\ d\sigma(\omega), \qquad [W]$$

where ∂V denotes the boundary of V. We define the normal at point x that is incorporated in $dA^\perp_\omega(x)$ to point outwards of the volume. Hence a positive S will cause a decrease in energy in $V \times \Omega$.

Using the theorem of Gauss, S can be transformed to

$$S = \int_\Omega \int_V \omega \cdot \nabla L(x, \omega)\ dV(x)\ d\sigma(\omega).$$

Since the system is in a stationary state, the radiance entering and exiting $V \times \Omega$ have to sum up to zero and thus can be combined to the balance equation for $V \times \Omega$:

$$S + A + O = E + I \tag{2.1}$$

Observe that the inner integration in I and O is over the whole sphere of directions. This includes radiance whose direction is in Ω both before and after scattering, which is neither contributing to an increase nor a decrease in radiative energy. In the balance equation this unwanted fraction cancels out, however, as it appears on both sides of the equation.
Because V and Ω have been chosen arbitrarily and the addends of 2.1 all have the same integration domain, the equality must also hold for the integrands. This leads to the *global balance equation*

$$\omega \cdot \nabla L(x, \omega) + \sigma_e(x) L(x, \omega) = L^V_e(x, w) + \sigma_s(x) \int_{S^2} f_p(\omega, x, \omega') L(x, \omega')\ d\sigma(\omega'), \tag{2.2}$$

for $x \in \mathcal{V}^0$ and $\omega \in S^2$, where $\sigma_e(x) := \sigma_a(x) + \sigma_s(x)$ is the *extinction coefficient*.
As boundary conditions for this integro-differential equation we define the exitant radiance L_o on surfaces through the local scattering equation for $x \in \partial\mathcal{V}$:

$$L_o(x, \omega) = L^S_e(x, \omega) + \int_{S^2} f_s(\omega, x, \omega') L_i(x, \omega')\ d\sigma^\perp_x(\omega'), \tag{2.3}$$

where f_s is the *bidirectional scattering distribution function* (BSDF) and L^S_e defines an emittance on surfaces. f_s describes the scattering properties of a surface and is analogous to $\sigma_s f_p$ in the medium. Energy conservation demands that

$$\int_\Omega f_s(\omega, x, \omega') d\sigma(\omega') \leq 1, \quad \forall \omega \in \Omega, x \in \partial\mathcal{V},$$

which guarantees that no energy is created through scattering. Note that equation 2.3 contains an integration over the whole sphere of directions, which allows reflective as well as transmissive surfaces.

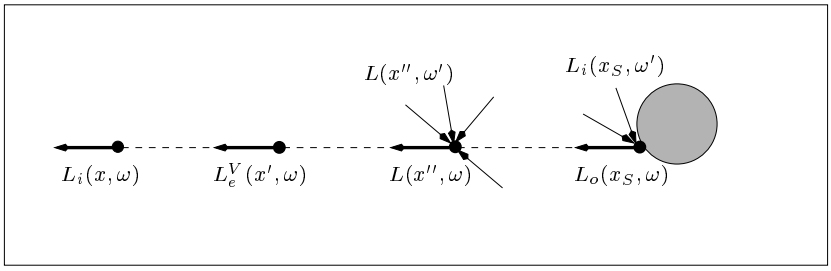


Figure 2.3: The integral form of the light transport equation expresses radiance as the sum of exitant surface radiance and in-scattered and emitted volume radiance.

2.2.1 Integral Form of the Light Transport Equation

From equation 2.2, an integral equation can be obtained by incorporating the boundary conditions 2.3. This is done by integrating equation 2.2 along a ray until a surface point is reached. We define the *path transmittance* τ between two points x and x' as

$$\tau(x, x') := e^{-\int_0^{\|x'-x\|} \sigma_e(x+s\omega_{xx'})ds}$$

Following [Arv93], L_i can be expressed as the integral equation:

$$\begin{aligned} L_i(x,\omega) &= \tau(x_S, x)L_o(x_S, \omega) + \int_0^{\|x_S - x\|} \tau(x', x) \\ &\quad \left[L_e^V(x', \omega) + \sigma_s(x') \int_{S^2} f_p(\omega, x', \omega')L(x', \omega')d\sigma_x^\perp(\omega')\right] ds, \end{aligned} \tag{2.4}$$

where $x' := x + s\omega_{xx_S}$ and $x_S := h(x, \omega)$ with the *ray casting function* $h(x, \omega)$, which determines the closest surface point from x in direction ω. Thus the radiance arriving at point x from direction ω is the sum of three components: The exitant radiance at x_S in direction ω, the emitted radiance in direction ω along the ray from x_S to x and the radiance that is scattered in direction ω along the same ray (see figure 2.3). All three factors are attenuated by the path transmittance.

Note that for h to be globally defined and equations 2.2 and 2.3 to be a complete description of the radiance distribution, the scene environment has to closed. This means that no radiation is allowed to leave $\mathcal{V}$, which can be achieved by completely surrounding $\mathcal{V}$ with a non-transmissive surface.

We define the following function and operators with x' and x_S as above:

- *Incident emittance*:

$$L_{i,e}(x, \omega) := \tau(x_S, x)L_e^S(x_S, \omega) + \int_0^{\|x_S - x\|} \tau(x', x)L_e^V(x', \omega)ds$$

- *Surface light transport operator*:

$$(\mathbf{T_S}L)(x, \omega) := \int_{S^2} f_s(\omega, x, \omega')L(x, \omega')d\sigma_x^\perp(\omega')$$

- *Volume light transport operator*:

$$(\mathbf{T_V}L)(x,\omega) := \int_0^{\|x_S - x\|} \tau(x',x)\sigma_s(x') \int_{S^2} f_p(\omega,x',\omega')L(x',\omega')\; d\sigma_x^{\perp}(\omega')ds$$

- *Light transport operator*:

$$\mathbf{T} := \mathbf{T_S} + \mathbf{T_V}$$

Now we can express L_i in operator notation:

$$L_i = L_{i,e} + \mathbf{T}L_i.$$

This equation can be written as the Neumann series

$$L_i = \sum_{i=0}^{\infty} \mathbf{T}^i L_{i,e} \tag{2.5}$$

given that $\|\mathbf{T}\| < 1$. This condition holds for all physically valid scene models where no perfect reflectors or transmitters exist.

2.3 Path Integral Formulation

To generate an image of size M, a set of *measurements* $I_1, \cdots, I_M$ has to be computed, where each I_j corresponds to a pixel value. A measurement is defined through the response of a *sensor* to radiance $L_i(x,\omega)$ incident upon it. This sensor is modeled as part of the scene geometry and defined through the *sensor responsivity*, a set of functions $W_e^{(j)}(x,\omega)$ measured in $[W^{-1}]$ (for more details see section 3.3.1). Combining $W_e^{(j)}$ and L_i in a scalar product yields the *measurement equation*:

$$I_j := \left\langle W_e^{(j)}, L_i \right\rangle := \int_{\mathcal{V}\times S^2} W_e^{(j)}(x,\omega)L_i(x,\omega)dA(x)d\sigma_x^{\perp}(\omega). \tag{2.6}$$

In this form, the computation of I_j requires the solution of the integral equation for L_i, which directly leads to recursive algorithms such as *path tracing* [Kaj86] or *particle tracing*, e.g. [SWH+95]. Alternatively, I_j can be expressed as a pure integral of the form

$$I_j = \int_{\Omega} f_j(\bar{x})d\mu(\bar{x}), \tag{2.7}$$

where $\bar{x}$ is a light transport path, Ω the set of all finite-length transport paths, μ a measure on Ω and f_j a function that assigns each path a *measurement contribution* (see below).

2.3.1 Path Space and Path Space Measure

A path of length k is represented by $k+1$ vertices x_i, where each x_i is either on a surface or in the volume. Let $b_i(l) \in \{0,1\}$ represent the value of the i-th bit of the binary representation of $l \in \mathbb{N}$, where b_0 denotes the least significant bit. Ω_k^l represents all paths of length k, where l specifies the combination of vertices on the surfaces or in the volume according to

$$\Omega_k^l := \{x_0x_1\cdots x_k \mid x_i \in \partial\mathcal{V}, \text{ if } b_i(l) = 1 \text{ and } x_i \in \mathcal{V}^0, \text{ if } b_i(l) = 0\},$$

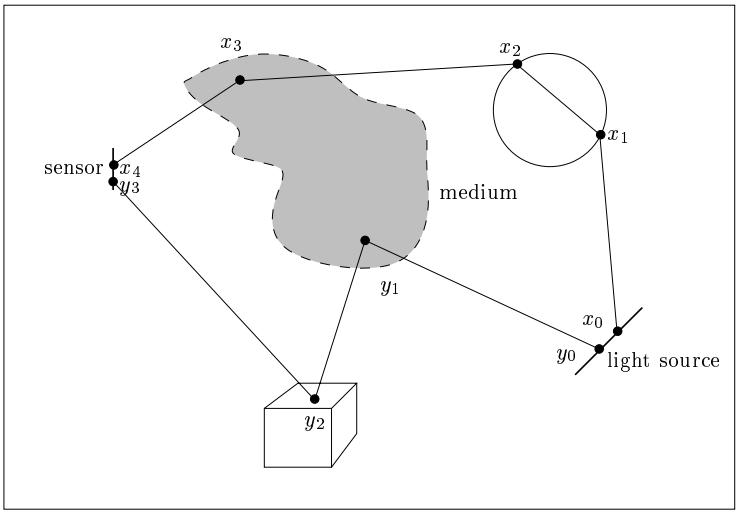


Figure 2.4: Two typical transport paths

with $1 \leq k < \infty$ and $0 \leq l < 2^{k+1}$. Figure 2.4 shows two transport paths; the upper transport path $\bar{x} = x_0x_1x_2x_3x_4$ is in $\Omega_4^{10111_b} = \Omega_4^{23}$, whereas the lower path $\bar{y} = y_0y_1y_2y_3$ is in $\Omega_3^{1101_b} = \Omega_3^{13}$. A measure μ_k^l is defined on Ω_k^l by

$$\mu_k^l(D) := \int_D \prod_{i=0}^{k} d\mu_{k,i}^l(\bar{x}),$$

where $D \subseteq \Omega_k^l$ and

$$d\mu_{k,i}^l(\bar{x}) := \begin{cases} dA(x_i), & \text{if } b_i(l) = 1 \\ dV(x_i), & \text{if } b_i(l) = 0 \end{cases}$$

for $\bar{x} = x_0 \cdots x_k$. This product measure is written in its differential form as

$$d\mu_k^l(\bar{x}) = \prod_{i=0}^{k} d\mu_{k,i}^l(\bar{x}).$$

Now we can define the *path space* Ω as the set of all finite-length paths:

$$\Omega := \bigcup_{k=1}^{\infty} \bigcup_{l=0}^{2^{k+1}-1} \Omega_k^l,$$

with the associated *path space measure* μ

$$\mu(D) := \sum_{k=1}^{\infty} \sum_{l=0}^{2^{k+1}-1} \mu_k^l(D \cap \Omega_k^l)$$

We omit the proof that μ is indeed a measure, as this can be verified with a simple check of the axioms.

2.3.2 Measurement Contribution Function

We will now introduce the "arrow-notation", which allows us to directly define the measurement contribution in terms of paths and path vertices. Let $x, y, z \in \mathcal{V}$. We define:

- $L(x \to y) := L(x, \omega_{xy})$
- $W_e(x \to y) := W_e(x, \omega_{xy})$
- $f_s(x \to y \to z) := f_s(\omega_{yx}, y, \omega_{yz})$
- $f_p(x \to y \to z) := f_p(\omega_{yx}, y, \omega_{yz})$
- $\tau(x \leftrightarrow y) := \tau(x, y)$
- *Visibility function* :

$$V_g(x \leftrightarrow y) := \begin{cases} 1 & \|x - y\| \leq \|x - h(x, \omega_{xy})\| \\ 0 & \text{otherwise,} \end{cases}$$

  which is one if x and y are mutually visible, i.e. if the connecting ray is not blocked by an object, and zero otherwise.

- *Geometric term* :

$$G(x \leftrightarrow y) = V_g(x \leftrightarrow y) \frac{|\omega_{yx} \cdot \hat{n}(x)||\omega_{yx} \cdot \hat{n}(y)|}{\|x - y\|^2},$$

  where $\hat{n}(x)$ is the surface normal at x for $x \in \partial\mathcal{V}$, or equal to ω_{xy} for $x \in \mathcal{V}^0$. $\hat{n}(y)$ is defined analogously.

