Mathematical Models and Monte Carlo Algorithms for Physically Based Rendering

Eric Lafortune

Department of Computer Science, Faculty of Engineering
Katholieke Universiteit Leuven

ABSTRACT

Algorithms for image synthesis render photo-realistic imagery, given the description of a scene. Physically based rendering specifically stresses the physical correctness of the algorithms and their results. The algorithms perform an accurate simulation of the behaviour of light, in order to faithfully render global illumination effects such as soft shadows, glossy reflections and indirect illumination. In this dissertation we investigate image-based Monte Carlo rendering algorithms. We pay special attention to their correctness, their versatility and their efficiency.

First of all, we discuss theoretical frameworks that describe the global illumination problem. These formal mathematical models are the first step to ensure the correctness of the eventual results. Moreover, they allow to apply standard numerical techniques to compute a solution. We give an overview of existing models, which are based on the rendering equation and the potential equation. We then introduce a model based on a new concept, called the global reflectance distribution function. It combines the ideas of radiance and potential into a single function, which is defined by a set of two integral equations. We later show that, while the three models are equivalent, a straightforward Monte Carlo approach to solve them leads to entirely different rendering algorithms.

We chose to apply Monte Carlo methods because of their versatility. First, we give an overview of Monte Carlo techniques in general. The variance of a technique provides a measure for the stochastic errors on its results. The basic strategy of Monte Carlo methods is to reduce the variance by averaging the results of large numbers of samples. Convergence is slow, however. Variance reduction techniques therefore try to improve the efficacy of the individual samples, by taking into account information about the integrand. We discuss techniques such as stratified sampling, importance sampling, the combining of estimators, control variates, Russian roulette and next event estimation. We stress their unbiasedness, which ensures that the estimators always converge to the exact solution. We specifically analyse aspects that are of importance for the global illumination problem and we present a few improvements to existing techniques.

Finally, we apply the Monte Carlo methods to the mathematical models of the global illumination problem. The different models give rise to entirely different algorithms. The rendering equation leads to the well-known path tracing algorithm, while the potential equation leads to the light tracing algorithm. We discuss their strengths and weaknesses. We study the application of the variance reduction techniques and present results from practical experiments. The global reflectance distribution function and its equations give rise to a new algorithm, which we have called bidirectional path tracing. This algorithm proves to have superior qualities for rendering typical indirectly illuminated interior scenes.

Acknowledgements

Many people have helped making this work possible. First of all, I am much indebted to my supervisor Prof. Yves Willems for his support and encouragements over the last few years. Although slightly disorganised, meetings and discussions with him have always been enriching. His critical comments, questions, suggestions and his enthusiasm provided an incessant stimulus to improve. He created the unique working atmosphere that I value a lot.

I would also like to thank the other members of the reading committee, Prof. Cools and Prof. Bultheel, who have studied the text meticulously. Their comments have been invaluable during the writing of this dissertation. I am grateful to the other members of the jury, Dr. Pattanaik, Prof. Bouatouch, Prof. Jansen and Prof. Dutré, for sparing the time to study this thesis. Sumant, Kadi and Erik have been there from the early start. They have really taught me everything, by their excellent example, their constructive criticism and their kind encouragements.

Philip Dutré and Philippe Bekaert have undoubtedly contributed most of all to this work. The discussions about Monte Carlo rendering and those few remaining aspects of the universe have kept us sharp. I still maintain that Philip is more stubborn than I am, although he may disagree. Together with my other colleagues at the department, especially Henk, Veroniek, Wim, Gerda and Remco, they made this group a great place to work.

Many regards go to my research colleagues and friends, Pete, Alan, Henrik and many others, who share the same passion and travel around the world to drink on it. They evidence the pleasure and fun of computer graphics research. After all, these integral equations are just an excuse to play and render colourful images.

My windsurfing, skiing and computing companions deserve a special mention. They showed understanding when being called out of bed because of a cold storm front passing through. They taught me a lot, picked me up after crashing and were the best company one could ever wish.

Finally, many thanks and love go to my parents, to Muriel and Jacques, and to other dear friends, for being there, and much more.

I acknowledge the financial support by a grant from the Flemish "Institute for the Promotion of Scientific and Technological Research in the Industry" (IWT #944045).

Contents

Chapter 1

Introduction

1.1 Rendering

Part of the research on computer graphics is concerned with image synthesis, i.e. the rendering of virtual scenes by means of a computer. The input of a rendering program is a description of a scene by way of its geometry and the optical characteristics of the objects. After specification of viewing parameters such as the viewpoint and the viewing direction the eventual output of the program is a synthetic image of that scene. Image synthesis has numerous practical applications, for instance:

- In mechanical engineering computer renderings aid the development of complex work pieces. Engineers can catch and correct errors at an early stage by interactively examining and changing the model.
- In industrial design the renderings may give an impression of the aesthetic and ergonomic qualities of the design. They can illustrate the effects of different shapes, textures and colours without physically having to construct different prototypes.
- The entertainment industry makes increasing use of computer graphics and of synthetic imagery for short or even full-length feature films, commercials and motion rides.
- Renderings that correctly simulate illumination effects can help civil engineers, illumination engineers and architects designing safe and ergonomic tunnels, buildings, houses, etc.

The requirements placed upon image synthesis algorithms strongly depend on the application. On the one hand the rendering process should be as fast as possible. For animation films this is typically of the order of a few minutes to a few hours per frame, while interactive design applications require at least several frames per second. On the other hand an increasing number of applications demand a high degree of realism. For mechanical engineering the images have to convey the geometry of the model unambiguously, in which case a simple visualisation suffices. Computer animators strive for visually more pleasing imagery. The images should not only present the geometry of the scene but also include illumination effects such as shadows and reflections. Illumination engineering applications additionally need the resulting images to be a physically truthful representation of reality. This requires an accurate simulation of the behaviour of light in the virtual scene, which is obviously computationally more expensive. Because these goals of speed on the one hand and realism on the other hand are contrary, all rendering algorithms have to make a trade-off. Current technology does not allow real-time photo-realistic rendering, which makes the trade-off all the more tangible. Research on image synthesis algorithms seeks to push the boundaries of both speed and realism.

The outset of this thesis is the aspect of realism and accuracy of synthetic imagery. We will stress the use of formal physical models and mathematically sound techniques in developing rendering algorithms. This well-founded theoretical basis is the initial step in a scientific approach to the problem. In the first place it offers an indication about the correctness of the eventual algorithms. The accuracy of the results obviously also depends on the accuracy of the input, on any restrictions of the physical model and on the numerical accuracy of the algorithm. We will discuss each of these aspects in the text.

In the second place the mathematical framework accommodates general numerical optimisations that produce equally accurate results with less computational effort. The basic algorithms that we will derive in a straightforward way from the mathematical models are physically correct but excruciatingly slow. Further improvements should therefore make them practical. Rather than resorting to very specific shortcuts we will look at common numerical techniques and examine how they can be applied to the problem at hand.

In the following sections we will further introduce the rendering problem and this work. First we will briefly discuss the typical input and the output of a rendering algorithm. Then we will present a cursory inspection of different types of physically-based rendering algorithms. We will conclude the introduction with a survey of the objectives of this thesis and an overview of the contents of the text.

1.2 The Input

As mentioned in the introduction the input of a rendering program consists of a description of the geometry and the optical characteristics of the model to be rendered. Additionally the viewing parameters have to be specified.

The geometry of a scene is in practice described by a combination of geometric primitives. They present either a solid representation or a boundary representation:

- A *solid representation* is based on three-dimensional primitives like spheres, cubes, cones, etc. A list of their coordinates and size parameters defines the entire geometry. Transformations, such as scaling and shearing, and constructive solid geometry operations, such as taking the union or the difference of primitives, further enhance the versatility of the representation. It is natural to work with in the sense that it closely corresponds to our everyday experience that complex objects often consist of simpler primitives. It is most commonly used by ray tracing algorithms, which can handle arbitrary geometries.

- A *boundary representation* on the other hand only describes the two-dimensional surfaces that delimit the solid objects. The surfaces are defined mathematically by the coefficients of implicit or parametric equations. Boundary element methods work on surfaces that are discretised, mostly into triangles or general polygons. Rendering algorithms in this case perform their illumination computations on the geometric elements. The radiosity algorithm is a typical example. The finite elements provide a uniform representation which is convenient to process.

The optical characteristics of the objects are likewise described by the parameters of appropriate models. The complexity of these models strongly depends on the application. In the simplest case each surface has Red/Green/Blue-components describing a colour assigned to it. Simple shading techniques such as the well-known Phong reflection model take into account light sources in the scene. Physically-based rendering requires more exact information about the optical characteristics of each surface:

- The *reflective properties* of a surface are quantified by the bidirectional reflectance distribution function, which will be discussed in Section 2.1.3. From a macroscopic point of view light that is incident to a surface is partially absorbed and partially reflected. Reflected light may be scattered in all directions. The amounts of reflection and absorption depend on parameters such as incident direction, exitant direction and the wavelengths of the incident and exitant light. A reflectance model has to represent these intricate phenomena. The Lambertian model for instance only approximates the surfaces as perfectly diffuse. They are then characterised by their diffuse reflectivities. More sophisticated models also take into account directional reflection such as specular and glossy reflection.

- The *self-emissive properties* of light sources are quantified by the self-emitted radiance function, which is further discussed in Section 2.1.1. Light is emitted as a result of an electrical or a chemical process. Radiation from surfaces such as the glowing wire inside a light bulb is mostly diffuse. Compound emitters such as the entire light bulb generally have more complex characteristics, which again have to be described by a suitable model.

The geometric models along with their surface characteristics are usually created by means of a modelling program or measured directly using 3D scanners and goniophotometers. Manufacturers of luminaires sometimes provide the required data to model light sources. We will assume that a complete model of the scene is available to begin with. This text therefore does not consider the problem of measuring and modelling the geometric sizes and optical characteristics of surfaces. Without limiting the scope of the algorithms presented, our implementation and test scenes are based on simple geometric and optical models.

In principle light can also interact with participating media like fog, smoke or dust. Measuring and modelling their geometries and optical characteristics and then taking into account their presence is a topic of ongoing research. Fortunately the influence of a medium like air is negligible in most common scenes. We will therefore concentrate on the interaction of light with surfaces only.

Even without participating media the resulting images can obviously only be as accurate as the input model. So far little work has been done on the relation between modelling and rendering. In practice the overall solution is not very sensitive to small errors in the input geometry or optical characteristics. For instance, a slight change in the position of an object in a scene will generally have little influence on the overall illumination and aspect of the scene. The images do tend to look more realistic when the input model is detailed, for instance by extensive use of texture maps to model optical details on the surfaces. In order to obtain visually pleasing images the modelling step may even be more important than the rendering step. Attention to the latter is still essential however for applications that require physically accurate images.

The final information required to render an image is a set of viewing parameters. This includes the viewpoint of the observer in the virtual scene, the viewing direction and the field of view. More sophisticated rendering techniques may require parameters such as the camera aperture and the plane of focus to realistically simulate camera artifacts.

1.3 The Output

The eventual output of a rendering program is an image of the input scene. As mentioned earlier the desired quality varies with the application. For simple design and engineering applications a wire frame representation may do. Colouring the surfaces, applying a simple shading model such as Phong shading and interpolating colours by means of Gouraud shading may further improve the appearance of the images. The chief advantage of these rendering techniques is that they are supported by modern graphics hardware.

Physically-based rendering has more ambitious goals. Cohen [14] notes that these goals are not clearly defined though. Ideally the aim would be to create the same visual impression as when watching a corresponding actual scene. Due to practical and technological constraints such as the available display hardware this is currently not feasible. An alternative goal is to render an image that is indistinguishable from a photograph. Appendix A discusses some technical implications of such a choice. This dissertation will however not go into the technological problems resulting from the limited dynamic ranges and colour gamuts of output devices for instance. We will study the so-called global illumination problem. Typical examples of global illumination effects are soft shadows, glossy reflections, indirect illumination and “colour bleeding” of coloured surfaces onto neighbouring surfaces due to diffuse reflection. Essential for physically-based rendering is the physical correctness of the results, irrespective of the way in which they are eventually presented to the viewer. We will concentrate on the lighting simulation which ensures this correctness.

1.4 Physically-Based Rendering Algorithms

The core problem of physically-based rendering is the global illumination problem. The visual perception of a scene is the result of light being emitted from light sources, interacting with the objects in the scene and arriving at the viewpoint. The interaction is in general a complex combination of reflection, transmission and scattering of light. Rendering a realistically looking and accurate image therefore requires a faithful simulation of these global illumination effects in the scene. The results of the simulation are radiance functions or radiant fluxes leaving the surfaces of the scene, as will be discussed in Chapter 2.