To derive an expression for f_j in equation 2.7 we can expand the Neumann series 2.5 to

$$L_i = L_{i,e} + \mathbf{T_S} L_{i,e} + \mathbf{T_V} L_{i,e} + \mathbf{T_S}\mathbf{T_S} L_{i,e} + \mathbf{T_S}\mathbf{T_V} L_{i,e} + \mathbf{T_V}\mathbf{T_S} L_{i,e} + \cdots$$

Inserting this into 2.6 yields

$$I_j = \sum_{k=1}^{\infty} \sum_{l=1}^{2^{k+1}-1} \int_{\Omega} \quad L_e(x_0 \to x_1) G(x_0 \leftrightarrow x_1) \tau(x_0 \leftrightarrow x_1) \cdot \prod_{i=1}^{k-1} \hat{f}(x_{i-1} \to x_i \to x_{i+1}) G(x_i \leftrightarrow x_{i+1}) \tau(x_i \leftrightarrow x_{i+1}) \cdot W_e^{(j)}(x_{k-1} \to x_k) d\mu_k^l(x_0 \cdots x_k),$$

where

$$L_e(x \to x') := \begin{cases} L_e^S(x \to x') & x \in \partial\mathcal{V} \\ L_e^V(x \to x') & x \in \mathcal{V}^0 \end{cases}$$

and

$$\hat{f}(x \to x' \to x'') := \begin{cases} f_s(x \to x' \to x'') & x' \in \partial\mathcal{V} \\ \sigma_s(x') f_p(x \to x' \to x'') & x' \in \mathcal{V}^0 \end{cases} .$$

This leads to the desired form

$$I_j = \int_\Omega f_j(\bar{x}) d\mu(\bar{x}),$$

with

$$\begin{aligned} f_j(\bar{x}) \quad = \quad & L_e(x_0 \to x_1) G(x_0 \leftrightarrow x_1) \tau(x_0 \leftrightarrow x_1) \cdot \\ & \prod_{i=1}^{k-1} \hat{f}(x_{i-1} \to x_i \to x_{i+1}) G(x_i \leftrightarrow x_{i+1}) \tau(x_i \leftrightarrow x_{i+1}) \cdot \\ & W_e^{(j)}(x_{k-1} \to x_k) \end{aligned}$$

Equation 2.7 defines a measurement as an integral over the infinite-dimensional path space. This allows the application of general integration methods for the evaluation of 2.7, as will be described in the subsequent chapters.

Chapter 3

Sampling

This chapter describes some of the methods and techniques we employ in the global illumination algorithms of chapter 4. Section 3.1 gives a brief introduction to Monte Carlo integration and the variance reduction methods used in our implementation. Section 3.2 discusses how we model participating media, followed by a description of our path sampling algorithm in section 3.3.

3.1 Monte Carlo Integration

Our goal is to evaluate an integral of the form

$$I = \int_{\Omega} f(\bar{x}) d\mu(\bar{x}).$$

Common quadrature rules such as Newton-Coles rules or Gauss-Legendre rules suffer from the *curse of dimensionality*, i.e. the error bound for N sample points is $O(N^{-\frac{r}{d}})$ for $f \in C^r$, where d is the dimension of the integrand. A different approach is *Monte Carlo* (MC) integration, where I is approximated by generating N random samples $X_1, \ldots, X_N$ according to a suitable probability density function (pdf) p and computing the estimate

$$F_N = \frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)}. \tag{3.1}$$

The expected value $\mathbf{E}[F_N]$ is then equal to I and the convergence rate of the RMS error is $O(N^{-\frac{1}{2}})$ [KW86]. More specifically, the Hölder-inequality states that

$$\mathbf{E}[|I - F_N|] \leq \frac{\sigma(F_1)}{\sqrt{N}}, \tag{3.2}$$

where σ denotes the standard deviation. This shows that MC integration converges with the inverse square root of N, independent of the dimension of f.
Equation 3.2 suggests two principal methods for reducing the error of the estimate. One way is to increase the number of samples N. This is met with a proportional increase in computation time, however, while the error diminishes only with $\frac{1}{\sqrt{N}}$. Therefore the second method, reducing the variance $\sigma^2(F_1)$, has been a major focus of research. A number of such *variance reduction* methods have been developed, the most common of which are:

- Stratification

- Control Variates
- Adaptive Sampling
- Correlated Sampling
- Importance Sampling

For a detailed discussion of these methods see [KW86] and [Kel98]. We will briefly sketch the idea of importance sampling, as it will be used extensively in the algorithms of chapter 4. Consider the estimate of equation 3.1. If p is proportional to f, the variance of F_N would be zero. Since this would require knowledge of I (as a normalization constant for p) this method is not practical. However, if we can find a p that is "roughly proportional" to f, a substantial reduction of variance can be achieved. A common method is to use factors of f for p, which is especially beneficial if these factors dominate the shape of f. As a tradeoff to the decreased variance we are faced with a potential increase in computation time caused by generating samples according to p. Thus it is mandatory that p can be sampled efficiently, preferably with the *inversion method* [Sob91], which requires p to be easily integrable.

Multiple Importance Sampling

The gain in variance reduction is greatly dependent on how well p resembles f over the full integration domain. If the ratio f/p shows a high variation, variance can even increase as compared to uniform sampling [OZ99]. *Multiple importance sampling* has been developed to mitigate this effect [Vea97]. Here samples are generated according to a number of different pdfs. These samples are then combined with carefully chosen weights to yield a *combined sample estimate*. Weights are computed according to a *heuristic* that considers the probability densities of the generated samples. Thus it is possible to sample the integral with a number of different techniques, each suitable for a certain part of the integration domain. We will apply multiple importance sampling to the path integral 2.7 in section 4.1.

3.2 Modeling Participating Media

In our implementation we only consider non-emitting media, i.e. $L_e^V = 0$. Furthermore, we assume that the absorption coefficient σ_a and the scattering coefficient σ_s are isotropic, i.e. independent of direction and that the phase function f_p is homogeneous, i.e. independent of location. These are reasonable assumptions for media such as fog or smoke, as they can be approximated by a distribution of identical, spherical particles[1]. Thus a participating medium in a finite volume V is fully described by specifying $\sigma_a(x)$ and $\sigma_s(x)$ for each $x \in V$ and $f_p(x, \omega, \omega')$ for each $\omega, \omega' \in \Omega$.

3.2.1 Absorption and Scattering Coefficient

We distinguish two cases:

- *Homogeneous media* have constant σ_a and σ_s over the whole volume and provide a simple model for foggy or dusty atmospheres.

[1] Regular structures such as crystals are not accounted for in this model.

- *Inhomogeneous media* allow σ_a and σ_s to vary with location and can be used to model more complex phenomena such as clouds or curls of smoke. Data for clouds and smoke can be retrieved from physical measurements or computed through simulations. We use data of the latter kind, which is stored on a discrete three-dimensional grid with intermediate values being computed through trilinear interpolation.

3.2.2 Phase Function

We use the following scattering models:

- *Isotropic scattering* is analogous to perfectly diffuse reflection.
- *Hazy Mie scattering* simulates fog with a small density of dissolved water particles.
- *Murky Mie scattering* models fog with a higher density.
- *Rayleigh scattering* provides a scattering model for gas molecules.

Figure 3.1 shows the phase functions and a test image for all four scattering models. The phase function is depicted on the left, with two possible scattering directions ω_1' and ω_2' shown for an incoming direction ω. The value of the phase function is proportional to the length of the dashed lines for each ω_i'. Θ is the angle between incoming and outgoint direction. The image on the right is a test scene (see figure 3.4 for the actual image), where the light source emits light only straight downwards. We excluded all paths except those of the form light source $\rightarrow$ medium $\rightarrow$ surface $\rightarrow$ sensor and scaled the image in brightness. As illustrated, Mie scattering describes highly directional forward scattering. Rayleigh scattering is similar to isotropic scattering, slightly emphasizing the forward and backward directions.
The phase functions of Mie and Rayleigh scattering are too complex for efficient sampling. Therefore we approximate f_p with a convex combination of two Schlick base functions [BLS93], i.e.

$$f_p(\omega, \omega') = \alpha_1 f_{Schlick}(k_1, \Theta) + \alpha_2 f_{Schlick}(k_2, \Theta), \tag{3.3}$$

where α_i are weights, Θ is the angle between ω and ω', and k_i are parameters of the base functions

$$f_{Schlick}(k, \Theta) := \frac{1 - k^2}{(1 - k\cos\Theta)^2}. \tag{3.4}$$

Schlick derived the following parameter choice for approximating the above phase functions:

phase function	α_1	α_2	k_1	k_2
$f_{HazyMie}$	0.12	0.88	-0.50	0.70
$f_{MurkyMie}$	0.19	0.81	-0.65	0.91
$f_{Rayleigh}$	0.50	0.50	-0.46	0.46

The main advantage of the Schlick base functions is that they can be sampled using the inversion method (see section 3.3).

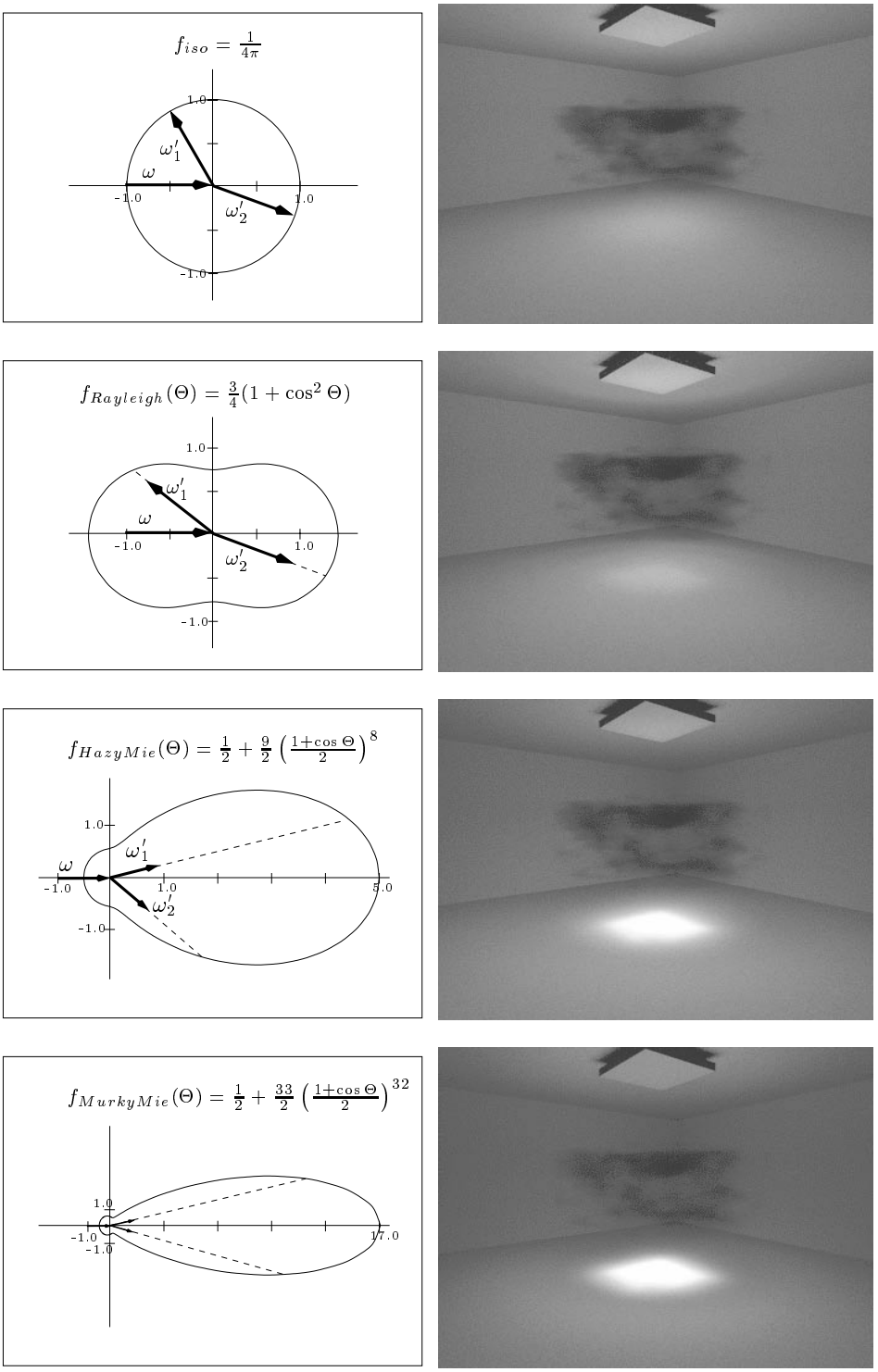


Figure 3.1: Scattering models used in our implementation.

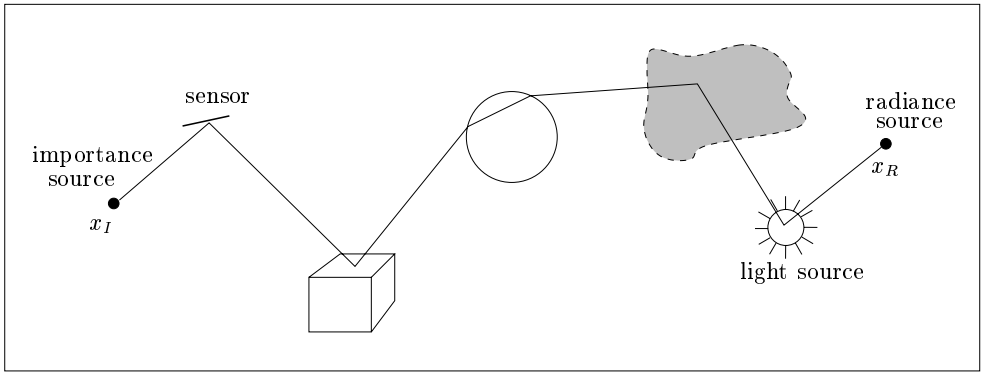


Figure 3.2: A typical transport path including the two artificial vertices x_R and x_I

3.3 Sampling Paths

To evaluate the path integral we need to generate paths and/or sections of paths according to an appropiate probability density p. Figure 3.2 shows a typical path, where we have included two additional vertices, x_I and x_R. These vertices are not part of the "physical" path, i.e. $x_R, x_I \notin \mathcal{V}$, but define artificial point sources for importance and radiance, respectively [VG94]. This formalism provides a concise description of how paths can be generated: Through a number of scattering and propagation events.

- Scattering chooses a direction at a given vertex according to pdf p_{scat}.
- Propagation determines the next interaction point in a given direction with pdf p_{prop}.

The pdf of the whole path will then be the product of the pdfs of the scattering and propagation events. Including x_R and x_I in our path description allows special cases at the ends of a path to be treated in the same way as intermediate vertices.
Recall that the integrand of equation 2.7 is the measurement contribution function f_j, consisting of various factors that are potential candidates for importance sampling. p_{scat} and p_{prop} have been chosen to incorporate as many of these factors as possible, while still enabling efficient sampling.

3.3.1 Scattering Event

Given vertex x, we want to generate a direction ω' according to p_{scat}. As illustrated in fig. 3.2, we have different types of vertices and each type has its own scattering characteristics:

- $x = x_R$: Scattering at x_R produces an artificial direction, which specifies the point on a light source that will be chosen in the next propagation step. In effect, we define a different light transport kernel for this point such that scattering and propagation from x_R produces the desired source emittance L_e. $p_{scat}(x_R, \omega')$ depends on the sampling strategy for light sources, e.g. $p_{scat}(x_R, \omega') = const$ generates a homogeneous source emittance.[2]

[2]This allows optimisation algorithms, e.g. for a large number of light sources, to be hidden in the scattering method for the radiance source.

- $x = x_I$: Scattering at x_I is analogous to scattering at x_R, only that here we generate W_e (in connection with the corresponding propagation event). For our implementation we use the pinhole camera model, specified through an eyepoint $x_E \in \mathcal{V}$ and a rectangular image plane. Each point x on the image plane is mapped to exactly one pixel of the image. The sensor responsivity for a pixel P of the image is defined as

$$W_e^P(x, \omega) := \delta(\omega - \omega_{x_E x}) f_P(x),$$

where δ is a Dirac delta function and f_P is a normalized filter function for pixel P.
More sophisticated camera models can enhance the realism of the produced images. Effects such as depth of field or motion blur can be taken into account by modeling the sensor with a lens and adapting the measurement equation to include an integration over a finite exposure time.

- x is on a light source: ω' is chosen by sampling according to the emissivity function of the light source, i.e. $p_{scat}(x, \omega') \propto L_e(x, \omega')$.

- $x \in \partial\mathcal{V}$: Scattering on surfaces is determined by the BSDF f_s. We use the Ward reflection model for isotropic scattering [War92] and also support singular scattering. Singular scattering idealizes highly directional scattering in that it allows only one scattered direction for each incident direction. When defining the material properties of a surface, the user can specify the probability P_{sing} for singular scattering, i.e. regular scattering occurs with probability $P_{reg} := 1 - P_{sing}$. This leads to

$$p_{scat}(\omega, x, \omega') = \begin{cases} P_{reg} p_{Ward}(\omega, x, \omega') & \text{for regular scattering} \\ P_{sing} \delta(\omega - \omega') & \text{for singular scattering,} \end{cases}$$

where ω is the incident direction, p_{Ward} is the pdf for regular scattering according to the Ward model and δ is the Dirac delta distribution. As we only consider terms of the form f/p, in case of singular scattering the delta distribution in the BSDF cancels out with the one in p.

- $x \in \mathcal{V}^0$: For vertices in the medium scattering is described by the phase function f_p, which is modeled with Schlick's base functions (see section 3.2). We derive the cumulative distribution function (cdf) for equation 3.4 as

$$F_{Schlick}(k, \Theta) = \frac{(1-k)(1+\cos\Theta)}{2(1-k\cos\Theta)}.$$

Inverting this expression leads to

$$\Theta = \arccos\left(\frac{2\xi + k - 1}{2k\xi - k + 1}\right),$$

where ξ is a uniformly distributed random variable in $[0, 1]$. This gives us the azimuthal angle for the scattered direction ω'. The polar angle for ω' is chosen uniformly, leading to the pdf

	homogeneous	inhomogeneous
$\tau(x, x')$	$e^{\sigma_e d}$	$e^{\int_0^d \sigma_e(x+t\omega)}dt$
pdf $p_m(x, \omega, x')$	$\sigma_e e^{\sigma_e d}$	$\sigma_e(x')e^{\int_0^d \sigma_e(x+t\omega)}dt$
cdf $F_m(x, \omega, x')$	$1 - e^{-\sigma_e d}$	$1 - e^{\int_0^d \sigma_e(x+t\omega dt)}$
sample distance	$d = -\frac{\ln(1-\xi)}{\sigma_e}$	$\ln(1-\xi) = \int_0^d \sigma_e(x+t\omega)dt$

Table 3.1: Sampling the position of the next interaction point x' starting from x. ξ is a uniformly distributed random variable in $[0, 1]$, $d := \|x' - x\|$ and $\omega := \omega_{xx'}$.

$$p_{Schlick}(k, \Theta) = \frac{1 - k^2}{4\pi(1 - k\cos\Theta)^2}.$$

As we use a convex combination of two Schlick bases (see equation 3.3), p_{scat} is given through

$$p_{scat}(\omega, x, \omega') = \alpha_1 p_{Schlick}(k_1, \Theta) + \alpha_2 p_{Schlick}(k_2, \Theta),$$

where Θ is the angle between ω and ω'.

3.3.2 Propagation Event

Given x and ω, the next interaction point x' is determined by sampling according to the pdf p_{prop}. For $x = x_R$ or $x = x_I$ the choice of x' depends on the light source or camera model (see section 3.3.1). For $x \in \mathcal{V}$ we want to make p_{prop} proportional to the path absorption τ and to sample p_{prop} with the inversion method[3]. Table 3.1 shows the relevant quantities for homogeneous and inhomogeneous media. We obtain the pdf p_m for sampling a point in the medium by normalizing τ. The cumulative distribution function F_m is derived by integrating p_m. Inverting F_m leads to the equation for the sample distance d.

Homogeneous Medium

Consider the case of a homogeneous medium in figure 3.3. First we determine the closest surface point $x_S = h(x, \omega)$ at distance $s := \|x_S - x\|$. Then we sample d according to the expression given in table 3.1. If $d \geq s$ we choose x_S as the new interaction point, otherwise we choose $x_d := x + d\omega$. $d < s$ occurs with probability

$$P_d := \int_0^s \sigma_e e^{-\sigma_e t} dt = 1 - e^{-\sigma_e s},$$

[3]Incorporating the inverse square of the geometric term yields a pdf that is computationally too expensive to sample from.

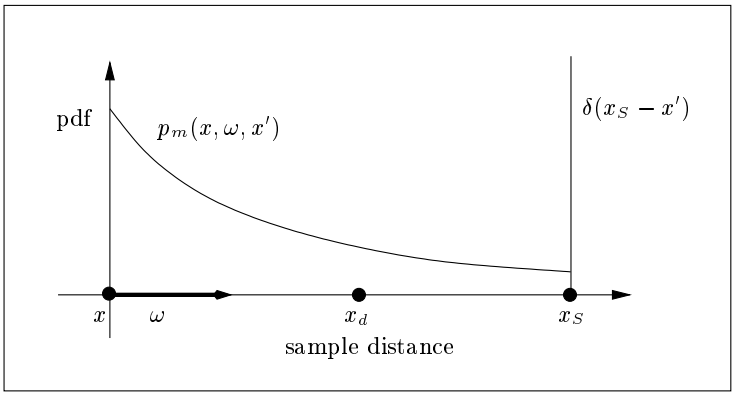


Figure 3.3: Sampling the next interaction point in a homogeneous medium.

thus the probability of choosing the surface point is

$$P_s := 1 - P_d = e^{-\sigma_e s}.$$

This leads to

$$p_{prop}(x, \omega, x') = \begin{cases} p_m(x, \omega, x') & \text{for interaction in the medium} \\ P_s \delta(x_S - x') & \text{for interaction on the surface} \end{cases}$$

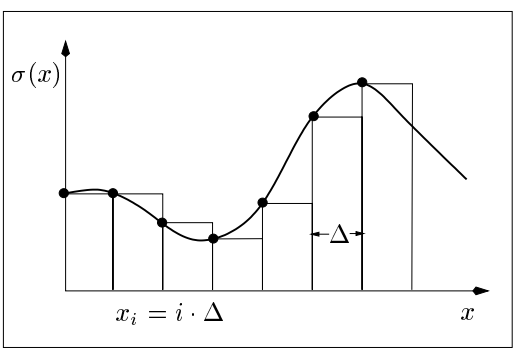
The probability P_d cancels out in the first row, as it is also the normalization constant for p_m in the interval $[x, x_S]$. The above equations imply that the whole ray segment from x to x_S is in the medium. If this is not the case, the pdfs have to be adapted accordingly.

Inhomogeneous Medium

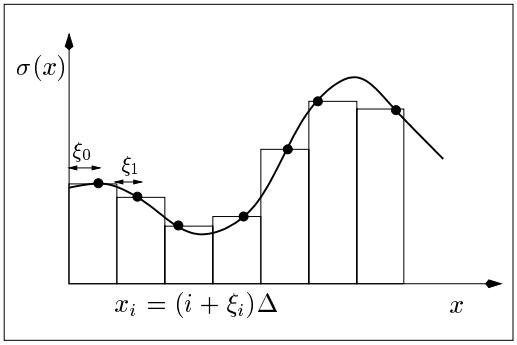
Sampling an inhomogeneous medium is more complicated. Here we do not have an explicit expression for the sampling distance d, but have to compute d from the implicit equation given in table 3.1. This is done with a *ray marching* algorithm [PH89], which accumulates σ_e along the ray (x, ω) until the threshold $\ln(1 - \xi)$ is reached or the surface point x_S is hit. In effect, a ray marching algorithm approximates a one-dimensional integral by dividing the ray into a number of disjunct segments and evaluating σ_e at certain points within each segment. We have implemented two versions of the algorithm:

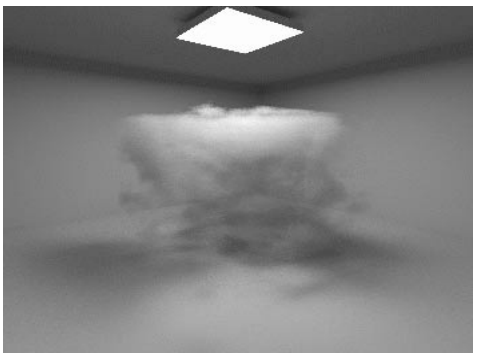
- *Equidistant sampling* traverses the ray with constant stepsize Δ, which is derived from a user-specified base stepsize Δ_{base}, taking into account various factors such as the extend of the medium and the distance to the observer.
- *Adaptive sampling* starts with the same Δ as equidistant sampling, but alters the stepsize during the traversal. If the difference between subsequent samples is bigger than a user-specified threshold, the stepsize is halved. If the difference falls below a lower threshold, Δ is doubled.