The introduction of the radiosity algorithm by Goral *et al.* [34] in '84 was the first incentive for a physically more exact approach to the rendering problem. Previous rendering techniques relied on *ad hoc* approximations of various visual effects. The radiosity algorithm solves a finite element model of diffuse illumination in a scene. As such it was the first to be based on an actual physical model. It expresses the model in mathematical terms and subsequently finds a solution by means of known numerical algorithms.

Another milestone was the rendering equation, which was introduced in computer graphics by Kajiya [53] in '86. It provided a more general physical framework for the rendering problem. Along with it Kajiya presented a stochastic ray tracing algorithm for solving it.

Both the radiosity algorithm and stochastic ray tracing have been improved and extended since. Excellent overviews can be found in the reference works by Cohen and Wallace [17], Sillion and Puech [118] and Glassner [33]. Without enumerating all contributions it may be noteworthy to mention two common classifications of physically-based rendering algorithms: object-based versus image-based algorithms and deterministic versus Monte Carlo algorithms.

1.4.1 Object-Based and Image-Based Algorithms

This classification tells something about *what* the algorithms try to solve. It is based on the space for which the algorithms find a solution.

- *Object-based rendering algorithms* try to solve the global illumination problem independently of the eventual viewing parameters such as viewpoint and field of view. For this purpose the problem has to be simplified first, as a completely general solution requires too much memory and too much computation time in practice. Typical examples are the radiosity method and similar finite element methods which compute a radiosity solution for all surfaces in the scene. Rendering the solution becomes a relatively simple post-processing step. On the basis of the viewing parameters the surfaces can be painted with any hidden-surface algorithm while the illumination solution provides the colours. In the case of the radiosity method graphics hardware can assist with the rendering, allowing interactive walk-throughs of moderately complex scenes. Important developments for object-based rendering algorithms have been hierarchical algorithms, higher order basis functions and wavelet algorithms, e.g. [122, 41, 63, 35, 104].

- *Image-based rendering algorithms* consider the viewing parameters as a basic component of the problem. They compute solutions specifically for all pixels or groups of pixels in the image. Although the underlying global illumination models are the same the resulting algorithms are usually quite different. Typical examples are stochastic ray tracing algorithms, e.g. [20, 19, 53], which compute radiance values for all individual pixels in turn. Image-based algorithms accordingly only pursue partial solutions for the global illumination problem, i.e. for the visible part of the scene rather than for the entire scene. In general they can handle more complex illumination models and geometries as a result. Their disadvantage is speed; with current technology it is still not possible to generate acceptable images fast enough for interactive frame rates.

This classification is of course not absolute. Several two-pass or multi-pass algorithms have been developed that combine an object-based view-independent pass with an image-based view-dependent pass, e.g. [117, 127, 11, 137]. More recent object-based rendering algorithms also take into account the eventual viewing position, for instance to guide hierarchical refinement of the finite elements [119, 7, 9] or to determine the ordering of the computations [10]. Image-based rendering algorithms on the other hand often store information about the illumination in object-space to speed up the computations [134, 133, 67, 50].

1.4.2 Deterministic and Monte Carlo Algorithms

Like all numerical algorithms rendering algorithms can also be subdivided in deterministic and Monte Carlo algorithms, which says more about *how* they solve the problem:

- *Deterministic rendering algorithms* are based on classical numerical techniques. In the case of the global illumination problem these techniques mostly involve specific cubature rules for integrating high-dimensional functions, e.g. [16, 8, 103, 105], and solution methods for large sets of linear equations, e.g. [15, 41, 35]. Theoretical and practical error bounds give an indication of their effectiveness.

- *Monte Carlo rendering algorithms* employ stochastic techniques to compute the high-dimensional integrals of the global illumination problem. Monte Carlo algorithms in general average their results from a large number of random samples. The samples are generated according to the underlying stochastic process of the problem. The more samples the algorithm can take the more reliable the results will be. The amount of stochastic noise in the results is a measure of their success, which is expressed by the variance of the algorithm. Chapter 3 further discusses the basic principles of Monte Carlo algorithms and several optimisations. In the context of rendering, Monte Carlo techniques are applied both in object-based algorithms, e.g. [86, 83, 81], and in image-based algorithms, e.g. [20, 19, 53]. Because of their simplicity they are somewhat more versatile, which makes them often the only practical alternative for image-based rendering algorithms.

Again this classification is not clear-cut. Deterministic rendering algorithms may apply Monte Carlo techniques to solve sub-problems. A deterministic radiosity method may compute form factors using a Monte Carlo method for instance [128, 41].

1.5 Objectives of this Thesis

The primary goal of this thesis is the development and improvement of physically-based rendering algorithms. We will concentrate on the global illumination problem which is the chief problem to be solved in this context. The algorithms should meet the following conditions:

- **Correctness.** The lighting simulation must by faithful to reality, so that the results can be relied upon. The basis of the correctness should be a formal physical model of the global illumination problem. Even if it has to be simplified to be practical this allows to identify any limitations of the algorithms that are derived from it. We will present several alternative models. On the basis of these we will be able to apply provably correct mathematical techniques and standard numerical algorithms to solve the problem. We will discuss these techniques and their application extensively.

- **Versatility.** We are interested in algorithms that are general, in addition to being correct. The input should not be restricted in any way, other than being physically possible. The algorithms should handle arbitrary types of geometry and optical characteristics of the surfaces. They should scale well to scenes of arbitrary complexity and detail.

- **Efficiency.** The basic algorithms are versatile but computationally expensive as a result. We will therefore study their efficiency and any optimisations that can reduce the computation time. Thanks to the formal mathematical framework we will be able to apply standard numerical optimisations. A theoretical study should indicate their potential while test results will then reveal their practical use.

1.6 Overview

The remaining chapters of this dissertation are organised as follows:

- **Chapter 2** introduces the physical concepts that are relevant to physically-based rendering and to the global illumination problem. The essential radiometric quantities are radiance and radiant flux. We will formalise the problem itself using alternative mathematical models. The classical model to describe the radiance function is the rendering equation. The potential function which has been introduced into computer graphics more recently is defined by the potential equation. This equation presents a dual view of the problem. We will then present a new concept, the global reflection distribution function, which combines the ideas of the radiance function and the potential function in a single function. We will show how it is defined by two equivalent integral equations.

- **Chapter 3** gives an overview of Monte Carlo methods in general. After an explanation of the basic principles the emphasis will lie on variance reduction techniques. We will present them in a coherent framework and stress the underlying ideas they have in common. We will note any elements that are of particular interest for application to the global illumination problem.

- **Chapter 4** applies the numerical techniques of Chapter 3 to the mathematical framework of Chapter 2. We will argue why we opt for an image-based approach and for Monte Carlo algorithms, given our objective of versatility. In this context we will show how the different models lead to different rendering algorithms. Most commonly known is the rendering equation as a basis for the path tracing algorithm. This algorithm gathers light starting from the viewpoint. Similarly the potential equation leads to the light tracing algorithm, which distributes light starting from the light sources. The specific strengths and weaknesses of these algorithms will be discussed. We will show how the global reflectance distribution function and its equations give rise to a new algorithm called bidirectional path tracing. This algorithm successfully combines the strengths of the previous approaches by simultaneously distributing and gathering light from the light sources and from the viewpoint respectively.

  The variance reduction techniques discussed in the previous chapter can improve the basic rendering algorithms further. We will present a systematic analysis of these general optimisations in the case of these specific algorithms.

- **Chapter 5** presents some practical test results. We will apply the various algorithms and optimisations of Chapter 4 to a set of test scenes. The results should give an impression of their qualities in practice.

- **Chapter 6** summarises the results of this work. We will draw some final conclusions and indicate possible future research directions.

Chapter 2

The Global Illumination Problem

Physically-based rendering is a simulation problem that is typically solved in three steps:

1. Identifying the physical problem,
2. Formalising the problem in a mathematical model,
3. Developing an algorithm to solve the model.

In this chapter we will concentrate on the first two steps. In physically-based rendering one is interested in light and its interaction with the environment. Light is emitted from light sources and reflected through the scene. Light that is emitted or reflected in the direction of the viewer contributes to the perceived image of the scene. Physically-based rendering is concerned with creating synthetic images of a given model by simulating these processes. The images should approximate real images by some measure. The major problem in physically-based rendering is the global illumination problem: determining the complex interreflection of light through the scene.

Throughout this dissertation we will consider the problem from the point of view of geometric optics. This corresponds to a particle model of light, in which photons travel in straight lines, with their energies adding up to macroscopically measurable quantities. We will not consider the wave model of light that can account for effects such as interference and polarisation.

We will first present the basic notions that are prerequisite to talk about global illumination. The key notions to quantify light are radiance and radiant flux. These are the radiometric quantities that we are actually trying to compute. Reflection at surfaces can be described by means of the bidirectional reflectance distribution function which quantifies the local reflective properties of the surfaces in a scene.

Using these notions we will then discuss a few alternative mathematical models of the global illumination problem. The most well-known formulation is the rendering equation. This integral equation determines the radiance function. Closely related to it is the potential equation. This dual version of the rendering equation determines the potential function. We then introduce a new concept, the global reflectance distribution function, along with two equivalent integral equations that determine its behaviour. The different models provide different views on the global illumination problem. In later chapters we will show how they can give rise to different algorithms.

2.1 Concepts

The basic notions for the global illumination problem are the radiometric quantities. They are usually considered over the visual spectrum that lies between 400 and 700 nm. Each radiometric quantity also has a spectral counterpart that expresses the quantity per unit wavelength. These quantities can be indicated by a subscript λ. In our discussions we will usually make abstraction of this wavelength dependency and omit the subscript.

2.1.1 Radiance

The *radiance* L for a given point x and direction Θ_x is defined as the differential power radiated per unit surface area and per unit time from point x in direction Θ_x (Fig. 2.1):

$$L(x, \Theta_x) = \frac{\partial^2 \Phi}{\partial A_{x\perp} \partial \omega_x}, \tag{2.1}$$

where:

- $\partial A_{x\perp}$ = a differential surface area around point x and at a right angle with direction Θ_x,
- $\partial \omega_x$ = a differential solid angle around Θ_x,
- $\partial^2 \Phi$ = the differential radiant power through $\partial A_{x\perp}$ and $\partial \omega_x$.

Radiance is expressed in $W/m^2 sr$. Note that steradians (sr) are not really physical units but they are commonly included for clarity. As a radiance value is defined for each point in the 3D space and for each direction in the 2D directional space the radiance function is 5-dimensional. It is fundamental as most radiometric and photometric quantities can be derived from it. It is also important because it is the quantity that is perceived eventually; a view is defined unambiguously by the radiance values that reach the viewer.

The radiance immediately before it hits a surface is of special interest and is called the *incoming radiance* or *field radiance*. We will denote it by L_i wherever required. The radiance immediately after it leaves a surface is called *outgoing radiance* or *surface radiance*. We will denote it by L_o. They can be defined formally with respect to the surface:

$$L_i(x, \Theta_x) = \frac{\partial^2 \Phi}{|\Theta_x \cdot N_x| \, \partial \mu_x \partial \omega_x}, \tag{2.2}$$

$$L_o(x, \Theta_x) = \frac{\partial^2 \Phi}{|\Theta_x \cdot N_x| \, \partial \mu_x \partial \omega_x}, \tag{2.3}$$

where:

- $|\Theta_x \cdot N_x|$ = the cosine (in absolute value) of the angle between direction Θ_x and the normal N_x to the surface at point x,
- $d\mu_x$ = a differential surface area on the surface around point x,
- $\partial \omega_x$ = a differential solid angle around Θ_x,
- $\partial^2 \Phi$ = the differential radiant power coming in, respectively going out, through $\partial \mu_x$ and $\partial \omega_x$.

Furthermore the *self-emitted radiance* L_e can be defined, which is radiance that is emitted from light sources as the result of some electrical or chemical process.

Invariance along a direction

The most important property of radiance for our applications is that it is invariant along the direction in which it is radiated, if the light travels through vacuum (Fig. 2.2). This property can easily be proven using the more general property of conservation of energy. As in most global illumination models and algorithms we will approximate the actual media –air usually– by vacuum. This approximation is valid for most daily scenes that do not contain strongly scattering, absorbing or emitting media such as smoke, steam, fog or fire.

The invariance of radiance along a line implies that the surface radiance at one point on a surface is equal to the field radiance at the point on the first surface along the given direction. That point can be defined formally by means of a *ray casting function* r that maps a point x and direction Θ_x on the nearest point on a surface along that direction (Fig. 2.3). The invariance relation then becomes

$$L_o(x, \Theta_x) = L_i(r(x, \Theta_x), \Theta_x). \tag{2.4}$$

In our derivations we will often implicitly use the following reciprocity relation of the ray casting function:

$$y = r(x, \Theta_x) \Leftrightarrow x = r(y, \Theta_x^{-1}),$$

where we denote the opposite direction of a direction Θ by Θ^{-1}. By convention the directions in most of the equations we will present point in the direction of the light transport we are interested in, to make it easier to visualise their physical meanings. Alternatively one could only consider directions pointing away from the surfaces, as in [26] for example.