A preliminary experimental comparison of the two methods did not yield significant differences in the quality of the rendered images. Our test data set was very limited, however, and further

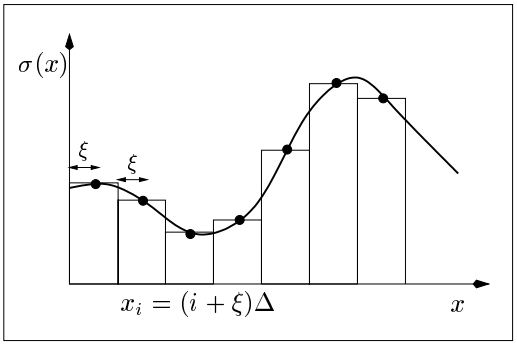


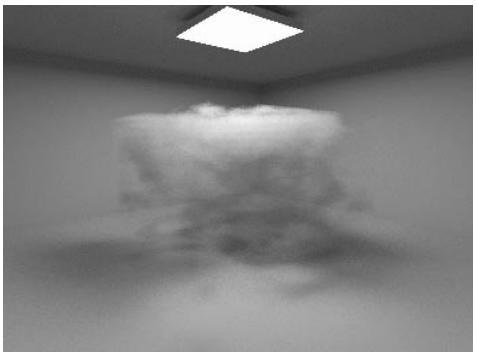
(a) Equidistant Sampling





(b) Stratified Sampling





(c) Random Offest Sampling

Figure 3.4: Different ray marching strategies. The left picture shows the sampling method, illustrated for box integration. On the right is an image generated with this method. Equidistant sampling clearly reveals aliasing artefacts which are no longer visible in the randomized versions of the ray marching algorithm.

investigations are necessary for a final judgement.

Equidistant and adaptive sampling both sample σ_e at the beginning of each ray segment. This produces visible artefacts due to aliasing as depicted in figure 3.4 (a). The explanation for the layers in the cloud is simple: Light is emitted downwards from the light source at the ceiling

and hits the cloud. As the traversal of the cloud data starts at the top surface of its cubic bounding box, the interactions in the medium occur roughly within the same horizontal layers, whose thickness is determined by the size of the ray segments, i.e. the stepsize of the traversal. Consequently, different transport paths that contribute to the same pixel are correlated. These effects can be eliminated by randomly perturbing the sample point within each ray segment, a method known as *jittering*. This leads to *stratified sampling*, a Monte Carlo method for numerical integration. While stratified sampling reduces aliasing (see 3.4 (b)), it is not a particular efficient sampling method for this kind of integration problem. Recall that Monte Carlo integration is particularly suitable for high-dimensional integrals with discontinuities in the integrand. Here we have a one-dimensional, rather smooth continuous function, favouring deterministic approaches. Therefore we implemented a combination of equidistant and stratified sampling. Instead of using independent random samples in each ray segment, we choose an initial random offset that is applied to all subsequent samples (see 3.4 (c)). This breaks the correlation of different transport paths (and hence reduces aliasing) but keeps the integration essentially deterministic and thus more efficient. We found an efficiency gain of about 30-45% for random offset sampling as compared to stratified sampling, which leads to a decrease in the total computation time of about 10% for the cloud scene of figure 3.4. As with the comparison of equidistant and adaptive sampling, further studies are necessary to quantify these benefits for a broader range of inhomogenous media.

Chapter 4

Simulation

In this chapter, we discuss two algorithms that evaluate the measurement equation 2.6 based on the path integral formulation:

- *Bidirectional Path Tracing* (BDPT) combines the ideas of path tracing and particle tracing and has been independently presented by Veach and Guibas [VG94] and Lafortune and Willems [LW93]. Although both algorithms are similar, they differ in the underlying theoretical framework. Veach and Guibas derive their algorithm from the path integral, whereas Lafortune and Willems base their work on the *global reflectance distribution function*.
- *Metropolis Light Transport* (MLT) is a new method [VG97] that applies the sampling technique known as *Metropolis sampling* to the path integral (see section 4.2).

4.1 Bidirectional Path Tracing

Bidirectional path tracing utilizes multiple importance sampling (see section 3.1) to compute an unbiased estimate of the path integral 2.7.

```
Path P_sensor, P_light
FOR i=1 TO NumOfPixels DO
   FOR j=1 TO Oversampling DO
      P_sensor = GenerateSensorPath (i)
      P_light = GenerateLightPath ()
      ConnectPaths (P_sensor, P_light)
      ComputePixelContribution (i)
   END
END
```

Figure 4.1: Structur of the bidirectional path tracing algorithm

Figure 4.1 shows the pseudocode of the algorithm. For each pixel we generate a pair of subpaths[1]; a sensor subpath that starts at the importance source x_I and a light subpath starting at the radiance source x_R. These subpaths are connected at certain vertices, possibly producing

[1] We will call a path a *complete* or *valid* path if it starts at the radiance source x_R and ends at the importance source x_I, otherwise the expression *subpath* or *path section* will be used.

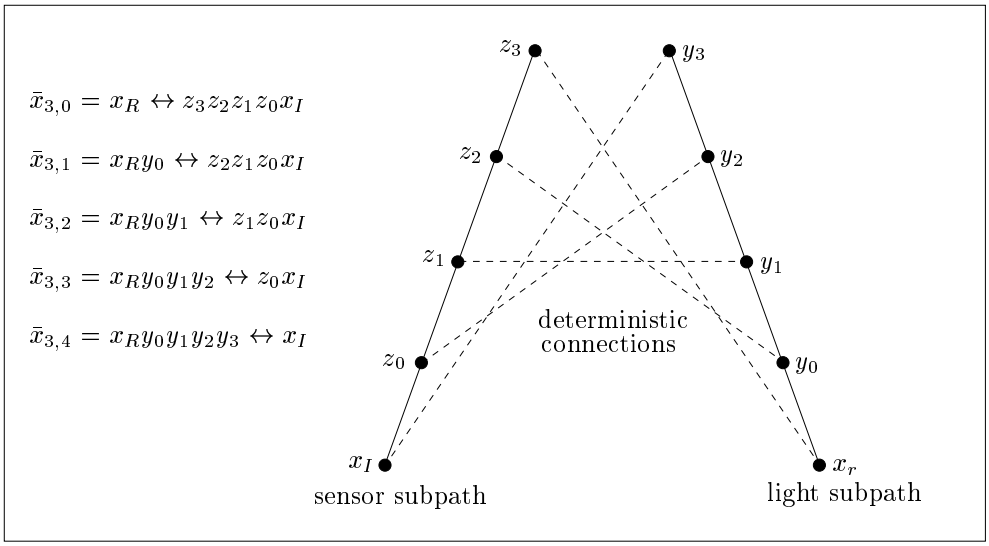


Figure 4.2: All methods to generate a path of length three. The $\leftrightarrow$ shows the position of the deterministic connection.

a number of valid paths whose contribution to the image is stored at the relevant pixel position. The subpaths are created independently of each other through a series of scattering and propagation events as described in section 3.3. Connecting the two subpaths is done as follows: Assume that we have generated a sensor subpath $\bar{z}_k = x_I z_0 \dots z_k$ and a light subpath $\bar{y}_k = x_R y_0 \dots y_k$. We want to compute all valid paths $\bar{x}_{k,i}$ of length k, where $i+1$ vertices are taken from the light subpath and $k-i+2$ are taken from the sensor subpath[2]. Figure 4.2 shows all possible configurations for a path of length three. We can distinguish three different types of connections:

- $i = 0$, i.e. $\bar{x}_{k,0} = x_R \leftrightarrow z_k \dots z_0 x_I$:
  This path is only valid if the sensor subpath ends on a light source.
- $0 < i < k+1$, i.e. $\bar{x}_{k,i} = x_R y_0 \dots y_i \leftrightarrow z_{k-i-1} \dots z_0 x_I$:
  Here we have to check with a shadow ray if y_i and z_{k-i-1} are mutually visible, i.e. $\bar{x}_{k,i}$ is valid if no object blocks the connection.
- $i = k+1$, i.e. $\bar{x}_{k,k+1} = x_R y_0 \dots y_k \leftrightarrow x_I$:
  y_k has to be a point on the sensor for this path to be valid.

Note that the connection step is purely deterministic, since once we have chosen the endpoints of the light and sensor sections, the connecting edge is fixed. The probability density $p_{k,i}$ for generating a path $\bar{x}_{k,i}$ is therefore the product of the pdfs of the two path sections $\bar{x}_{k,i}$ consists of. As an example, consider the path $\bar{x}_{3,2}$ of figure 4.2. Its pdf is the product of the pdf of the light section $x_R y_0 y_1$ and the pdf of the sensor section $x_I z_0 z_1$.
Each $p_{k,i}$ describes a distinct *method* for evaluating the path integral, e.g. $p_{k,0}$ corresponds to pure Monte Carlo path tracing. Methods have their characteristic strengths and weaknesses as illustrated in figure 4.3. The two upper diagrams show paths that are generated with method

[2]Note that with the two artificial vertices x_R and x_I a path of length k has $k+3$ vertices.

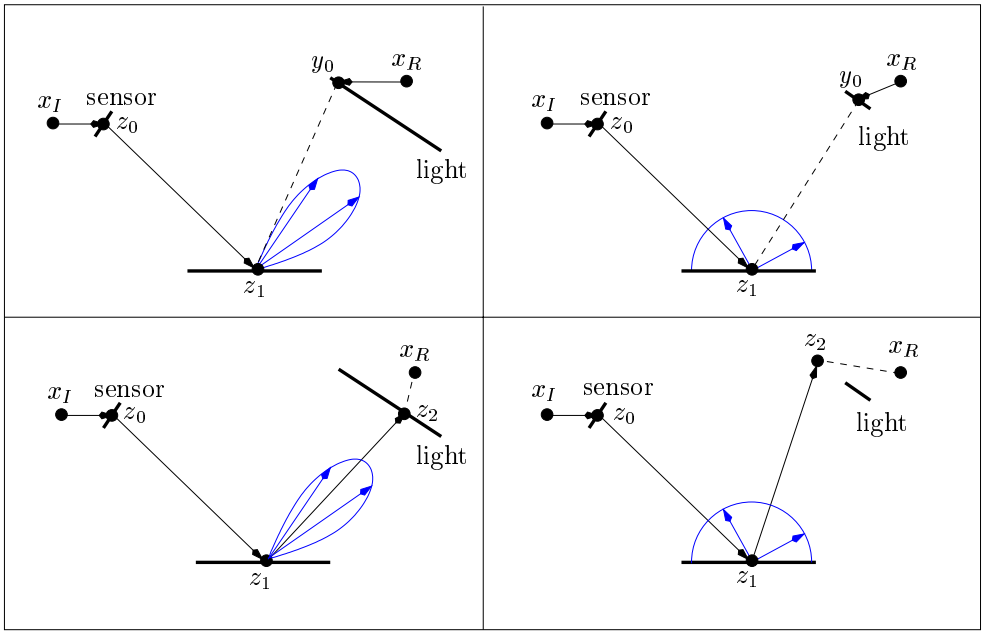


Figure 4.3: Schematic illustration of the behaviour of two sampling methods. Direct lighting (top row) is superior where path tracing (bottom row) fails and vice versa.

$p_{2,1}$, i.e. direct lighting calculation. In the left picture the surface is highly specular and the solid angle subtended by the light source at z_1 is relatively large. Thus by sampling a point y_0 on the light source and connecting it with z_1, we have a high probability that the BSDF at z_1 will have a low value or even be zero. This entails a low measurement contribution and therefore $f_j(\bar{x}_{2,1})/p_{2,1}(\bar{x}_{2,1})$ will be small for such paths. There is, however, a small set of points on the light source which leads to valid paths with a high measurement contribution. Consequently, the estimate of method $p_{2,1}$ will have a high variance for this kind of lighting situations. On the other hand, method $p_{2,1}$ performs much better in the upper right picture, where we have a perfectly diffuse surface and a small light source. Here the BSDF is constant for all directions, so that the position of y_0 does not significantly affect the measurement contribution. Thus direct lighting will produce a low-variance estimate for these configurations.

The exact opposite is true for method $p_{2,0}$, shown in the lower row of diagrams. This technique will perform poorly on right picture, because sampling according to the BSDF at z_1 will generate many reflected rays that will not hit the light source and thus have zero measurement contribution. In the situation of the left picture, however, variance of the estimate will be much lower, as the reflected rays will all hit the light source in the same small region and thus have a similar measurement contribution.

The essence of bidirectional path tracing is that it combines all $p_{k,i}$ in a multiple sample estimate, which preserves the strengths of each single method. The estimate can be expressed in a weighted sum as

$$F_1 = \sum_{k=1}^{k_{max}} \sum_{i=0}^{k+1} w_{k,i}(\bar{x}_{k,i}) \frac{f_j(\bar{x}_{k,i})}{p_{k,i}(\bar{x}_{k,i})}, \tag{4.1}$$

with weights $w_{k,i}$. The goal is to choose the weights such that the variance of F_1 is minimal. Veach and Guibas have presented various heuristics to compute the weights of 4.1 [Vea97]. In accordance with their results and the analysis of Kollig in [Kol99], we found the *power heuristic* to be most effective. The $w_{k,i}$ combine all pdfs for generating a path of length k in

$$w_{k,i} = \frac{p_{k,i}^{\beta}}{\sum_j p_{k,j}^{\beta}}.$$

We obtained the most satisfactory results with the parameter choice $\beta = 2$; $\beta = 1$ leads to the *balance heuristic*, which in general is slightly less effective.

4.2 Metropolis Light Transport

Metropolis sampling has been developed in the context of computational physics, where high-dimensional integration problems dominate the computations [MRR+53]. The first application of this sampling method to global illumination, the *Metropolis light transport* (MLT) algorithm, has been presented by Veach and Guibas in [VG97]. Contrary to bidirectional path tracing, the MLT algorithm does not estimate single pixel values, but tries to approximate the intensity distribution of the whole image at once. To do so, the algorithm generates a random walk $\bar{x}_0, \bar{x}_1, \ldots$ through path space, such that all $\bar{x}_i$ are valid paths from a light source to the sensor. For each $\bar{x}_i$ it determines the pixels of the image, whose filter support contains the image location of $\bar{x}_i$, and updates their value accordingly. The brightness of a pixel thus depends on the number of paths that cover its filter support. The desired image is obtained by distributing the paths according to their contribution to the final image, specified in the image contribution function $f(\bar{x})$ (see below). To generate a random walk distributed according to $f(\bar{x})$, we use Metropolis sampling, also known as M(RT)^2 sampling.
Section 4.2.1 briefly reviews the basic idea of Metropolis sampling and section 4.2.2 explains how it can be adapted to the path integral. Then we describe the MLT algorithm in more detail and present our extensions to incorporate participating media.

4.2.1 Metropolis Sampling

Let Ω be a state space, $f : \Omega \to \mathbb{R}_0^+$ a non-negative function and $X_0 \in \Omega$ an inital state. The Metropolis sampling algorithm generates a Markov chain $X_0, X_1, \ldots$, which in the limit is distributed proportionally to f.
X_{i+1} is a random variation of X_i according to the *transition function* K, where $K(X \to Y)$ is the pdf for the transition from X to Y. We can derive the pdf for X_{i+1} as

$$p_{i+1}(X) = \int_{\Omega} K(Y \to X) p_i(Y) d\mu(Y),$$

where p_i is the density for X_i and μ is a measure on Ω. This recursively defines a series of density functions, which converges to the stationary distribution p, given that $K(X \to Y) > 0$, $\forall X, Y \in \Omega$. The idea of Metropolis sampling is to construct a transition function K such that the resulting stationary distribution is proportional to the given f.
A sample X_{i+1} of the Markov chain will be generated by first proposing a *mutation* Y_i of X_i according to the tentative transition function T, where $T(X \to Y)$ denotes the probability density for mutating X to Y. Then we compute the *acceptance probability* α and set $X_{i+1} := Y_i$

with probability $\alpha(X_i \to Y_i)$ and $X_{i+1} := X_i$ with probability $1 - \alpha(X_i \to Y_i)$. The transition function $K(X_i \to Y_i)$ is then proportional to $f(X_i)T(X_i \to Y_i)\alpha(X_i \to Y_i)$, if $p_i \propto f$. The acceptance probability is subject to a condition known as *detailed balance*:

$$f(X)T(X \to Y)\alpha(X \to Y) = f(Y)T(Y \to X)\alpha(Y \to X),$$

which ensures that the stationary distribution p is proportional to f. Making $\alpha(X \to Y)$ and $\alpha(Y \to X)$ as large as possible leads to fast convergence of the p_i, hence we choose

$$\alpha(X \to Y) := \min\left\{1, \frac{f(X)T(X \to Y)}{f(Y)T(Y \to X)}\right\}$$

Note that the samples generated by the Metropolis sampling algorithm are highly correlated. While this does not affect the expect value, we can no longer apply common Monte Carlo variance tests.

4.2.2 Metropolis Sampling for the Path Integral

Recall that we want to evaluate the measurement equation

$$I_j = \int_\Omega f_j(\bar{x})d\mu(\bar{x}). \tag{4.2}$$

The image contribution function f is defined through

$$f_j(\bar{x}) = h_j(\bar{x})f(\bar{x}),$$

where h_j contains all factors of f_j that depend on the sensor $W_e^{(j)}$ and $f(\bar{x}) := 0$ for all incomplete paths. Let b denote the total radiant power passing through the image plane, i.e.

$$b := \int_\Omega f(\bar{x})d\mu(\bar{x}).$$

We can now rewrite equation 4.2 as

$$\begin{aligned} I_j &= \int_\Omega h_j(\bar{x})f(\bar{x})d\mu(\bar{x}) \\ &= b\int_\Omega h_j(\bar{x})\frac{f(\bar{x})}{b}d\mu(\bar{x}). \end{aligned}$$

If we interpret $p := f/b$ as a density distribution (with normalization constant $1/b$) we get

$$I_j = \mathbf{E}\left[\frac{b}{N}\sum_{i=1}^{N} h_j(\bar{x}_i)d\mu(\bar{x}_i)\right]$$

for a path sequence of length N. Thus to compute an image we need to evaluate b and generate a random walk in Ω according to the density f/b.

One problem of Metropolis sampling is that the samples X_i are only asymptotically distributed according to p, yet in the estimate we treat all samples as if their pdf was p. This causes an initialization error, which can be diminished by discarding a suitable number k of samples at the beginning of the random walk, until the p_i have approximately converged to p. Determining k is difficult, however; if k is too large, we will needlessly waste computation time, if k is too

```
Path x̄, ȳ
double b
Image I
ClearImage(I)
b = ComputeImageBrightness()
x̄ = GetInitialPath()
FOR i=1 TO NumOfMutations DO
    ȳ = Mutate (x̄)
    α = AcceptanceProb (x̄ → ȳ)
    IF Random() < α THEN x̄ = ȳ
    RecordSample (I, x̄, b)
END
```

Figure 4.4: Metropolis light transport algorithm

small, the estimate will greatly depend on the initial state X_0, which introduces to so called *start-up bias*. We will show in section 4.2.3 how we can avoid start-up bias by carefully choosing the initial path of the random walk.

Figure 4.4 gives an overview of the MLT algorithm. First the total image brightness b is estimated and an intial path $\bar{x}$ is chosen (see below). This path is then iteratively mutated according to one of several mutation strategies (see section 4.2.4) and the corresponding acceptance probability is computed. The mutated path is either accepted or rejected and the contribution of the path is recorded at the relevant pixel positon of the image.

4.2.3 Initialization

To compute b we adapt the multiple sample estimate 4.1 as

$$b_1 = \sum_{k=1}^{k_{max}} \sum_{i=0}^{k+1} w_{k,i}(\bar{x}_{k,i}) \frac{f(\bar{x}_{k,i})}{p_{k,i}(\bar{x}_{k,i})},$$

i.e. we use bidirectional path tracing to approximate the total image brightness. For all scenes that we rendered with the MLT algorithm (see chapter 5) a small number ($N \in [10^3, 10^5]$) of sensor and light subpath pairs proved sufficient for a reasonable approximation b_N of b.
The initial path for the Metropolis stage of the algorithm is selected randomly from the set of paths used in calculating b_N. The pdf for choosing $\bar{x}$ as initial path is

$$p_N(\bar{x}) = \frac{f(\bar{x})}{b_N}.$$

This means that paths with a high image contribution are more likely to be chosen as intial path. Note that $\lim_{N\to\infty} p_N = f/b$ and that if $\bar{x}_0$ was distributed according to f/b, all subsequent samples would be too. Thus by increasing N we not only improve the accuracy of b, but also diminish the initialization error. Note that the MLT algorithm using the above initialization step is unbiased, i.e. when averaging over many runs the result will converge towards the correct solution.

4.2.4 Mutation Strategies

The first step in generating the next sample of the random walk is to propose a mutation. Metropolis light transport uses a set of different *mutation strategies* and together they specify the

tentative transition function T. Veach and Guibas identified a number of desirable properties for mutation strategies in [VG97]:

- *Ergodicity*
  To ensure convergence of the random walk to the same stationary distribution independent of the initial state, we require $T(X \to Y) > 0$ whenever $f(X) > 0$ and $f(Y) > 0$.

- *High acceptance probability*
  A low acceptance probability leads to long path sequences with the same sample, which increases the variance of the estimate.

- *Large changes to the path*
  To sufficiently cover the path space even for short random walks, mutation strategies should include substantial changes the current path.

- *Changes to the image location*
  As we are interested in the intensity distribution on the image plane, frequent changes to the sensor point are desirable.

- *Low cost*
  The efficiency of a mutation strategy depends on the amount of computation time required to generate the mutation. Since ray intersections are the most expensive operations in doing so, we want to minimize their number.

Obviously, some of these properties compete with each other, e.g. large changes to the current path will often require many ray intersections and thus entail high computational cost. Therefore, mutation strategies have to be chosen carefully, always keeping in mind the tradeoffs between effectiveness and efficiency. By using a set of different mutation strategies, each specifically designed to meet a subset of the above goals, we can balance these tradeoffs. Note that at least one mutation strategy has to be ergodic to assure convergence to a unique stationary distribution.
In the following, $\bar{x} = x_R x_0 \dots x_k x_I$ denotes the current transport path, i.e. x_0 is a point on a light source and x_k is a point on the sensor. Similarly, $\bar{y} = x_R y_0 \dots y_l x_I$ is the proposed mutation of $\bar{x}$.