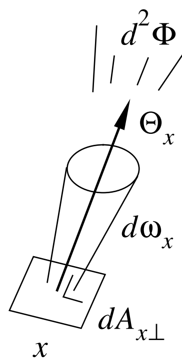


Figure 2.1: Explanation of the symbols used in the definition of radiance. Radiance expresses the amount of light radiated from a given point x in a given direction Θ_x.

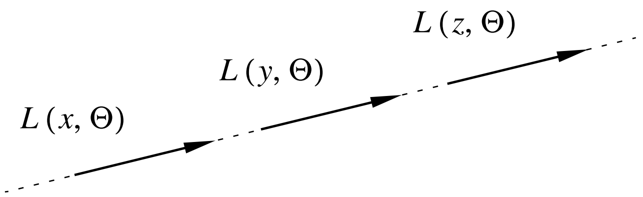


Figure 2.2: In absence of a participating medium radiance is invariant along the direction in which it is radiated.

2.1.2 Radiant Flux

The *radiant flux* Φ through a given set of points and directions S is defined as the total amount of radiant energy per unit time that is being radiated from that set S. It is expressed in W. By definition it is the integral of the radiance function $L(x,\Theta_x)$ over all points and directions of S:

$$\Phi = \int\int_S L(x,\Theta_x)\,|\Theta_x \cdot N_x|\, d\omega_x d\mu_x. \tag{2.5}$$

For surface radiance the expression can be rewritten as an integral over all surface points and all directions with the help of a suitable function $W_e(x,\Theta_x)$:

$$\Phi = \int_A \int_{\Omega_x} L(x,\Theta_x) W_e(x,\Theta_x)\,|\Theta_x \cdot N_x|\, d\omega_x d\mu_x, \tag{2.6}$$

where:

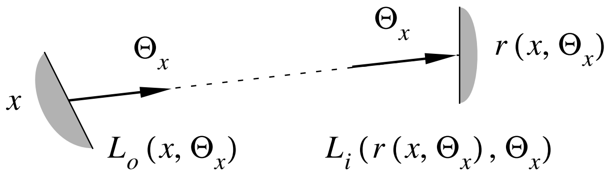


Figure 2.3: The ray casting function r maps a point x and a direction Θ_x on the nearest point as seen from point x along direction Θ_x. Due to the invariance of radiance along direction Θ_x the surface radiance L_o at (x,Θ_x) equals the field radiance L_i at $(r(x,\Theta_x),\Theta_x)$.

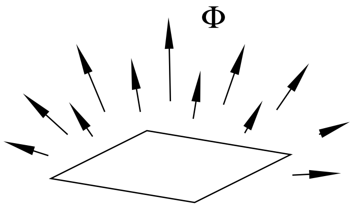


Figure 2.4: Radiant flux is the amount of radiant energy per unit time being radiated from a set of points and directions. In radiosity methods the set typically consists of all the points of a surface patch along with all the corresponding hemispherical directions.

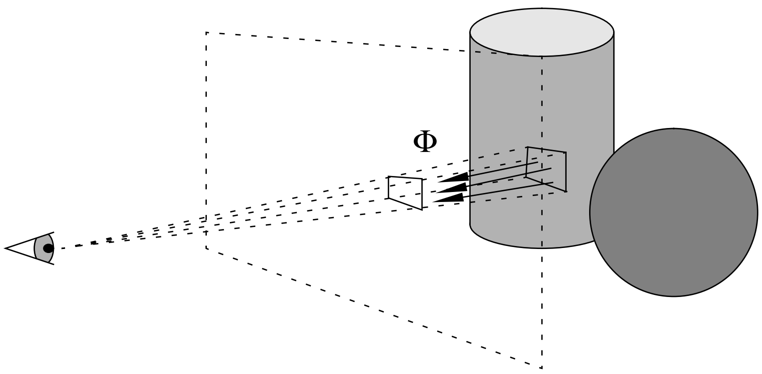


Figure 2.5: A set that is of interest for image-based rendering techniques is the set of surface points and directions that point through a pixel on a virtual screen.

- A = the combined surfaces in the scene,
- Ω_x = the hemisphere of outgoing directions above the surface at point x,
- $W_e(x, \Theta_x)$ = 1 for all pairs of points and directions that belong to the set, and 0 otherwise (the function is called W_e for reasons which will become clear later on),
- $d\omega_x$ = a differential solid angle around direction Θ_x,
- $d\mu_x$ = a differential surface area on the surface around point x.

The expression for the flux may be written in short using a dot product notation:

$$\Phi = \langle L, W_e \rangle, \tag{2.7}$$

where the dot product is defined as

$$\langle f, g \rangle = \int_A \int_{\Omega_x} f(x, \Theta_x) g(x, \Theta_x) \left| \Theta_x \cdot N_x \right| d\omega_x d\mu_x.$$

A set in which one may be interested is the set of points on a surface patch along with all the hemispherical directions as in Fig. 2.4. Classical radiosity methods compute these fluxes in one form or another for each of the patches of the discretised scene. Of greater interest to us here will be the set of points and directions corresponding to a pixel in a virtual camera as in Fig. 2.5. Image-based methods try to compute the corresponding fluxes for each of the pixels.

Note that one can alternatively define a weighted average $\overline{L}$ of the radiance function over some part of the domain, rather than the total flux, by defining $W_e(x, \Theta_x)$ as a weight function that may have other values than 0 or 1. As a weight function it is normalised as follows:

$$\int_A \int_{\Omega_x} W_e(x, \Theta_x) \left| \Theta_x \cdot N_x \right| d\omega_x d\mu_x = 1.$$

The expression remains the same as the expression for the flux:

$$\overline{L} = \int_A \int_{\Omega_x} L(x, \Theta_x) W_e(x, \Theta_x) \left| \Theta_x \cdot N_x \right| d\omega_x d\mu_x.$$

This is of interest when computing a weighted average of the radiance function over a pixel for instance. As computing a flux and computing a weighted average are essentially the same our discussion will be in terms of fluxes, implicitly assuming the alternative possibility.

2.1.3 The Bidirectional Reflectance Distribution Function

The *bidirectional reflectance distribution function* (BRDF) f_r for a surface point x, an incoming direction Θ_i and an outgoing direction Θ_o gives a measure for the amount of radiance impinging at point x along direction Θ_i that is reflected along direction Θ_o (Fig. 2.6). Formally:

$$f_r(x, \Theta_i, \Theta_o) = \frac{\partial L_{output}(x, \Theta_o)}{L_{input}(x, \Theta_i)\, |\Theta_i \cdot N_x|\, \partial\omega_i}, \tag{2.8}$$

where:

- $\partial L_{output}(x, \Theta_o)$ = the differential surface radiance reflected at point x in direction Θ_o as a result of the field radiance $L_{input}(x, \Theta_i)$ at point x through the differential solid angle $\partial\omega_i$ around direction Θ_i.

The units of the BRDF are $1/sr$. The BRDF defines the local reflective properties of the surface. For metals it is the macroscopic result of interactions with facets at a microscopic level. For these facets the reflection and refraction of light is perfectly described by the Fresnel equations and Snell's law. The distributions of their slopes and directions determine the macroscopic

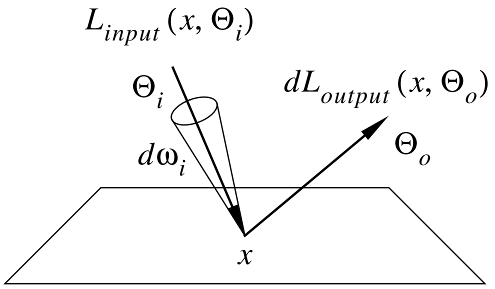


Figure 2.6: Explanation of the symbols used in the definition of the bidirectional reflectance distribution function.

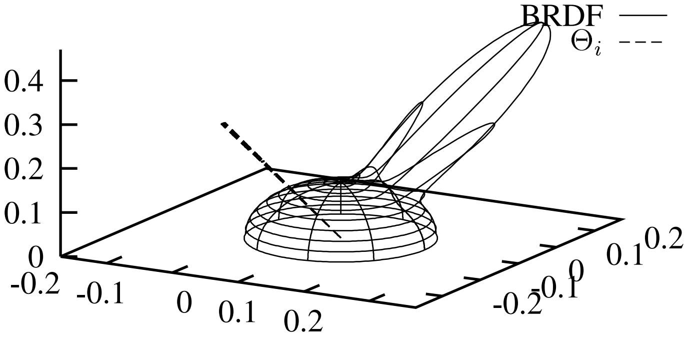


Figure 2.7: A polar diagram of a simple modified Phong BRDF for a fixed incoming direction. The BRDF is the sum of a constant diffuse part and a specular lobe around the perfect specular reflectance direction.

behaviour that is described by the BRDF. The BRDF can therefore be estimated on the basis of a model of the surface consisting of specular microfacets [21, 22, 42]. For other types of surfaces, such as plastics and varnished materials, the reflectance characteristics are the result of more complex sub-surface scattering. In general the BRDF can be measured directly and tabulated using a goniophotometer as described in [131]. For practical applications it is often represented by means of an empirical model [131, 100, 31]. The parameters of a model can then be estimated or fitted to actual measurements using a least squares method. For an overview of BRDF models we refer the reader to [102]. Figure 2.7 gives an example of a simple model. We will always assume that the BRDF is known and that it can be represented using a suitable model. A special case that is noteworthy is an ideal Lambertian reflector for which the BRDF is simply constant over the hemisphere:

$$f_r(x, \Theta_i, \Theta_o) = f_r, \quad \forall \Theta_i \in \Omega_i, \forall \Theta_o \in \Omega_o.$$

The definition of the BRDF can easily be extended to include refraction by considering the entire sphere of directions rather than only the hemisphere above the surface. We will implicitly assume this generalisation in our discussions.

Conservation of energy

A physically correct BRDF must be energy-conserving. For any incoming direction the power reflected over the hemisphere can never be more than the incident power. Any power that is not reflected is absorbed and transformed into heat. Formally, the so-called *directional-hemispherical reflectance* should be less than or at most equal to 1:

$$\rho(x, \Theta_i) = \int_{\Omega_o} f_r(x, \Theta_i, \Theta_o) \, |\Theta_o \cdot N_x| \, d\omega_o \leq 1, \quad \forall \Theta_i \in \Omega_i. \tag{2.9}$$

The reflectance is dimensionless.

Helmholtz reciprocity

Another important property of the BRDF is *Helmholtz reciprocity*. Because the reflective properties of a surface do not depend on the direction in which light travels, the roles of the incoming and the outgoing directions can be exchanged (Fig. 2.8):

$$f_r(x, \Theta_i, \Theta_o) = f_r(x, \Theta_o^{-1}, \Theta_i^{-1}). \tag{2.10}$$

BRDF models that are energy-conserving and reciprocal are called *physically plausible* [72, 73].

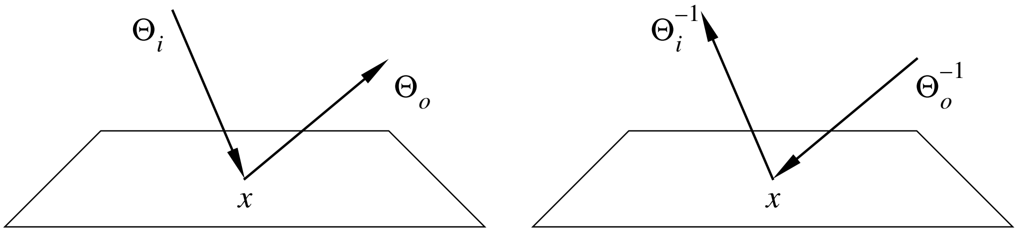


Figure 2.8: Helmholtz reciprocity: the value of the BRDF for a particular point and incoming and outgoing directions remains the same if the incoming and outgoing directions are exchanged.

2.2 The Rendering Equation

The previous paragraphs showed how field radiance and surface radiance are related through the invariance of radiance along a line and through local reflection. Combining these properties will lead to the rendering equation, which describes the global reflection of light throughout a given scene. It was first introduced in computer graphics by Kajiya (in terms of a quantity called *intensity*) [53].