Bidirectional Mutation

Bidirectional mutation is central to the algorithm. Since any valid path is created with a probability bigger than zero it ensures ergodicity and is also responsible for large changes to the current path. Generating the mutated path can be divided into three steps as depicted in figure 4.5. First we choose two vertices of $\bar{x}$ and delete the section between these vertices. This creates a sensor subpath and a light subpath, both of which are extended through a number of scattering and propagation events. Finally we connect the two subpaths as described in section 4.1 to obtain $\bar{y}$. The mutation is immediately rejected, i.e. the acceptance probability is set to zero, if the connection is not valid. Note that if x_R and x_I are chosen in the first step, a completely new path is generated.
The deleted section is determined by importance sampling of all a priori known factors of the acceptance probability as described in [Kol99]. The length of the inserted path section l_{ins} is sampled according to an empirical pdf p_{ins}, depending on the length l_{del} of the deleted section:

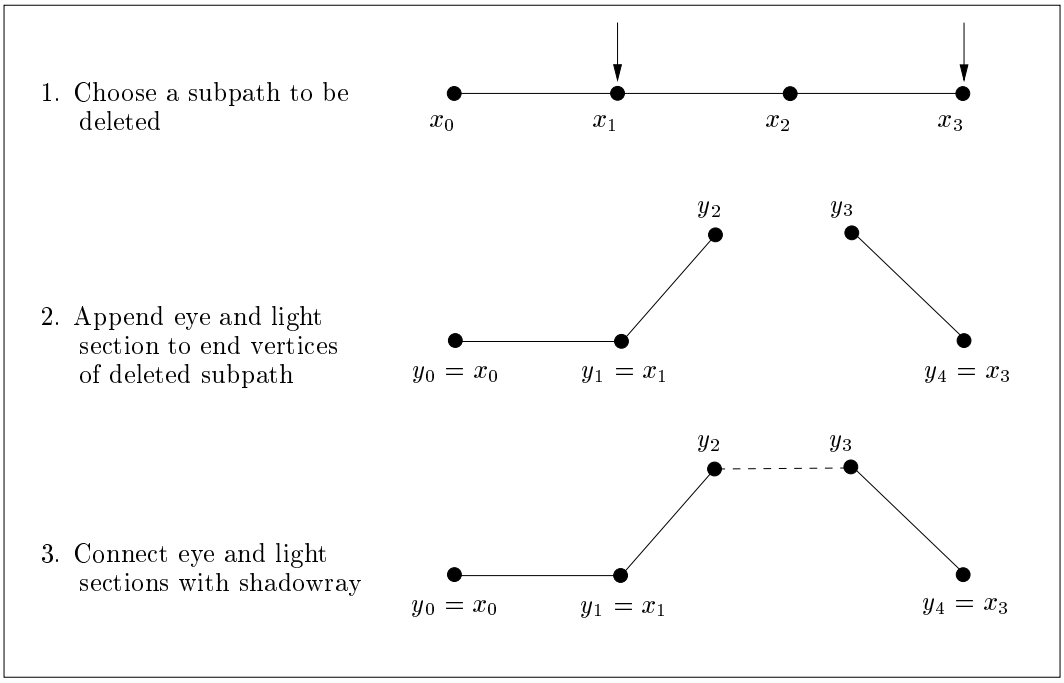


Figure 4.5: Three steps of a typical bidirectional mutation.

- $p_{ins}(l_{del}) := 0.5$
- $p_{ins}(l_{del} \pm 1) := 0.15$
- $p_{ins}(l_{del} \pm i) := 0.2 \cdot 2^{-i}$ for $i > 1$

As the new section is created by extending both the sensor and the light subpath, we choose the number of added vertices n_{sensor} and n_{light} on each section uniformly, subject to $n_{sensor} + n_{light} = l_{ins}$.

Perturbations

These mutation strategies exploit the fact that small variations to the path most likely lead to similar image contributions and hence a high acceptance probability. We distinguish two types of perturbations:

- *Scattering perturbations* displace the direction vector at a certain vertex.
- *Propagation perturbations* displace the interaction point along a certain ray segment.

The mutated path is then created by retracing the original path while preserving the scattering characteristics. This means that the mutated path $\bar{y}$ will consist of the same number of vertices as the original path $\bar{x}$ and each vertex of $\bar{y}$ will employ the same scattering mode (singular or regular) as the corresponding vertex of $\bar{x}$.

1. Scattering Perturbations

   We have implemented two scattering perturbations:

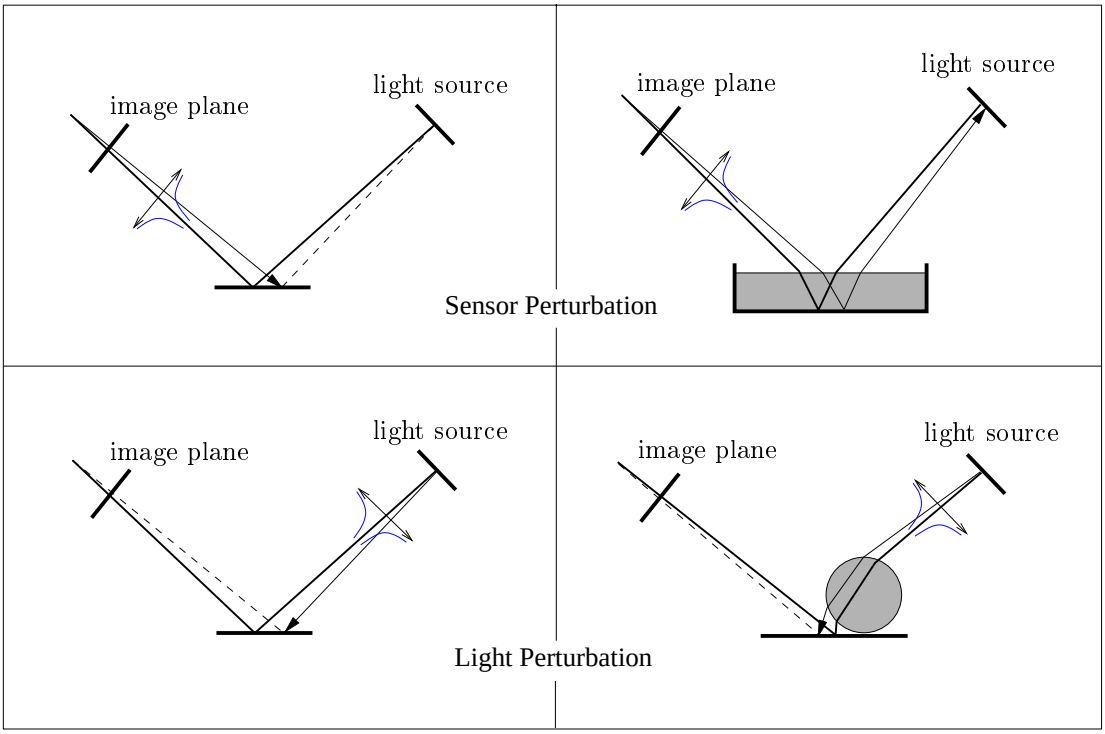


Figure 4.6: Sensor and light perturbation. The distribution of the directional displacement is indicated at the perturbed edge. The simple case without singular scattering is depicted on the left. Paths that include singular scattering are shown on the right.

- *Sensor perturbation* alters the location x_k on the image plane by moving it a distance D in a uniformly distributed direction. The distance D is sampled according to the pdf

$$p(D) \propto \frac{1}{D}, \quad D \in [D_{min}, D_{max}],$$

  where D_{min} and D_{max} specify the minimal and maximal distance, respectively. Integrating and inverting $p(D)$ (with an adequate normalization constant) leads to

$$D = D_{max} e^{-\ln(\frac{D_{max}}{D_{min}})\xi},$$

  with ξ uniformly distributed in $[0, 1]$.

- *Light perturbation* perturbs the direction vector from x_{k-2} to x_{k-1} using the same distribution as sensor perturbation. The path is then retraced towards the eye and the new sensor location y_k is determined through the ray from y_{k-1} to the eye point.

Figure 4.6 illustrates the two types of scattering perturbations. The top right picture shows a sensor perturbation of a path with two singular scatterings. Starting from the eye the ray hits the surface of the water and is singularly refracted towards the diffuse pool bottom. As the subsequent scattering is singular as well, we enforce this scattering mode on the corresponding vertex of $\bar{y}$, in this case creating a completely new path.

2. Propagation Perturbation

   This mutation strategy is specificially designed for participating media. If $x_{k-1} \in \mathcal{V}^0$, it moves this interaction point along the line from x_{k-2} to x_{k-1} and connects the new point y_{k-1} with the eye to determine the sensor point y_k. x_{k-1} is moved in either direction along $x_{k-2}x_{k-1}$, using the same distribution as for the scattering perturbations (see figure 4.7). If y_{k-1} falls outside the medium or $x_{k-1} \notin \mathcal{V}^0$ the mutation is rejected.

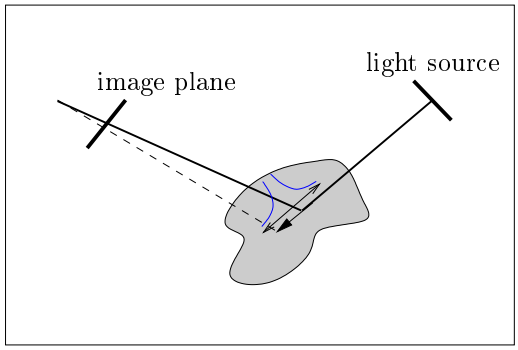


Figure 4.7: Propagation perturbation. The interaction point is displaced according to the indicated distribution.

In a sense, scattering and propagation mutations are complementary. The first perturbs a direction hoping to obtain a similiar interaction point, while the latter perturbs an interaction point hoping to obtain a similar direction. The idea of both is to sample path space locally. Once an important path has been found, neighbouring paths are explored as well. This is especially beneficial for bright areas of the image, such as caustics.
An important feature of perturbations is that they alter the image location. This leads to a better distribution of paths over the image plane and significantly reduces the variance of the generated images.

Chapter 5

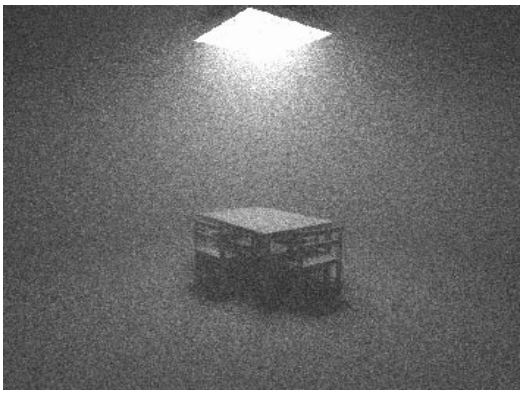
Results and Discussion

We have implemented both bidirectional path tracing and Metropolis light transport in *C++*, based on the experimental ray tracing kernel *McRender* [Kel98]. Central to both algorithms is a class library that provides the sampling methods described in chapter 3. The formalism introduced in section 3.3 directly matches the implementation. Camera model, emittance functions and scattering models are treated as black boxes with clearly defined interfaces and can easily be extended or improved.
We will first discuss some aspects of Metropolis light transport and then compare the algorithm with bidirectional path tracing. More information on the latter in the context of participating media can be found in [LW96] and [Arn97].

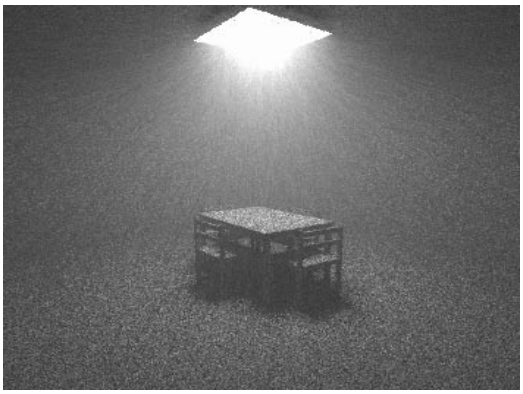
5.1 Metropolis Light Transport

After computing the total image brightness in the initialization step, the actual Metropolis stage generates the random walk through path space. We split this second stage into two phases; the contribution of paths of length one is calculated in a separate pass, as these can be estimated more efficiently using standard ray tracing techniques. Metropolis sampling is thus only applied for paths of length two or more.
As described in section 4.2.4, we use a set of different mutation strategies to propose a mutation of the current path of the random walk. For each mutation we randomly choose a mutation strategy by sampling according to a discrete pdf that assigns each strategy a constant probability. For scenes without participating media, Kollig investigated the effects of different pdfs on the RMS-error and maximum pixel distance[1] for a series of test images [Kol99]. He found that a balanced pdf, i.e one that uses the same weight for each mutation strategy produces the most satisfactory results. We can generally confirm these findings for the extended set of mutation strategies. Since we have included the propagation perturbation, there are additional aspects that need to be considered. Recall that this mutation strategy can only be applied if the first interaction point of the transport path from the sensor lies within a medium. Thus the acceptance probability will highly depend on the scene environment. If only a small fraction of the relevant scene volume is occupied by a medium and/or the medium is optically thin, the acceptance probability will be low on average, which increases the variance of the image. On the other hand, propagation perturbation is a computationally cheap mutation strategy,

[1] i.e. the maximal L_2-Norm distance of the color value of any pixel with respect to a reference image



Mutation Set 1: Bidirectional Mutation



Mutation Set 2: Bidirectional Mutation & Propagation Perturbation

Mutation Set 3: Bidirectional Mutation & Scattering Perturbations

Mutation Set 4: Bidirectional Mutation, Scattering & Propagation Perturbations

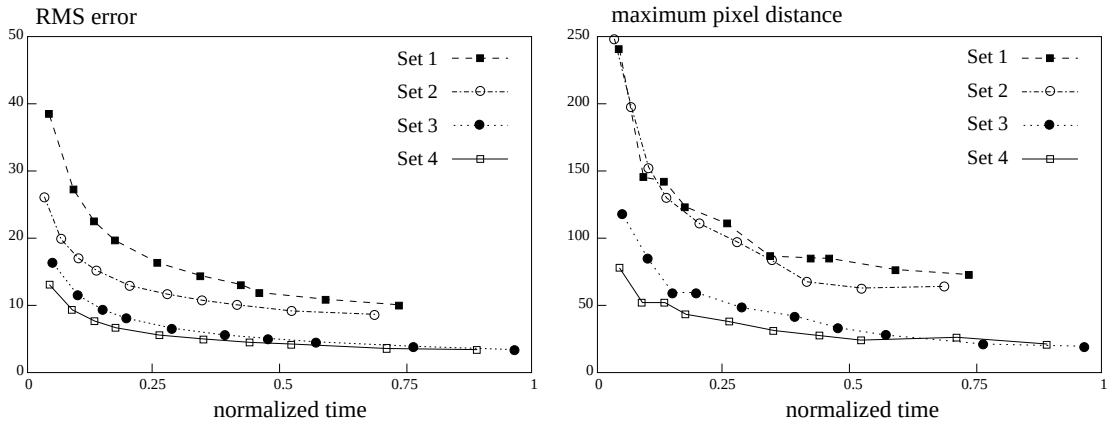


Figure 5.1: Four different sets of mutation strategies, where all strategies are weighted equally within each set. The images correspond to a normalized time of 0.25.

at most requiring one ray intersection and one occlusion test. We have analysed the performance of different mutation strategies for a test scene with a room that is completely filled with a homogeneous medium (see figure 5.1). The mutation set consisting of all four mutation

strategies clearly produces the best results. In particular, the RMS error is about 10-20% lower as compared to mutation set 3. This gain in efficiency is specific to this scene, however. The decision whether to include propagation perturbation in the set of mutation strategies or not will depend on the input data.
A potential weakness of MLT is the stratification of the samples on the image plane. In contrast to BDPT, where samples are evenly stratified, we have no explicit control over the distribution of paths across the image plane. Designing a mutation strategy that specifically addresses this issue is problematic: recall that a mutation must always be symmetric, i.e. we need a positive transition probability for mutating X to Y and vice versa (see section 4.2.1). This symmetry in connection with the locality of a mutation makes it difficult to achieve a globally even distribution of paths on the image plane.

5.2 Comparison of BDPT and MLT

Bidirectional path tracing and Metropolis light transport both evaluate the path integral 2.7, i.e. are based on the same physical simulation model. For both algorithms there are no restrictions in the geometry and scattering models that can be supported. The memory requirements of both methods are similar: Apart from the scene description and the image matrix, there is minimal additional overhead. In particular, no complex data structures are required and no data needs to be stored on the geometrical primitives (as with radiosity algorithms, for example). In our comparison we will therefore concentrate on the quality of the images or, in other words, on the computation time needed to generate images of comparable quality.
Figure 5.2 gives an example of a volume caustic, generated by light being refracted from a glass sphere into an isotropic homogeneous medium. The ceiling is indirectly illuminated through reflection from the sphere and the medium. These images clearly illustrate the above mentioned

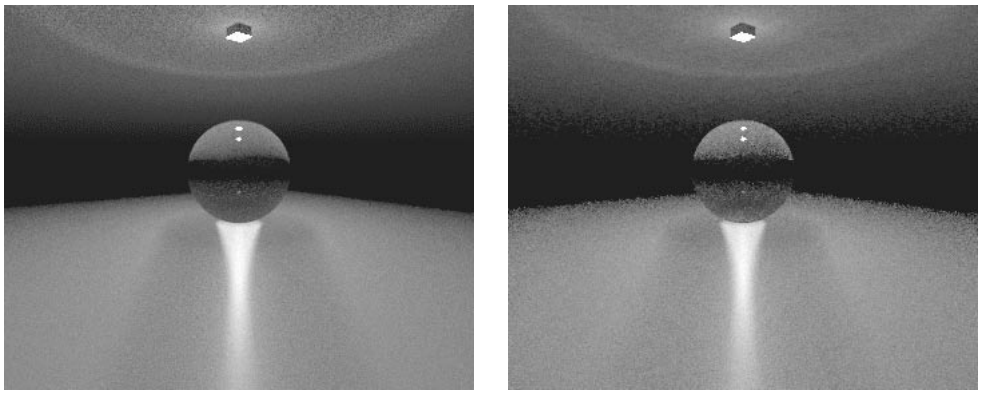


Figure 5.2: A volume caustic rendered with BDPT (left) and MLT (right) in approximately the same computation time.

unevenness in the sample distribution of the MLT algorithm. More samples are concentrated in the bright regions of the image, whereas the darker parts are visited by fewer transport paths. This characteristic feature of MLT is particularly noticable on the ceiling. Immediately around the light source the MLT image is less noisy than its BDPT counterpart. If we move

further towards the darker regions, however, this effect is reversed and the BDPT image is much smoother.
Figure 5.3 shows an inhomogeneous medium, where we use hazy Mie scattering to model the phase function of the cloud. Most of the scene is directly illuminated by a single light source. For this simple scene, BDPT clearly produces the better image.

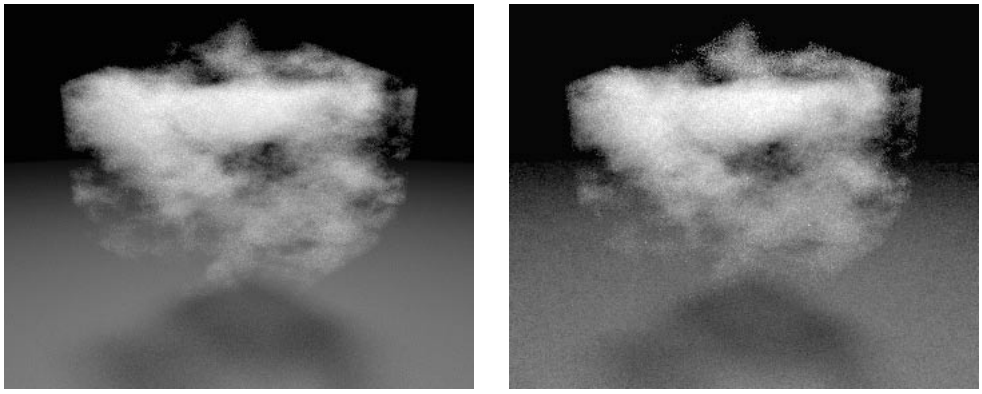


Figure 5.3: A cloud rendered with BDPT (left) and MLT (right) in approximately the same computation time.

Figure 5.4: 100 interconnected rooms rendered with MLT.

Figure 5.5 shows a test scenes with a more difficult lighting situation. The room is entirely illuminated by indirect light passing through the half-open door. The scene contains glossy surfaces, e.g. the floor, transparent objects, e.g. the glass ball, and an inhomogeneous medium "streaming" through the door. Bidirectional path tracing performs poorly in this scene, because very few connections between the light and sensor subpaths will be valid. Thus for most paths the measurement contribution will be zero, which appears as noise in the image. Metropolis light transport is far superior in this setting. The locality of the perturbation strategies leads to a better coverage of the relevant transport paths. MLT will in general perform better if

substantial amounts of the transport paths with a high measurement contribution are clustered in a "small region" of path space. The strong correlation of subsequent samples of the random walk ensures that these regions are sampled adequately. This observation is confirmed in figure 5.4. The scene consists of 100 identical interconnected rooms filled with a homogeneous medium. Each room has its own light source, yet only very few of these contribute noticably to the perceived illumination. MLT automatically detects these important light sources without requiring specific optimization methods for such configurations.
A possible application in building design is given in figure 5.6, where the top image shows a conference room without participating medium. In the lower image the room is filled with homogeneous medium to analyse the visibility of the fire exit signs in case of smoke development. Figure 5.7 illustrates that Metropolis light transport is well suited for scenes with high geometrical and lighting complexity. The scene *Candlestick Theatre* contains 102 light sources, thousands of geometrical primitives a variety of surface materials. The scene *Naval Cruiser* is even more complex and consists of 595 light source and hundreds of thousands of surfaces. An inhomogeneous medium is included in both scenes. While rendering times would be prohibitive for BDPT, MLT produces high-quality images within acceptable time bounds (see table 5.1).

Scene	Image Resolution	Rendering Algorithm	Oversampling/ Mutationrate	Computation Time (min.)
Volume Caustic	352×288	BDPT	100	17.3
		MLT	250	17.5
Cloud	352×288	BDPT	100	20.9
		MLT	200	23.2
Table Room	352×288	MLT	570	18.6
100 Table Rooms	352×288	MLT	570	52.2
Invisible Date	768×576	BDPT	160	692.4
		MLT	700	652.8
Conference Room (no medium)	768×576	MLT	800	510.0
(with homogeneous medium)	768×576	MLT	1000	546.4
Candlestick Theatre	768×576	MLT	1000	1036.8
Naval Cruiser	768×576	MLT	700	601.9

Table 5.1: Computation times for the images of this chapter. All images were rendered on a DEC Alpha Station with 333 MHz. *Oversampling* denotes the number of samples per pixel for BDPT, *Mutationrate* is the number of mutations per pixel for MLT.

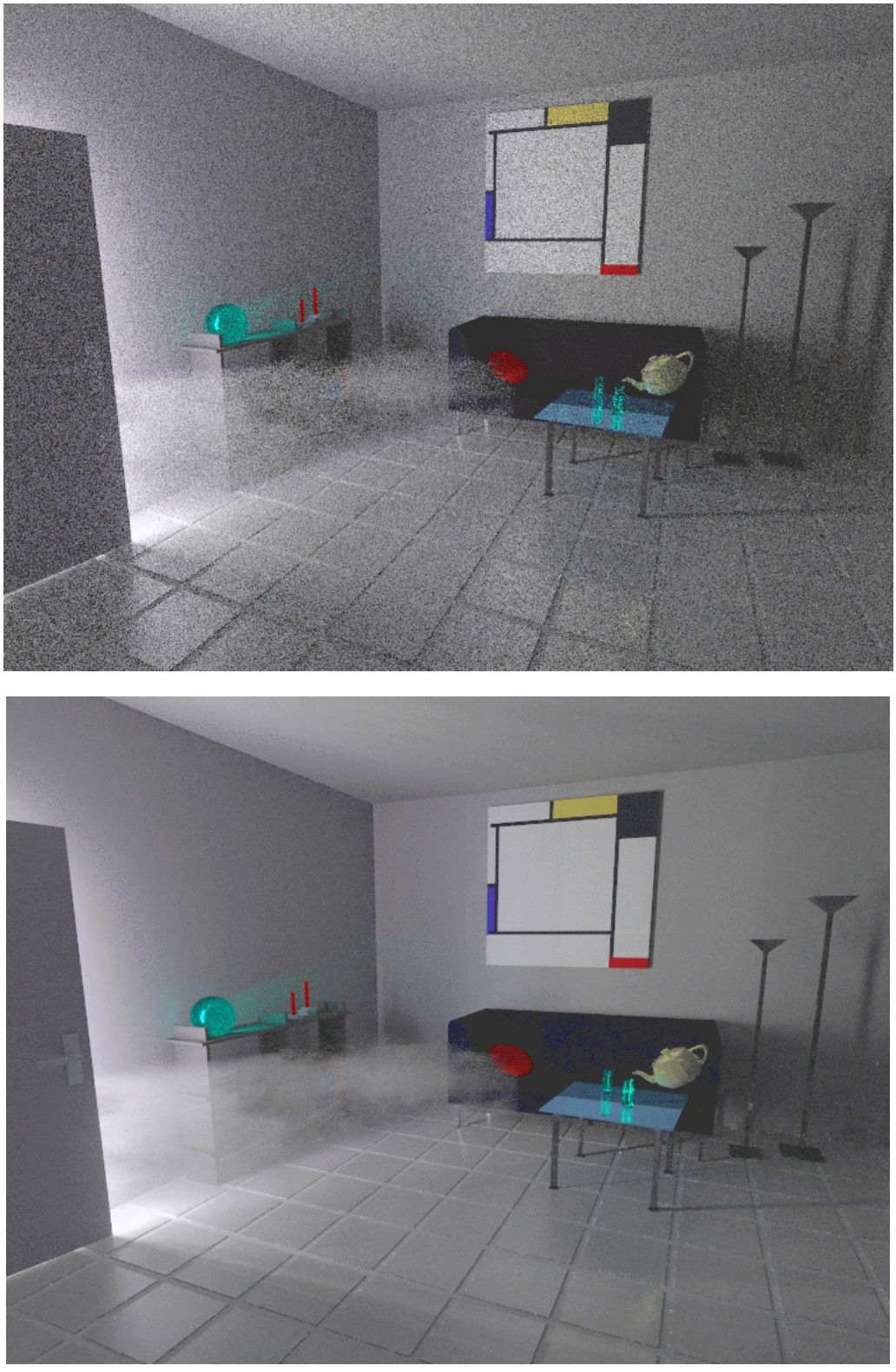


Figure 5.5: The scene *Invisible Date* rendered with BDPT (top) and MLT (bottom) in approximately the same computation time.