Radiance leaving a surface is the sum of self-emitted radiance and reflected radiance. The BRDF describes the relation between field radiance and reflected surface radiance. Using the definition of the BRDF (2.8) the total surface radiance can therefore be written as follows:

$$\begin{aligned} L_o(x,\Theta_o) &= L_e(x,\Theta_o) + L_r(x,\Theta_o) \\ &= L_e(x,\Theta_o) + \int_{\Omega_i} L_i(x,\Theta_i) f_r(x,\Theta_i,\Theta_o) \left|\Theta_i \cdot N_x\right| d\omega_i. \end{aligned}$$

Because each field radiance corresponds to a surface radiance, as expressed by Eq. (2.4), this equation can be rewritten as

$$L_o(x,\Theta_o) = L_e(x,\Theta_o) + \int_{\Omega_i} L_o(y,\Theta_i) f_r(x,\Theta_i,\Theta_o) \left|\Theta_i \cdot N_x\right| d\omega_i,$$

where point y is such that $y = r(x, \Theta_i^{-1})$ (Fig. 2.9). We will only talk about surface radiance from here on and simply omit the "o" subscript. We will also refer to a hemisphere of incoming directions by "Ω^{-1}". The equation then becomes

$$L(x,\Theta_x) = L_e(x,\Theta_x) + \int_{\Omega_x^{-1}} L(y,\Theta_y) f_r(x,\Theta_y,\Theta_x) \left|\Theta_y \cdot N_x\right| d\omega_y. \tag{2.11}$$

This is the *rendering equation* expressed in terms of radiance, sometimes called the *radiance equation*. It is a linear integral equation known as a *Fredholm equation of the second kind*. As Arvo [3] noted though, it is particular in the sense that the domain of the integral is not the entire domain of the radiance function. The domain consists of a set of points along with a single direction for each point, which has measure 0 compared to the entire domain.

The rendering equation formally specifies the global illumination problem. Most commonly the light sources are known by way of their self-emitted radiance, the reflective properties of the scene by way of the BRDFs of the surfaces and the geometry of the scene by way of the ray casting function and the normals at the surfaces. The resulting surface radiance function is then determined unambiguously.

The rendering equation can be written in short using an integral operator:

$$L = L_e + TL, \tag{2.12}$$

where the integral operator or transport operator T is defined as

$$(Tf)(x,\Theta_x) = \int_{\Omega_x^{-1}} f(y,\Theta_y) f_r(x,\Theta_y,\Theta_x) \left|\Theta_y \cdot N_x\right| d\omega_y.$$

The operator transforms the surface radiance function $L(y,\Theta_y)$ into the surface radiance function $L_r(x,\Theta_x)$, which is the resulting radiance after one reflection. Gershbein *et al.* [32] and Arvo [3] show that the transport operator can be split in a *local reflection operator* that accounts for the local scattering of incident radiant power and a *field radiance operator* that accounts for the geometry of the scene. We refer the reader to the literature for further details.

For some purposes it is more convenient to write the integral of the rendering equation as an integral over all surfaces instead of an integral over all directions in the hemisphere. This can be achieved by means of a simple transformation:

$$\begin{aligned} L(x, \Theta_x) &= L_e(x, \Theta_x) \\ &\quad + \int_A L(y, \Theta_{y \to x}) f_r(x, \Theta_{y \to x}, \Theta_x) v(x, y) \frac{|\Theta_{y \to x} \cdot N_x| \, |\Theta_{y \to x} \cdot N_y|}{\|x - y\|^2} d\mu_y \\ &= L_e(x, \Theta_x) + \int_A L(y, \Theta_{y \to x}) f_r(x, \Theta_{y \to x}, \Theta_x) G(x, y) d\mu_y, \end{aligned} \tag{2.13}$$

where:

- $\Theta_{y \to x}$ = the direction from point y to point x,
- $v(x, y)$ = the visibility function, which is 1 if there are no objects between point x and point y, and 0 otherwise,
- $\|x - y\|$ = the Euclidean distance between point x and point y,
- $G(x, y)$ = the so-called *geometry function*.

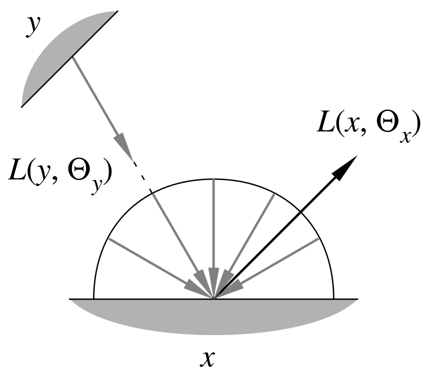


Figure 2.9: Explanation (in 2D) of the symbols used in the rendering equation.

2.3 The Potential Equation

Although adjoint integral equations were already known for solving integral problems in general, e.g. [30, pp. 197–198], and practically used in nuclear physics in the 60's and 70's, e.g. [48], the potential equation has only been introduced in computer graphics in '93 by Pattanaik [84, 83, 88, 87, 85, 89]. He presented the *potential function* or *potential capability* and the *potential equation* as the adjoints of the radiance function and the rendering equation. In '92 Smits *et al.* [119] had already proposed the discrete equivalent for diffuse radiation under the name of *importance*. While Pattanaik used the potential equation to actually solve the global illumination problem, Smits used it to direct the radiosity computations to regions that are important for the resulting image. This idea was extended to general discretised non-diffuse radiation by Aupperle and Hanrahan [7] and by Christensen *et al.* [13].

In this section we will present the definition of potential and mention its most important properties. From the definition follows an expression of radiant flux in terms of potential. Then we will derive the potential equation. The structure is similar to the discussion of radiance and the rendering equation, as these are dual to potential and the potential equation. Finally we will show that the rendering equation and the potential equation are actually adjoint.

The *potential* W for a certain point x and direction Θ_x is defined as the flux that is radiated through a given set S as the result of a single unit of radiance being emitted through a differential volume around point x and direction Θ_x (Fig. 2.10). Formally:

$$W(x,\Theta_x) = \frac{\partial^2\Phi_{output}}{L_{input}(x,\Theta_x)\partial\omega_x\partial A_{x\perp}}, \tag{2.14}$$

where:

- $\partial^2\Phi_{output}$ = the differential radiant power through the given set S, as a result of the input radiance $L_{input}(x,\Theta_x)$ through the differential surface area $\partial A_{x\perp}$ at point x and through the differential solid angle $\partial\omega_x$ around direction Θ_x.

Potential is dimensionless. As with the radiance function the potential function is invariant along each given direction when travelling through vacuum. Note that somewhat confusingly the potential function is not a potential function in the usual physical sense, but for the sake of continuity we will proceed using this name.

Of particular interest to us is the potential function for points and directions leaving the surfaces in the scene. It can be written as

$$W(x,\Theta_x) = \frac{\partial^2\Phi_{output}}{L_{input}(x,\Theta_x)\partial\omega_x\,|\Theta_x\cdot N_x|\,\partial\mu_x}. \tag{2.15}$$

Given this definition the radiant flux through a given set S as a result of self-emitted radiance L_e can now be expressed in terms of the potential:

$$\Phi = \int_A\int_{\Omega_x} L_e(x,\Theta_x)W(x,\Theta_x)\,|\Theta_x\cdot N_x|\,d\omega_x d\mu_x. \tag{2.16}$$

Or, using the same dot product notation as defined for the radiance function in Eq. (2.7):

$$\Phi = \langle L_e, W\rangle. \tag{2.17}$$

Just as the radiance function is governed by the rendering equation the potential function is governed by an integral equation called the *potential equation*. The potential W for a point x and a direction Θ_x is also a sum of two terms. If (x, Θ_x) belongs to the given set S, any radiance will contribute to the flux directly. This can be taken into account by a first term $W_e(x, \Theta_x)$ which will be 1 in this case, and 0 otherwise, as defined for Eq. (2.6). Point x and direction Θ_x can also contribute indirectly to the flux through reflection, which can be expressed in terms of the potential values at the nearest point from point x along direction Θ_x (Fig. 2.11). Formally:

$$\begin{aligned} W(x,\Theta_x) &= W_e(x,\Theta_x) + W_r(x,\Theta_x) \\ &= W_e(x,\Theta_x) + \int_{\Omega_y} W(y,\Theta_y) f_r(y,\Theta_x,\Theta_y)\,|\Theta_y \cdot N_y|\,d\omega_y, \end{aligned} \tag{2.18}$$

where $y = r(x, \Theta_x)$. The function $W_e(x, \Theta_x)$ can be considered as *self-emitted potential*, in analogy with the self-emitted radiance function $L_e(x, \Theta_x)$; hence the symbol we use in our discussions.

Again the potential equation can be written in short using an integral operator:

$$W = W_e + T^* W, \tag{2.19}$$

where the integral operator or transport operator T^* is defined as

$$(T^* f)(x, \Theta_x) = \int_{\Omega_y} f(y,\Theta_y) f_r(y,\Theta_x,\Theta_y)\,|\Theta_y \cdot N_y|\,d\omega_y.$$

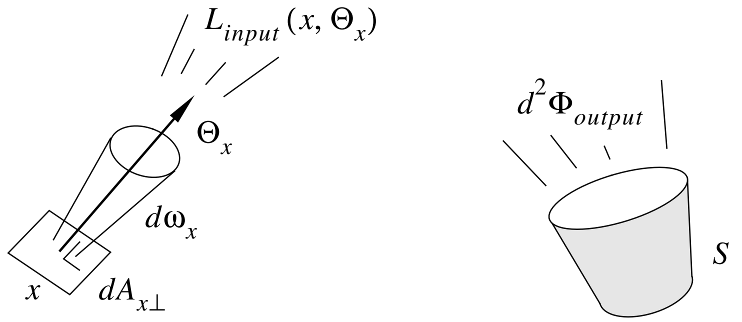


Figure 2.10: Explanation of the symbols used in the definition of potential. The set of points and directions S can be seen as a detector that measures the differential flux as a result of radiance that is emitted through a differential volume around point x and direction Θ_x. The measured ratio gives the potential for point x and direction Θ_x.

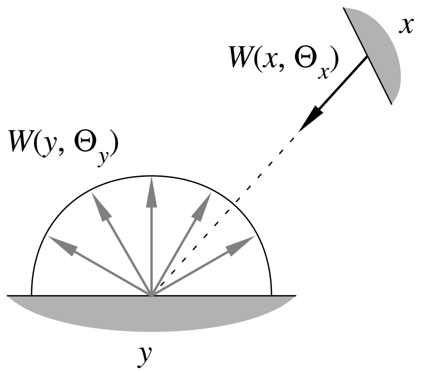


Figure 2.11: Explanation of the symbols used in the potential equation.

The potential equation can also be written as an integral over all surfaces instead of an integral over all directions in the hemisphere:

$$\begin{aligned} W(x,\Theta_x) &= W_e(x,\Theta_x) \\ &\quad + \int_A W(y,\Theta_{y\to z}) f_r(y,\Theta_x,\Theta_{y\to z}) v(y,z) \frac{|\Theta_{y\to z}\cdot N_y|\,|\Theta_{y\to z}\cdot N_z|}{\|y-z\|^2} d\mu_z \\ &= W_e(x,\Theta_x) + \int_A W(y,\Theta_{y\to z}) f_r(y,\Theta_x,\Theta_{y\to z}) G(y,z) d\mu_z. \end{aligned} \tag{2.20}$$

Adjointness of the Rendering Equation and the Potential Equation

Two operators T and T^* and their respective equations

$$\begin{aligned} f &= f_e + Tf \\ g &= g_e + T^* g \end{aligned}$$

are called *adjoint* with respect to a dot product $\langle \cdot, \cdot \rangle$ if and only if

$$\langle f, g_e \rangle = \langle f_e, g \rangle.$$

for each f_e and g_e. The rendering equation and the potential equation –or more precisely: their respective operators– are therefore adjoint, because their corresponding expressions (2.6) and (2.16) for the flux Φ must yield the same results for any set S (i.e. for any function $W_e(x,\Theta_x)$) and for any self-emitted radiance function $L_e(x,\Theta_x)$:

$$\Phi = \langle L, W_e \rangle = \langle L_e, W \rangle.$$

2.4 The Global Reflectance Distribution Function

From the previous sections it is clear that the radiance function $L(x, \Theta_x)$ is defined with respect to a fixed self-emitted radiance function $L_e(x, \Theta_x)$, and that the potential function $W(x, \Theta_x)$ is defined with respect to a fixed function $W_e(x, \Theta_x)$ corresponding to the set S. The relationship between $L(x, \Theta_x)$ and $W(x, \Theta_x)$ is not defined by the expressions, making it difficult to combine them into some model or algorithm. In [65] we have therefore proposed the notion of the *global reflectance distribution function* (GRDF) which is defined with respect to a given scene but which does not depend on any particular emission function such as $L_e(x, \Theta_x)$ nor on a fixed set such as S.

Similarly to the discussions of radiance and potential in the previous sections this section will present the definition of the GRDF and the defining integral equations. We will also express flux in terms of the GRDF.