Figure 5.6: A conference room with and without participating medium rendered with MLT.

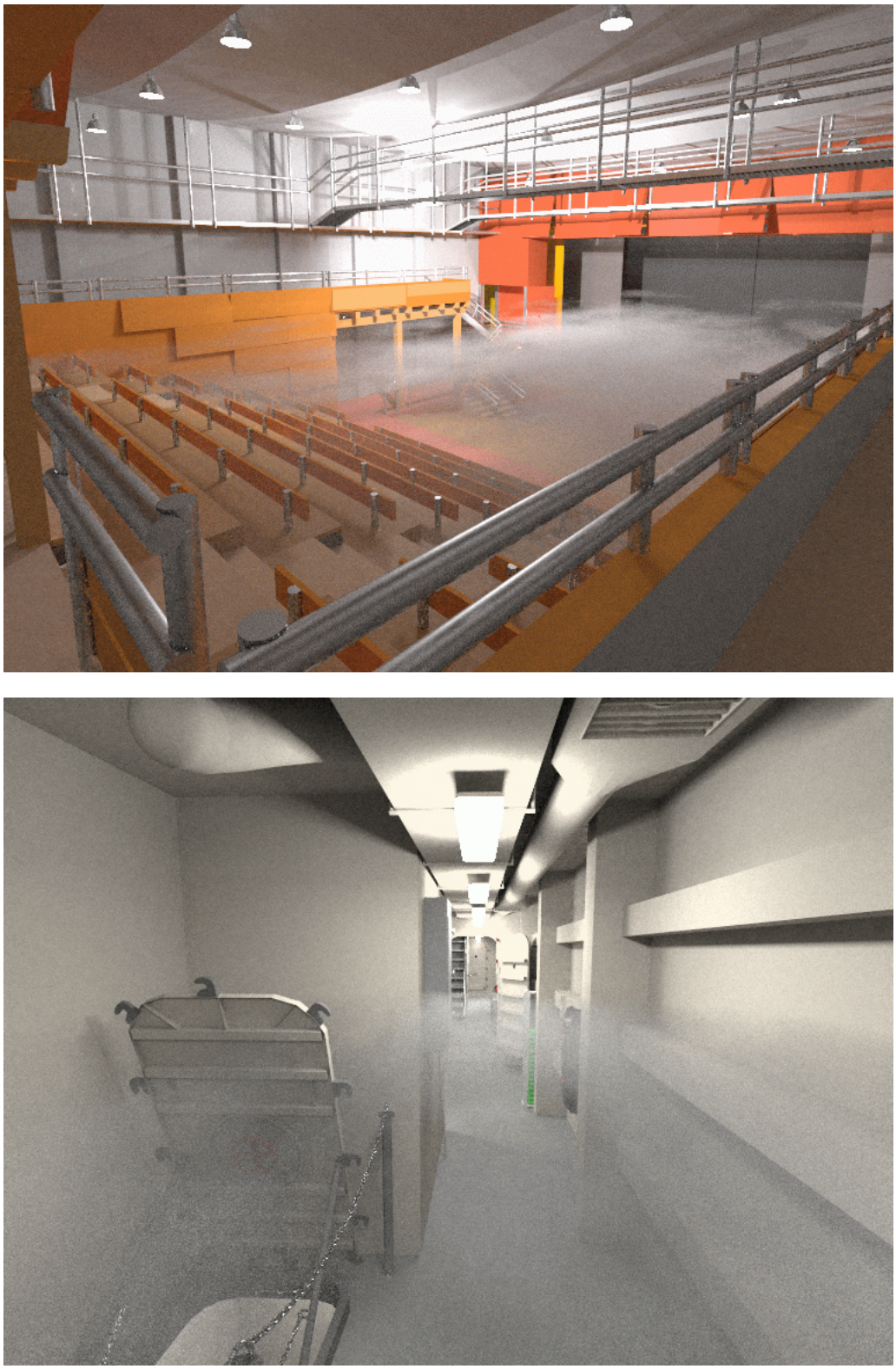


Figure 5.7: Two complex scene models rendered with MLT: Candlestick Theater (top), Naval Cruiser (bottom).

Chapter 6

Conclusions

6.1 Summary

We have given a derivation of the light transport equation for participating media in the context of radiative transfer. Based on this equation we generalized the path integral formulation for light transport to incorporate volumetric scattering interactions. To efficiently sample inhomogeneous media we improved the standard ray marching algorithm by combining equidistant and stratified sampling. We implemented two global illumination algorithms, bidirectional path tracing and Metropolis light transport, and analysed their behaviour for a wide range of inputs. Both methods are physically based, general-purpose, robust Monte Carlo algorithms that are capable of generating high-quality photorealistic images. We found that Metropolis light transport is an essential improvement over bidirectional path tracing, due to its ability to efficiently handle difficult lighting situations and scenes with high geometrical complexity.

6.2 Improvements and Future Research

As described in section 5.1, separating the calculation of paths of length one from the actual Metropolis stage yields a better performance of the algorithm. This approach could be taken further to include direct lighting. To this point, we have not implemented explicit direct lighting calculations, as we want to keep the algorithm robust for scenes with many light sources, a setting generally considered difficult for standard direct lighting methods. Shirley et al. presented a method for dealing with this situation [SWZ96] and it might be worthwhile to integrate this approach into our implementation.
Various extensions to our light source model are possible:

- a generalization of the diffuse emitters to support more sophisticated emittance functions such as directional light sources.
- a realistic simulation of daylight to render outdoor scenes with clouds, fog or smoke with a high degree of realism.
- emitting media such as flames or plasma.

More work needs to be done to optimize the ray marching algorithm. A hierarchical approach may be suitable for large data sets, e.g. using a wavelet representation to store the medium data. The relevant ray section could then be traversed adaptively at different levels of detail.

The propagation perturbation is just one possible mutation strategy that is specifically designed for participating media. Other variations are conceivable, e.g. perturbation of different interaction points or a mutation strategy that takes into account the density gradient of inhomogeneous media.
Further research is necessary to determine the optimal choice for the various parameters of the MLT algorithm. The effects of the empirical pdfs used in the mutation strategies (see section 4.2.4) and the weights for selecting from different strategies need to be studied in more detail.
Parallelizing both algorithms is straightforward, e.g. different processes could compute separate images that are then averaged to obtain the final result.

Appendix A

List of Symbols

$\mathbb{R}$	set of real numbers
$\mathcal{V}$	closed volume in $\mathbb{R}^3$
$\partial\mathcal{V}$	boundary of $\mathcal{V}$
$\mathcal{V}^0$	All points of $\mathcal{V}$ not on the boundary
S^2	unit sphere in $\mathbb{R}^3$
A	area measure
V	volume measure
$A^\perp_\omega$	projected area measure
σ	solid angle
$\sigma^\perp_x$	projected solid angle
Ψ	phase space
ϕ	phase space flux
L	radiance
L_i	incident radiance
L_o	exitant radiance
L^V_e	volume emittance
L^S_e	surface emittance
L_e	emittance
$L_{i,e}$	incident emittance
f_p	phase function
f_s	bidirectional scattering distribution function (BSDF)
$\hat{f}$	scattering function
σ_s	scattering coefficient
σ_a	absorption coefficient
σ_e	extinction coefficient
τ	path transmittance
h	ray casting function
$\mathbf{T_V}$	volume light transport operator
$\mathbf{T_S}$	surface light transport operator
$\mathbf{T}$	light transport operator
I_j	measurement of sensor j
$W_e^{(j)}$	sensor responsivity
Ω^l_k	set of paths of length k
Ω	set of all finite length paths
μ^l_k	measure on Ω^l_k
μ	path space measure
V_g	visibility function
G	geometric term

Bibliography

[Arn97] Arnold, T.
Photorealistische Bildsynthese mit Volumenvisualisierung
Diplomarbeit, University of Kaiserslautern, 1997

[Arv93] Arvo, J.
Transfer Equations in Global Illumination
Global Illumination, SIGGRAPH '93 Course Notes, volume 42, 1993

[BLS93] Blasi, P., Le Saëc, B. & Schlick, C.
A Rendering Algorithm for Discrete volume Density Objects
Computer Graphics Forum (Proc. of Eurographics '93), pp. 201-210, 1993

[Cha60] Chandrasekar, S.
Radiative Transfer
Dover Publications, New York, 1960

[Kaj86] Kajiya,J.
The Rendering Equation
Computer Graphics (SIGGRAPH 86 Proceedings), pp. 143-150, 1986

[Kel98] Keller, A.
Quasi-Monte Carlo Methods for Photorealistic Image Synthesis
PhD thesis, Shaker Verlag, Aachen, 1998

[Kol99] Kollig, T.
Lichtsimulation mit dem Metropolis Algorithmus
Diplomarbeit, University of Kaiserslautern, 1999

[KW86] Kalos, M. & Whitlock, P. Monte Carlo Methods, Volume I:Basics
John Wiley & Sons, New York, 1986

[LW93] Lafortune, E. & Willems, Y.
Bidirectional Path Tracing
CompuGraphics Proceedings, pp. 145-153, 1993

[LW96] Lafortune, E. & Willems, Y.
Rendering Participating Media with Bidirectional Path Tracing
Proc. 7th Eurographics Workshop on Rendering, pp. 91-100, 1996

[MRR+53] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A. & Teller, E.
Equation of State Calculations by Fast Computation Machines
Journal of Chemical Physics 21, pp. 1087-1092, 1953

[OZ99] Owen, A.B. & Zhou, Y.
Safe and effective importance sampling
Technical Report, Department of Computer Science, Stanford University, 1999

[PH89] Perlin, K. & Hoffert, E.
Hypertexture
Computer Graphics (SIGGRAPH Journal, vol. 23), pp. 253-262, 1989

[Rus94] Rushmeier, H.E.
Rendering Participating media: Problems and solutions from application areas
Proc. 5th Eurographics Workshop on Rendering, pp. 35-56, 1994

[SWH+95] Shirley, P., Wade, B., Hubbard, P. M., Zareski, D., Walter, B. & Greenberg, D. P.
Global illumination via density-estimation
Proc. 6th Eurographics Workshop on Rendering, pp. 219-230, 1995

[SWZ96] Shirley, P., Wang, C. & Zimmerman, K.
Monte Carlo Techniques for Direct Lighting Calculations
ACM Trans. Graphics, vol. 15, pp. 1-36, 1996

[Sob91] Sobol, I.
Die Monte-Carlo-Methode
Deutscher Verlag der Wissenschaften, Berlin, 1991

[Vea97] Veach, E.
Robust Monte Carlo Methods for Light Transport Simulation
PhD thesis, Stanford University, 1997

[VG94] Veach, E. & Guibas, L.
Bidirectional Estimators for Light Transport
Proc. 5th Eurographics Workshop on Rendering, pp. 147-162, 1994

[VG97] Veach, E. & Guibas, L.
Metropolis Light Transport
Computer Graphics (SIGGRAPH 97 Proceedings), pp. 65-76, 1997

[War92] Ward, G.
Measuring and Modeling Anisotropic Reflection
Computer Graphics (SIGGRAPH 92 Proceedings), pp. 265-272, 1992