2.4.1 Definition

The definition bears some similarity to the approach often taken in system theory, where a system is studied on the basis of its response to some input signal. The *global reflectance distribution function* F_r for points and directions (x, Θ_x) and (y, Θ_y) expresses the differential amount of radiance leaving point y along direction Θ_y as the result of a single unit of emitted radiance from point x on a surface along direction Θ_x (Fig. 2.12). Formally:

$$\begin{aligned} F_r(x, \Theta_x, y, \Theta_y) &= \frac{\partial^2 L_{output}(y, \Theta_y)}{L_{input}(x, \Theta_x)\partial\omega_x \partial A_{x\perp}} \\ &= \frac{\partial^2 L_{output}(y, \Theta_y)}{L_{input}(x, \Theta_x)\partial\omega_x \left|\Theta_x \cdot N_x\right| \partial\mu_x}. \end{aligned} \tag{2.21}$$

The GRDF may be considered as a generalisation of the BRDF. While the latter only specifies the local behaviour of light reflecting at a single surface, the GRDF defines the global illumination effects of light reflecting through a complete scene. The units of the GRDF are $1/m^2sr$. It is theoretically defined over the entire space of points and directions and therefore 10-dimensional. As before we will only use it at the surfaces of the scene though. Because we assume that there

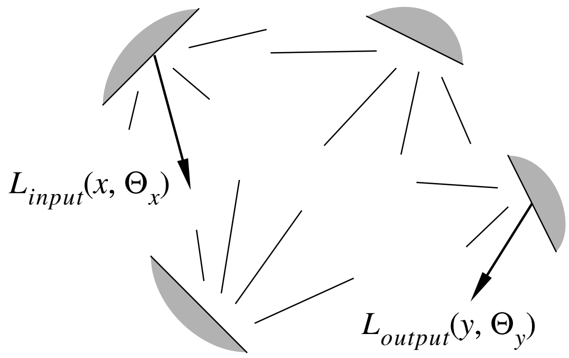


Figure 2.12: The global reflectance distribution function gives a measure for the fraction of the radiance emitted at point x along direction Θ_x that is eventually output at point y along direction Θ_y through all possible reflections throughout the scene.

is no participating medium the function is constant when moving point x along direction Θ_x and point y along direction Θ_y between two surfaces, by virtue of the invariance of the *input* and *output* radiance functions along these directions.

2.4.2 Flux in Terms of the GRDF

Now that the GRDF has been defined we can use it to determine the flux through the given set of points and directions S (defined by $W_e(x, \Theta_x)$) and with respect to the given self-emitted radiance function $L_e(x, \Theta_x)$. From the definition of the GRDF (2.21) and the expression of the flux in terms of radiance (2.6) we can derive

$$\begin{aligned}\Phi \;=\; & \int_A \int_{\Omega_x} \int_A \int_{\Omega_y} L_e(x, \Theta_x) W_e(y, \Theta_y) F_r(x, \Theta_x, y, \Theta_y) \\ & \qquad |\Theta_x \cdot N_x| \, |\Theta_y \cdot N_y| \, d\omega_y d\mu_y d\omega_x d\mu_x. \end{aligned} \tag{2.22}$$

Note that it is only at this time that the functions $L_e(x, \Theta_x)$ and $W_e(x, \Theta_x)$ come into play; the GRDF itself is totally independent of them. If required, it remains possible to express the radiance function with respect to a given $L_e(x, \Theta_x)$ and the potential function with respect to a given $W_e(x, \Theta_x)$ in terms of the GRDF:

$$L(x, \Theta_x) = \int_A \int_{\Omega_x} L_e(x, \Theta_x) F_r(x, \Theta_x, y, \Theta_y) \, |\Theta_x \cdot N_x| \, d\omega_x d\mu_x,$$

$$W(x, \Theta_x) = \int_A \int_{\Omega_y} W_e(y, \Theta_y) F_r(x, \Theta_x, y, \Theta_y) \, |\Theta_y \cdot N_y| \, d\omega_y d\mu_y.$$

2.4.3 Equations Defining the GRDF

The GRDF is specified by a set of two integral equations. Firstly, one can look at the behaviour of the function for a fixed (x, Θ_x). Similarly to the derivation of the rendering equation, the radiance at (y, Θ_y) is the result of two contributions. If both pairs of points and directions are equal, the *input* radiance contributes directly to the *output* radiance. In this case the *output* radiance is not differential but finite, so that the fraction is a Dirac impulse. The second contribution results from the reflection of incoming radiance at point y which arrives from points z, possibly through multiple reflections throughout the scene (Fig. 2.14). This can be expressed formally as

$$\begin{aligned} F_r(x, \Theta_x, y, \Theta_y) \;=\; & \delta(x, \Theta_x, y, \Theta_y) \\ & + \int_{\Omega_y^{-1}} f_r(y, \Theta_z, \Theta_y) F_r(x, \Theta_x, z, \Theta_z) \, |\Theta_z \cdot N_y| \, d\omega_z, \end{aligned} \tag{2.23}$$

where $z = r(y, \Theta_z^{-1})$. The function $\delta(x, \Theta_x, y, \Theta_y)$ is the Dirac impulse which is 0 if $(x, \Theta_x) \neq (y, \Theta_y)$, but which integrates to 1 over the domain of (x, Θ_x) for a given (y, Θ_y) and vice versa:

$$\begin{aligned} \int_A \int_{\Omega_x} \delta(x, \Theta_x, y, \Theta_y) \, |\Theta_x \cdot N_x| \, d\omega_x d\mu_x &= 1, \\ \int_A \int_{\Omega_y} \delta(x, \Theta_x, y, \Theta_y) \, |\Theta_y \cdot N_y| \, d\omega_y d\mu_y &= 1. \end{aligned}$$

Alternatively, one can look at the behaviour of the function for a fixed (y, Θ_y). This dual viewpoint resembles that of the potential equation. The question here is how radiance at (x, Θ_x)

can contribute to the *output* radiance. Once again, if both pairs of points and directions are equal the *input* radiance contributes directly to the *output* radiance, which is expressed by the Dirac impulse. The *input* radiance can also contribute indirectly, through reflection in all directions around point z which is seen by point x in the direction Θ_x (Fig. 2.14). This results in another recursive equation:

$$\begin{aligned} F_r(x,\Theta_x,y,\Theta_y) &= \delta(x,\Theta_x,y,\Theta_y) \\ &\quad + \int_{\Omega_z} f_r(z,\Theta_x,\Theta_z) F_r(z,\Theta_z,y,\Theta_y)\,|\Theta_z \cdot N_z|\,d\omega_z, \end{aligned} \tag{2.24}$$

where $z = r(y, \Theta_z^{-1})$.

It is interesting to see that these dual equations are fully equivalent, by virtue of the bidirectional nature of light. The fraction of the radiance emitted from one point that is received by another point equals the fraction that would be received if the roles of the emitter and the receiver are exchanged, thus reversing the paths the light follows. This property can be expressed elegantly in terms of the GRDF:

$$F_r(x,\Theta_x,y,\Theta_y) = F_r(\tilde{y},\Theta_y^{-1},\tilde{x},\Theta_x^{-1}),$$

where $\tilde{x} = r(x,\Theta_x)$ and $\tilde{y} = r(y,\Theta_y)$ as shown in Fig. 2.15. This expression is equivalent to the Helmholtz reciprocity of the BRDF (Eq. (2.10)).

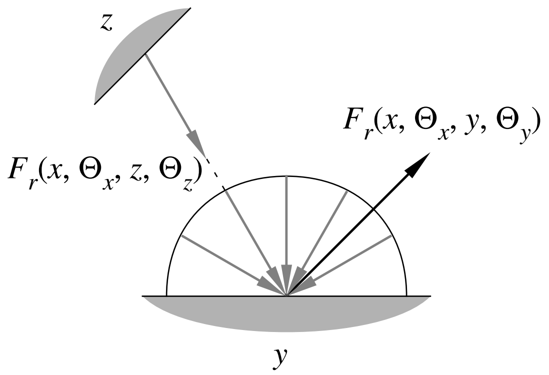


Figure 2.13: Explanation of the symbols used in the first equation that defines the GRDF. It is analogous to the rendering equation.

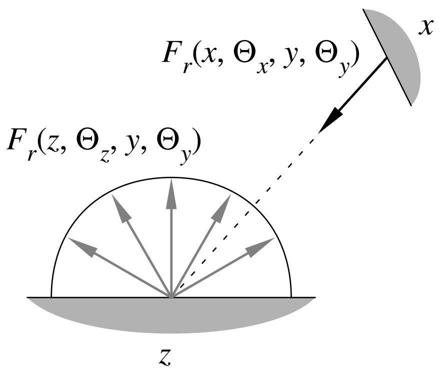


Figure 2.14: Explanation of the symbols used in the second equation that defines the GRDF. It is analogous to the potential equation.

Substitution of these properties in both sides of the second integral equation (2.24) yields

$$\begin{aligned} F_r(\tilde{y},\Theta_y^{-1},\tilde{x},\Theta_x^{-1}) &= \delta(x,\Theta_x,y,\Theta_y) \\ &+ \int_{\Omega_z} f_r(z,\Theta_z^{-1},\Theta_x^{-1}) F_r(\tilde{y},\Theta_y^{-1},\tilde{z},\Theta_z^{-1})\, |\Theta_z \cdot N_z|\, d\omega_z. \end{aligned}$$

Renaming the variables appropriately results in the first integral equation (2.23). It implies that any of the two dual equations in combination with the reciprocal property is sufficient to define the GRDF unambiguously.

The integral equations can again be rewritten in short, using operators. Using the same operator T as the rendering equation (2.12), Eq. (2.23) becomes

$$F_r(x,\Theta_x,y,\Theta_y) = \delta(x,\Theta_x,y,\Theta_y) + T F_r(x,\Theta_x,y,\Theta_y), \tag{2.25}$$

where T operates on the variables y and Θ_y, treating x and Θ_x as constants. Using the same operator T^* as the potential equation (2.19), Eq. (2.24) on the other hand becomes

$$F_r(x,\Theta_x,y,\Theta_y) = \delta(x,\Theta_x,y,\Theta_y) + T^* F_r(x,\Theta_x,y,\Theta_y), \tag{2.26}$$

where T^* now operates on the variables x and Θ_x, treating y and Θ_y as constants.

The occurrence of a Dirac impulse in both integral equations may seem a bit awkward for practical use in algorithms. Luckily one will always want to integrate the GRDF in order to obtain some flux. Therefore it is an advantage rather than a disadvantage, since integrating a Dirac impulse is easy by virtue of its definition.

We have defined the GRDF in terms of *outgoing* input radiance and *outgoing* output radiance. The four alternative combinations of incoming/outgoing input/output radiance lead to different definitions and different sets of two basic integral equations. Some of these could be slightly more practical than the other ones for specific applications. We will simply stick to the presented definition.

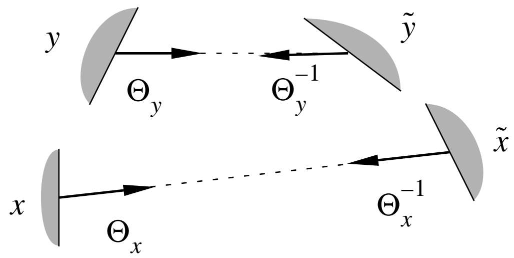


Figure 2.15: Explanation of the symbols used in the reciprocal property of the GRDF.

2.5 Summary

The most important radiometric quantities to describe radiant power are radiance and flux. They have total and spectral variants, depending on whether the entire visual spectrum of wavelengths is considered or only a single wavelength. The radiance function describes the amount of light radiated from each position and direction as the result of some light sources in the scene. We are mostly interested in surface radiance (leaving the surfaces) and field radiance (incident to the surfaces). Under the assumption or approximation that only surfaces reflect light and that the medium does not scatter light, radiance is invariant along its direction. This property relates surface radiance to field radiance.

The starting point to describe global illumination is local illumination. The local reflective properties of a surface are described by its bidirectional reflectance distribution function (BRDF). It relates light reflected from the surface along an outgoing direction to light coming in along an incident direction. This property therefore also relates surface radiance to field radiance.

Combining both relations between surface radiance and field radiance yields the rendering equation (2.11). This integral equation expresses surface radiance as a function of self-emitted radiance, geometry and reflective properties of the surfaces. Together with the expression for radiant flux in terms of radiance (2.6) it yields a first mathematical formalisation of the global illumination problem:

$$\Phi = \int_A \int_{\Omega_x} L(x,\Theta_x) W_e(x,\Theta_x) \left|\Theta_x \cdot N_x\right| d\omega_x d\mu_x, \tag{2.6}$$

$$L(x,\Theta_x) = L_e(x,\Theta_x) + \int_{\Omega_x^{-1}} L(y,\Theta_y) f_r(x,\Theta_y,\Theta_x) \left|\Theta_y \cdot N_x\right| d\omega_y. \tag{2.11}$$

The potential function gives a measure for the fraction of light emitted at each position and direction that eventually reaches a given detector (a set of points and directions). As with radiance potential can be described by means of the potential equation (2.18), which is adjoint to the rendering equation. Together with the expression for radiant flux in terms of potential (2.16) it yields an alternative description of the global illumination problem:

$$\Phi = \int_A \int_{\Omega_x} L_e(x,\Theta_x) W(x,\Theta_x) \left|\Theta_x \cdot N_x\right| d\omega_x d\mu_x, \tag{2.16}$$

$$W(x,\Theta_x) = W_e(x,\Theta_x) + \int_{\Omega_y} W(y,\Theta_y) f_r(y,\Theta_x,\Theta_y) \left|\Theta_y \cdot N_y\right| d\omega_y. \tag{2.18}$$

We have introduced the global reflectance distribution function (GRDF), which gives a measure for the fraction of light emitted at each position and direction that is eventually radiated through any other position and direction. As such it is a 10-dimensional function of pairs of all combinations of points and directions. It only depends on the geometry and the reflective properties of the scene, not on the self-emitted radiance function as the radiance function does, nor on the self-emitted potential function as the potential function does. Its behaviour is described by two equivalent integral equations (2.23) and (2.24) that correspond to the rendering equation and to the potential equation respectively. The expression of the flux in terms of the GRDF (2.22) now contains the factors depending on the self-emitted radiance and potential. Together this set of equations forms a third description of the global illumination problem:

$$\begin{aligned}\Phi &= \int_A \int_{\Omega_x} \int_A \int_{\Omega_y} L_e(x,\Theta_x) W_e(y,\Theta_y) F_r(x,\Theta_x,y,\Theta_y) \\ &\qquad \left|\Theta_x \cdot N_x\right| \left|\Theta_y \cdot N_y\right| d\omega_y d\mu_y d\omega_x d\mu_x,\end{aligned} \tag{2.22}$$

$$\begin{aligned} F_r(x, \Theta_x, y, \Theta_y) &= \delta(x, \Theta_x, y, \Theta_y) \\ &+ \int_{\Omega_y^{-1}} f_r(y, \Theta_z, \Theta_y) F_r(x, \Theta_x, z, \Theta_z) \left|\Theta_z \cdot N_y\right| d\omega_z, \end{aligned} \tag{2.23}$$

$$\begin{aligned} F_r(x, \Theta_x, y, \Theta_y) &= \delta(x, \Theta_x, y, \Theta_y) \\ &+ \int_{\Omega_z} f_r(z, \Theta_x, \Theta_z) F_r(z, \Theta_z, y, \Theta_y) \left|\Theta_z \cdot N_z\right| d\omega_z. \end{aligned} \tag{2.24}$$

In Chapter 4 we will show that these alternative mathematical models are not only of theoretical interest. Applying the same standard mathematical techniques to each of them will lead to different rendering algorithms. We will start by explaining these techniques in Chapter 3.

Chapter 3

Monte Carlo Methods

In this chapter we will present the basic principles of Monte Carlo methods that we require for application to the global illumination problem. More extensive discussions of Monte Carlo methods in general can be found in the books by Hammersley and Handscomb [40], by Kalos and Whitlock [54] and by Shreider [114]. Ermakow presents the principles in a mathematically more elaborate and precise way in [30]. Davis and Rabinowitz [23] discuss Monte Carlo methods in the broader context of numerical integration.

3.1 Introduction

Monte Carlo methods have a long history. In 1873 Hall described a noteworthy experiment for estimating the value of π by throwing a needle on a paper with parallel lines at equal distances and then counting the number of times the needle crosses the lines [36]. Monte Carlo methods only became practical with the advent of automatic computers though. During World War II John von Neumann used Monte Carlo simulations in his work on the atomic bomb. After the war Metropolis and Ulam published the first systematic study on the subject [77]. Monte Carlo techniques gained a great popularity during the 50's and 60's. Later on they received a lot of criticism due to their indiscriminate application to every conceivable mathematical problem.

Monte Carlo methods compute results on the basis of random numbers. Historically they are often applied to problems of a probabilistic nature. Monte Carlo methods can then be straightforward simulations of the actual random processes. In nuclear physics for instance the behaviour of neutrons is essentially random. Macroscopic neutron fluxes and other quantities can then be computed by means of a so-called *analogue simulation*. Individual particles are emitted, reflected, scattered and absorbed stochastically, in a way that is faithful to reality. As in reality the computed quantities are subject to some probabilistic uncertainty also. The uncertainty on the computed results can be reduced by performing a large number of simulations. For many natural phenomena it is impossible to even come close to the actual number, so more intelligent optimisations have to be found.

Monte Carlo methods can also handle problems of a deterministic nature. In this case a problem is formalised by means of a mathematical model that can be seen as a description of a random process. Simulation of the random process yields a solution to the original problem. Monte Carlo methods can therefore be helpful for problems that cannot be solved easily by other analytical or numerical means. Even problems of a probabilistic nature are usually formulated as a mathematical problem and can be worked on as such before solving them with possibly different Monte Carlo techniques.

The technique of solving a different problem than the original problem with Monte Carlo methods is sometimes called *sophisticated Monte Carlo*. It results from analysis and abstraction of the problem and is mostly aimed at improving the efficiency of the simulations. This is essentially what happens for the global illumination problem as well. The stochastic behaviour of photons is formalised by the equivalent mathematical models presented in Chapter 2. As a result Monte Carlo methods do not necessarily simulate the emission, reflection and absorption of photons directly but simulate the underlying random processes of these models. We will discuss these approaches to the global illumination problem in Chapter 4, but first we will present the basic principles of Monte Carlo methods in general.

3.2 Monte Carlo Integration

All Monte Carlo methods can be regarded directly or indirectly as techniques for numerical integration. Consider the definite integral of the function $f(x)$ over the interval $[0, 1]$, the *estimand*:

$$I = \int_0^1 f(x)dx. \tag{3.1}$$

We will assume that $f(x)$ is a function in $L^2(0, 1)$, i.e. $\int_0^1 f^2(x)dx$ exists. The integral can be estimated by taking a uniform random number ξ over $[0, 1]$ and evaluating $f(\xi)$. $f(\xi)$ is a so-called *primary estimator* of the integral, which we will denote with angular brackets:

$$\langle I \rangle_{prim} = f(\xi). \tag{3.2}$$

If the estimator is evaluated for a specific sample it is called an *estimate*. It is equal to the surface area of the rectangle below the evaluated point (Fig. 3.1). The estimator is called *unbiased* because its expected value $E(\langle I \rangle)$ is equal to the actual integral (the probability density function is equal to 1 for now):

$$E(\langle I \rangle_{prim}) = \int_0^1 f(x)dx = I.$$

We will assume that it is possible to take perfectly random, i.e. uncorrelated, uniform samples. In practice one usually has to resort to computer-generated pseudo-random numbers that have passed certain statistical tests. More details on the generation of random and pseudo-random numbers can be found in the standard references.

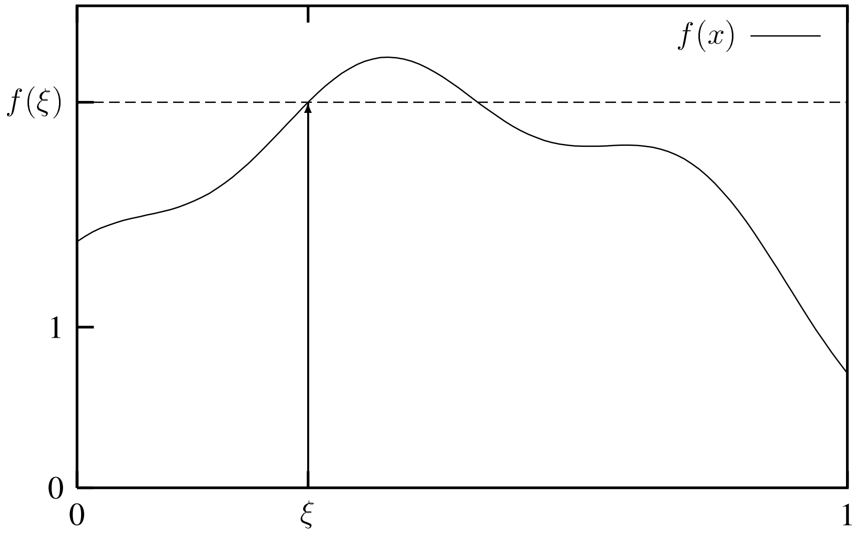


Figure 3.1: A simple primary estimator of the definite integral of a function $f(x)$ over an interval is the surface area of the rectangle defined by the evaluation $f(\xi)$ at a uniformly selected random point ξ.

There is an uncertainty on the result which is expressed by the *variance* V and the *standard error* or *standard deviation* σ of the estimator:

$$\begin{aligned} V(\langle I \rangle_{prim}) = \sigma^2_{prim} &= \int_0^1 [f(x) - I]^2 dx \\ &= \int_0^1 f^2(x) dx - I^2. \end{aligned} \tag{3.3}$$

This expression is not very helpful as such, as neither of the integrals are known in practice, but it can serve as a theoretical reference against which to compare optimisations.

The variance is usually unacceptably large when only taking this single sample. The basis of Monte Carlo methods is to reduce the uncertainty by taking more, say N, independent samples ξ_i and averaging their corresponding primary estimators into a *secondary estimator*. This is equivalent to rewriting the original integral (3.1) as a sum of integrals (Fig. 3.2):

$$\begin{aligned} I &= \int_0^1 f(x) dx \\ &= \sum_{i=1}^{N} \int_0^1 \frac{f(x)}{N} dx \\ &= \sum_{i=1}^{N} I_i. \end{aligned}$$

The secondary estimator is then the sum of the primary estimators:

$$\begin{aligned} \langle I \rangle_{sec} &= \sum_{i=1}^{N} \langle I_i \rangle_{prim} \\ &= \frac{1}{N} \sum_{i=1}^{N} f(\xi_i). \end{aligned} \tag{3.4}$$

This average of unbiased primary estimators is unbiased as well. Considering the set of N samples as a single N-dimensional sample the expected value can be computed as follows:

$$\begin{aligned} E(\langle I \rangle_{sec}) &= \int_0^1 \dots \int_0^1 \frac{1}{N} \sum_{i=1}^{N} f(x_i) dx_1 \dots dx_N \\ &= I. \end{aligned}$$

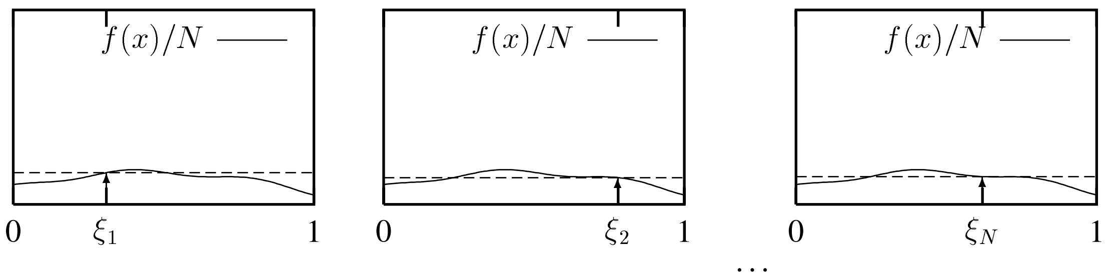


Figure 3.2: The secondary estimator of an integral is the average of a set of primary estimators.

The variance is reduced by a factor N:

$$\begin{aligned}\sigma^2_{sec} &= \int_0^1 \dots \int_0^1 [\frac{1}{N}\sum_{i=1}^{N} f(x_i)]^2 dx_1 \dots dx_N - I^2 \\ &= \frac{1}{N}\int_0^1 f^2(x)dx - \frac{1}{N}I^2 \\ &= \sigma^2_{prim}/N.\end{aligned} \tag{3.5}$$

This is a fundamental observation. It means that the standard error of any secondary estimator of this type is proportional to $1/\sqrt{N}$, which is not all that spectacular when compared to other numerical techniques. Research on Monte Carlo methods is therefore mostly concentrated on reducing the other factor, being the variance of the primary estimator. Some important variance reducing techniques such as stratified sampling, importance sampling and control variates will be discussed briefly in the following sections. Most of them make use of additional information that is available about the integral. Almost invariably they can be seen as transformations of the original integral.

For simplicity we have only discussed one-dimensional integrals. The same Monte Carlo principles can be easily extended to multi-dimensional integrals though. An obvious way would be to treat the multi-dimensional integrals as multiple one-dimensional integrals. However, as with deterministic methods the number of function evaluations, which we are trying to minimise, increases exponentially with the dimension of the integral. Computing a d-dimensional integral with N samples per dimension requires N^d samples. This increase quickly becomes prohibitive for problems such as the global illumination problem where the sampling of pixels, light sources and hemispheres and effects such as depth of field and motion blur all correspond to different dimensions. The effect is known as the curse of dimensionality.

An alternative approach is to consider the multi-dimensional integrals as a whole and to take samples over their entire domains. All definitions and properties of one-dimensional Monte Carlo integration still hold. Integration domains that are irregularly shaped may pose a problem in practice. Rejection sampling then presents a simple albeit often inefficient solution by repeatedly taking a sample in the bounding box of the integration domain until it lies inside of it. More sophisticated solutions apply importance sampling which transforms the integration domain, as will be discussed in Section 3.4. It is worth noting that the convergence rate of Monte Carlo methods is not affected by the dimension of the integral. This makes them an attractive alternative for solving high-dimensional integrals where the complexity of deterministic techniques becomes a problem.

3.3 Stratified Sampling

One can easily imagine that the clumping together of sample points reduces their effectiveness considerably. Without any prior knowledge about the integrand the most effective distribution of the samples is as uniform as possible. *Stratified sampling* tries to meet this goal by splitting the integration domain into several subdomains and estimating the partial integrals in each of these subdomains using Monte Carlo integration. This approach ensures that each of the subdomains at least gets a given number of samples. In its simplest form all subdomains are of equal size and each of them gets a single sample. The original integral (3.1), to be estimated with N samples, can be rewritten as

$$\begin{aligned} I &= \int_0^1 f(x)dx \\ &= \sum_{i=1}^{N} \int_{A_i} f(x)dx \\ &= \sum_{i=1}^{N} I_i, \end{aligned}$$

where A_i is subdomain or *stratum* i of the entire integration domain; in this case the interval $[(i-1)/N, i/N]$. If all samples x_i are selected with uniform probability density N over these subdomains A_i the primary estimators for I_i are $f(\xi_i)/N$. The total estimator becomes the sum of these partial primary estimators:

$$\begin{aligned} \langle I \rangle_{strat} &= \sum_{i=1}^{N} \langle I_i \rangle_{prim} \\ &= \frac{1}{N} \sum_{i=1}^{N} f(\xi_i). \end{aligned} \tag{3.6}$$

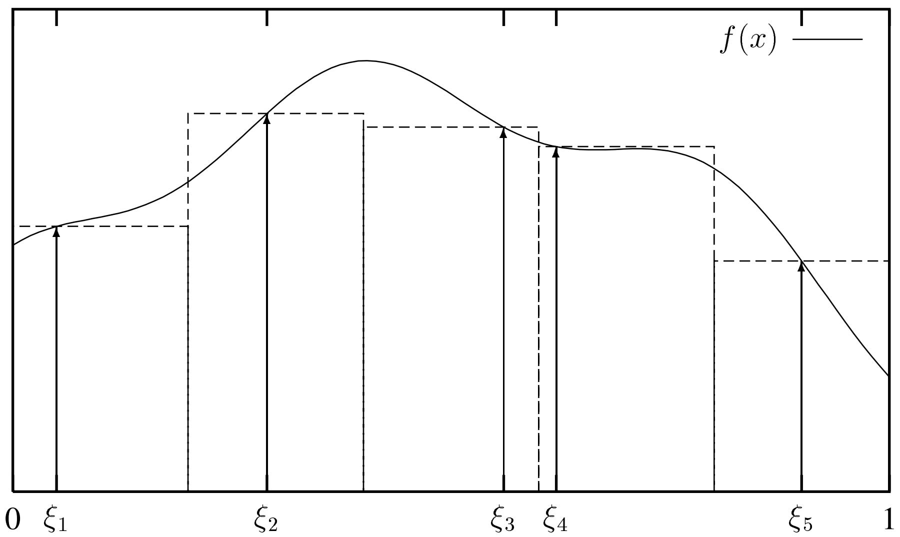


Figure 3.3: Stratified sampling consists in subdividing the original integration domain and estimating the resulting partial integrals by means of one or more samples per integral.

This expression is identical to expression (3.4) for the ordinary secondary estimator. The samples are chosen in a different way though. While the expected value of the estimator remains the exact integral for any number of samples, the variance differs from the variance of the ordinary secondary estimator. It is the sum of the variances of the partial estimators:

$$\begin{aligned}\sigma^2_{strat} &= \sum_{i=1}^{N}\left[\int_{A_i}[\frac{f(x_i)}{N}]^2 N dx_i - I_i^2\right] \\ &= \frac{1}{N}\int_0^1 f^2(x)dx - \sum_{i=1}^{N} I_i^2.\end{aligned} \tag{3.7}$$

Due to the latter term always being larger than or equal to the I^2-term of Eq. (3.5), the variance of stratified sampling is always less than or equal to the variance of the ordinary secondary estimator with the same number of samples:

$$\sigma^2_{strat} \leq \sigma^2_{sec}. \tag{3.8}$$

Yet the cost of distributing the samples over the strata is negligible, which makes stratified sampling a fundamental optimisation. It is different from most other variance reduction techniques in that each sample is used to estimate a different integral. The estimator as a whole is unbiased because the average of M estimators with N strata each, converges to the exact value for $M \to \infty$. In practice however one will construct $M \times N$ strata instead. From this point of view stratified sampling has the potential to perform better than the typical $1/\sqrt{N}$ convergence rate of Monte Carlo methods.

The success of the variance reduction partially depends on the choice of the strata. In one dimension equally sized subintervals are the most natural alternative, if no further information about the integrand is known. Choosing different strata such as unions of small randomly distributed subintervals would in general decrease the efficiency, as the technique derives its improvement from any coherence in the integrand. A sample in a stratum is supposed or at least hoped to be representative for the entire function in that region. For integrands that change rapidly within the strata stratified sampling yields little additional variance reduction over an ordinary secondary estimator.

For multi-dimensional integrals the same rules hold. Choosing appropriate strata is much less obvious however. They should be "compact" in a sense, such that the integrand is likely to be more or less constant in these regions. Prior knowledge about the integrand may be helpful for determining acceptable strata. For a square multi-dimensional integration domain splitting each interval corresponding to a dimension in N subintervals yields N^d strata and samples, as with the ordinary secondary estimator. This approach suffers from the same curse of dimensionality as discussed in the previous section.

N-rooks sampling is a computationally cheap alternative to extend stratified sampling to higher dimensions. The technique is named after the chess problem, in two dimensions. The two-dimensional square integration domain is subdivided in N by N squares. N of these squares are selected in a random N-rooks pattern, as illustrated in Fig. 3.4. A point is then sampled in each of the selected squares. The technique ensures that each row and each column of squares receive exactly one sample.

N-rooks sampling can be further extended to any arbitrary number of dimensions d. For this purpose each integration interval of the d dimensions is subdivided in N subintervals. For each dimension i a random permutation $P_{i1}, \ldots, P_{iN}$ of the indices $1, \ldots, N$ is created in a preprocessing step. For each sample s the intervals to be selected in the consecutive dimensions are indicated by the permutations

$$(P_{1s}, P_{2s}, P_{3s}, \ldots).$$

Figure 3.5 gives an example. The permutations must be randomised for each individual integral to avoid correlation between the results.

N-rooks sampling is particularly interesting if the integrand is separable over its dimensions (e.g. if the integrand is constant along one of the dimensions) because the samples are perfectly stratified in each individual dimension. In general one can expect the efficiency gain to decrease as the number of samples increases, as they can start clumping together again over the domain. Other techniques may then become more attractive in spite of the increased complexity.

Quasi-Monte Carlo methods may present an interesting alternative to classical stratification techniques. The goal of stratification is to sample the integrand more efficiently by spreading the samples evenly over the domain. For Monte Carlo methods the samples are still random in principle. In practice however they are generated by a computer and only pseudo-random as a result. Quasi-Monte Carlo methods drop the randomness and concentrate on the uniform distribution of the samples, e.g. [136, 120, 121]. The basic property of unbiasedness no longer holds for these distributions as they are deterministic rather than stochastic. Several theorems

	2			
4				
				5
		1		
			3	

Figure 3.4: Stratification of the samples in one dimension can be extended to two dimensions using N-rooks sampling. The two-dimensional domain is subdivided in a grid of $N \times N$ squares. N squares, indicated by a numeral here, are selected in a random N-rooks pattern. Each selected square receives exactly one sample.

	Dimension #				
Interval #	1	2	3	4	...
Sample #1	3	4	3	1	...
#2	2	1	4	5	...
#3	4	5	2	2	...
#4	1	2	1	3	...
#5	5	3	5	4	...

Figure 3.5: N-rooks sampling in its turn can be extended to higher dimensions by creating a random permutation of the interval indices $1, \ldots, N$ for each dimension (shown as columns here) in a preprocessing step and selecting the correct index for each sample (shown as rows here) and each dimension. The first two dimensions in this example correspond to the two-dimensional N-rooks pattern of Fig. 3.4.

have been derived however that bound their integration errors on the condition that the integrand is of bounded variation, e.g. [136]. Due to this desirable property Monte Carlo techniques are sometimes developed and analysed as such and then fed with quasi-Monte Carlo distributions in anticipation of an improved convergence.

3.4 Importance Sampling

Another powerful optimisation for Monte Carlo methods is *importance sampling*. Consider the basic integral (3.1) again. Some regions of the integral may be more important than other regions because they have higher function values. Even a few samples in these regions may have a great

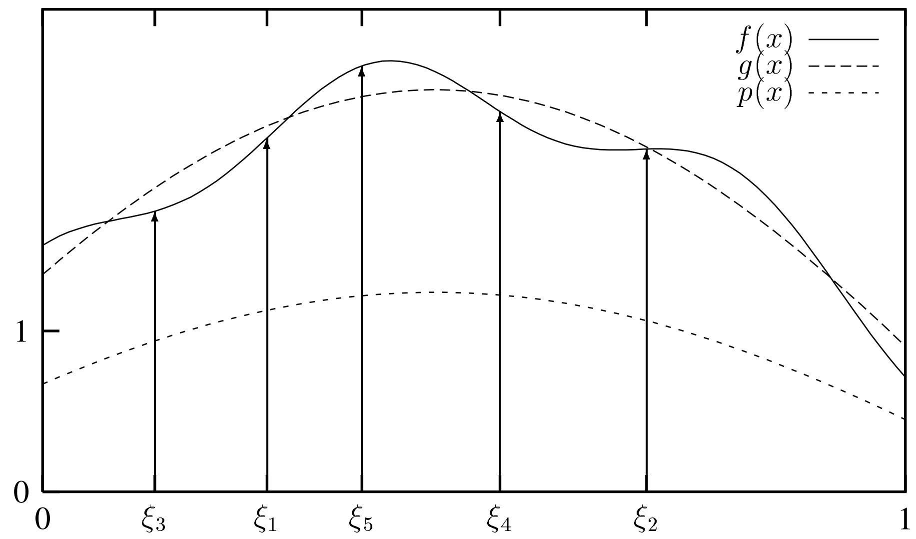


Figure 3.6: Importance sampling strives to select more samples in regions of the integrand $f(x)$ where the function values are high, by sampling according to an appropriate probability density function $p(x)$ and dividing the estimator by $p(x)$. The probability density function is the normalised version of an approximation $g(x)$ of the integrand.

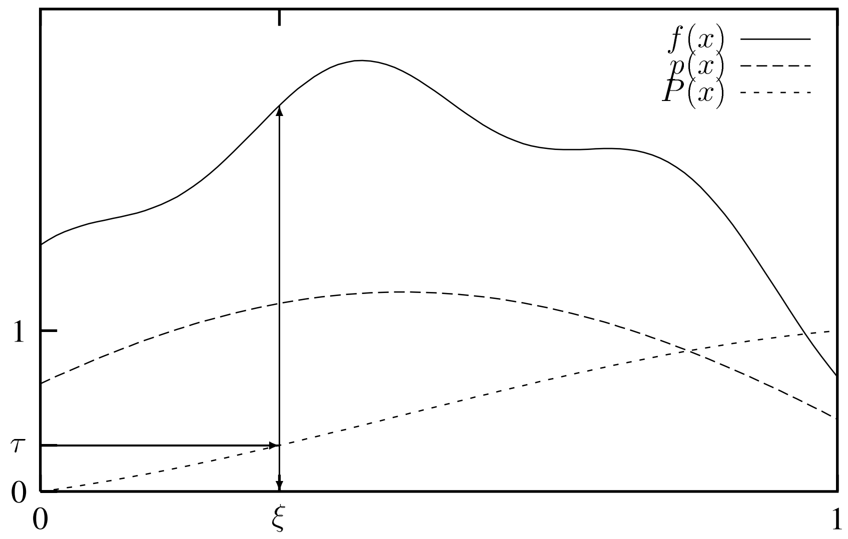


Figure 3.7: Sampling according to a PDF $p(x)$ is usually done by taking a uniform sample τ and transforming it to ξ with the inverse of the probability distribution function $P(x)$: $\xi = P^{-1}(\tau)$. The non-uniform sample ξ can then be used in an estimator for the integral of the function $f(x)$.

influence on the result. In these cases it is more effective to take more samples in these regions by sampling according to a *probability density function* (PDF) $p(x)$ that is of a similar shape as the integrand, rather than uniformly (Fig. 3.6). The estimator then has to be adapted accordingly so as not to introduce a bias. The integral can be rewritten as follows:

$$\begin{aligned} I &= \int_0^1 f(x)dx \\ &= \int_0^1 \frac{f(x)}{p(x)}p(x)dx. \end{aligned}$$

If a random variable ξ is sampled according to the PDF $p(x)$ over $[0,1]$ the primary estimator is

$$\langle I \rangle_{imp} = \frac{f(\xi)}{p(\xi)}. \tag{3.9}$$

The expected value of the estimator still is the actual value of the integral. The variance now becomes

$$\begin{aligned} \sigma^2_{imp} &= \int_0^1 \left[\frac{f(x)}{p(x)}\right]^2 p(x)dx - I^2 \\ &= \int_0^1 \frac{f^2(x)}{p(x)}dx - I^2. \end{aligned} \tag{3.10}$$

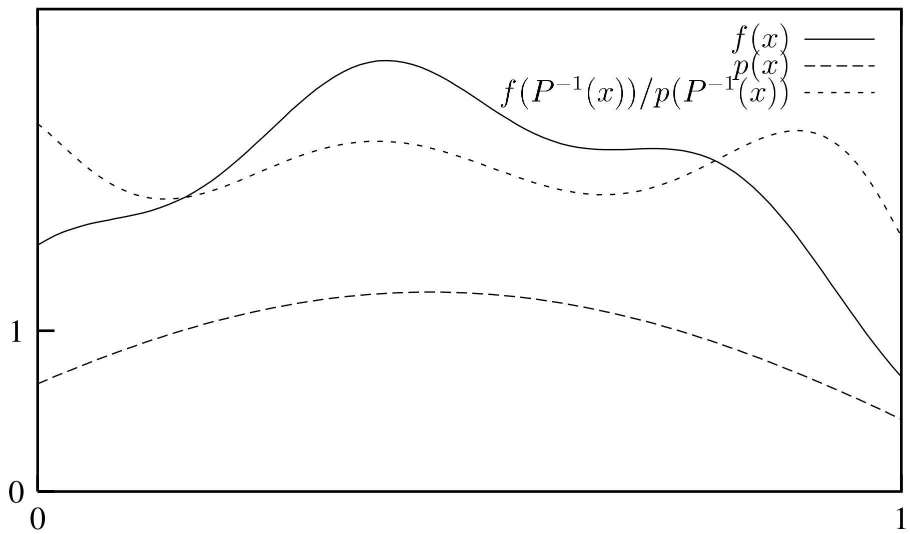


Figure 3.8: Importance sampling according to a PDF $p(x)$ to compute the integral of a function $f(x)$ is equivalent to computing the integral of the transformed function $f(P^{-1}(x))/p(P^{-1}(x))$ using uniform sampling. If the PDF is chosen appropriately the transformed function is more constant and therefore easier to integrate. The value of the integral remains the same, but the variance is reduced.

Any function $p(x)$ over the integration domain that meets the following requirements can be used as a probability density function:

- $p(x) > 0$ for each $x \in [0, 1]$ for which $f(x) \neq 0$,
- $\int_0^1 p(x)dx = 1$,
- It is possible to take a sample x such that the probability density of selecting it is $p(x)$. More specifically, it has to be possible to compute the inverse $P^{-1}(x)$ of the probability distribution function $P(x)$. The *probability distribution function* $P(x)$ is the cumulative function of the probability density function $p(x)$:

$$P(x) = \int_0^x p(t)dt. \tag{3.11}$$

For non-negative functions the PDF should ideally be proportional to the integrand: $p(x) = f(x)/I$, in which case the variance would be 0. Unfortunately the normalisation factor is precisely the integral which we are actually trying to compute, so this is not a practical option. However, any other function that has a similar shape as the integrand and that can be normalised, integrated and then inverted may yield a PDF that reduces the variance of the sampling process. If the integrand can also be negative the PDF should be proportional to the absolute value of the integrand, yielding a minimal but now non-zero variance. If the integrand contains a singularity it is important to include it in the PDF. If this is not possible the variance will in general be very high and possibly even infinite, because the singularity is then squared in the expression for the variance.

Selecting a sample ξ according to $p(x)$ is usually performed by taking a uniform random sample τ over $[0, 1]$ and then transforming it: $\xi = P^{-1}(\tau)$ (Fig. 3.7). Estimator (3.9) is therefore equivalent to the following estimator:

$$\langle I \rangle_{imp} = \frac{f(P^{-1}(\tau))}{p(P^{-1}(\tau))}. \tag{3.12}$$

This observation leads to the alternative interpretation of importance sampling as a transformation of the original integral:

$$\begin{aligned}
I &= \int_0^1 f(x)dx \\
&= \int_0^1 f(P^{-1}(t))\frac{dx}{dt}dt \\
&= \int_0^1 f(P^{-1}(t))\frac{dP^{-1}(t)}{dt}dt \\
&= \int_0^1 \frac{f(P^{-1}(t))}{p(P^{-1}(t))}dt.
\end{aligned}$$

Uniformly sampling the latter integral is equivalent to sampling the original integral according to $p(x)$. Figure 3.8 illustrates this idea. The variance reduction results from the transformed integral being more constant and therefore easier to integrate. Note that importance sampling can therefore easily be combined with stratified sampling. It is just uniform sampling after a transformation and the uniform samples τ can easily be stratified.

Using PDFs in higher dimensions is somewhat more cumbersome. For each dimension a sample has to be taken so as to obtain the desired multi-dimensional distribution. Consider for instance the two-dimensional PDF $p(x_1, x_2)$ over $[0,1] \times [0,1]$. In order to sample a pair (ξ_1, ξ_2) according to this distribution one can first sample ξ_1 according to the following one-dimensional distribution:

$$p_1(x_1) = \int_0^1 p(x_1, x_2) dx_2.$$

Once ξ_1 is known ξ_2 is sampled according to the distribution

$$p_2(x_2|x_1 = \xi_1) = \frac{p(\xi_1, x_2)}{\int_0^1 p(\xi_1, x_2) dx_2}.$$

The resulting distribution matches the desired distribution because its PDF is the product of the one-dimensional PDFs: $p(x_1, x_2) = p(x_1)p(x_2|x_1)$. Shirley [109] presents a few transformations for two-dimensional PDFs that are common in computer graphics.

Importance sampling can in general be achieved by means of various alternative transformations. Especially in higher dimensions however the strata in the original parameter space may be heavily distorted in the transformed parameter space. The coherence of the integrand over the strata may decrease as a result, affecting the efficacy of stratified sampling. This presents an additional concern when selecting a suitable transformation. Shirley [108] and Kolb *et al.* [61] for instance suggest techniques that map uniformly distributed samples in a square onto uniformly distributed samples on a disc. They distort the shape of the strata as little as possible and they are therefore especially suited for stratified sampling. Another interesting example is the transformation presented by Arvo [4] that maps uniformly distributed samples in a square onto uniformly distributed spatial angles inside a spherical triangle.

The main challenge with importance sampling lies in finding a function that approximates the integrand and that can be used as a PDF after normalisation. It requires knowledge about the integrand and therefore has to be studied specifically for each problem at hand. Most often a single factor of the integrand, one that determines its shape and is practically tractable, is split off and used as a PDF. In Chapter 4 we will investigate the alternatives for the integral equations describing the global illumination problem.

3.5 Combining Estimators

Sometimes it is hard to construct a single PDF that follows the shape of the entire integrand. It may be easier to construct several PDFs each of which approximates some specific part of the integrand. A common instance is an integrand that is a product of two factors. Often each of the factors on their own can be transformed into a PDF, but not the product. Yet both factors may predominantly determine the shape of the integrand in some regions. There are a few alternatives: picking a single PDF, combining the PDFs and selecting a single sample, or selecting one sample per PDF and combining the estimators.

In [113] Shirley *et al.* address the typical problem in physically-based rendering whether to sample according to the BRDF or according to the light sources when computing the direct illumination of a point on a surface. They suggest to identify the different factors as having a low, a high or an unknown variation. A single sample is then sampled according to the factor with the highest known variation, if the importance sampling is worth the additional computational cost. When sampling the light sources they suggest to make a linear combination of the PDFs corresponding to the individual luminaires. Again a single sample is taken according to the resulting PDF. Unfortunately many choices in these techniques are intuitive and one has to rely on rules of thumb to select acceptable parameters.

In [125, 126] Veach and Guibas present a novel approach in the contexts of direct illumination computations and of bidirectional path tracing. Although the results seem quite fundamental we have not found them in the general Monte Carlo literature. As it has proven to be very useful for the algorithms discussed in Chapter 4 we will give it some more attention. We will also present some minor improvements.

If one has several primary estimators, all computed with different PDFs, then it is not obvious how to combine them into a single estimate. Simply averaging them will not be optimal in general, as some estimators may be more reliable than others. The different estimators may even have different qualities in different regions of the integration domain. Veach and Guibas propose to make a weighted average of the estimators where the weights depend on the positions of the samples. We will derive the condition for the result to be unbiased for a slightly simplified case with a single sample per PDF. Suppose n random variables $\xi_1, \xi_2, \ldots, \xi_n$ are sampled over $[0,1]$ according to the respective PDFs $p_1(x), p_2(x), \ldots, p_n(x)$. The weighted sum of primary estimators of integral (3.1) can then be written as

$$\langle I \rangle_{combine} = \sum_{i=1}^{n} w_i(\xi_i) \frac{f(\xi_i)}{p_i(\xi_i)}, \tag{3.13}$$

where $w_i(x_i)$ are weight functions. The estimator should remain unbiased:

$$\begin{aligned} E(\langle I \rangle_{combine}) &= \sum_{i=1}^{n} \int_0^1 \left[w_i(x_i) \frac{f(x_i)}{p_i(x_i)} \right] p_i(x_i) dx_i \\ &= \int_0^1 \left[\sum_{i=1}^{n} w_i(x) \right] f(x) dx \\ &\equiv \int_0^1 f(x) dx. \end{aligned}$$

This condition is satisfied for all possible integrands $f(x)$ if and only if

$$\sum_{i=1}^{n} w_i(x) \equiv 1. \tag{3.14}$$

The variance of the combined estimators is the sum of the variances of the individual terms as they are stochastically uncorrelated:

$$\begin{aligned}\sigma^2_{combine} &= \sum_{i=1}^{n}\left[\int_0^1\left[w_i(x_i)\frac{f(x_i)}{p_i(x_i)}\right]^2 p_i(x_i)dx_i - \left[\int_0^1 w_i(x_i)\frac{f(x_i)}{p_i(x_i)}p_i(x_i)dx_i\right]^2\right] \quad (3.15)\\ &= \int_0^1\left[\sum_{i=1}^{n}\frac{w_i^2}{p_i(x)}\right]f(x)dx - \sum_{i=1}^{n}\left[\int_0^1 w_i(x)f(x)dx\right]^2.\end{aligned}$$

The choice of the weight functions is free as long as they satisfy condition (3.14). Some alternatives are:

- $w_i(x) = \frac{1}{n}$, yielding a simple averaging of the n estimators:

$$\langle I\rangle_{average} = \frac{1}{n}\sum_{i=1}^{n}\frac{f(\xi_i)}{p_i(\xi_i)}. \quad (3.16)$$

- $w_i(x) = 1$ for i equalling a fixed index k, and $w_i(x) = 0$ for all other i. This results in the selection of a single sample and its estimator k from the set of estimators:

$$\langle I\rangle_{select} = \frac{f(\xi_k)}{p_k(\xi_k)}. \quad (3.17)$$

  Of course it is mostly of theoretical interest; it will not be used in practice as it wastes all samples but one. Still, this choice can be optimal if its estimator is perfect while the others are not. The variance obviously equals the variance of the selected estimator.

- $w_i(x) = 1$ if $p_i(x) > p_j(x)$ for all $j \neq i$, and 0 otherwise yields the so-called *maximum heuristic*. The resulting estimator is

$$\langle I\rangle_{maximum} = \sum_{i=1}^{n}(p_i(\xi_i) > p_j(\xi_i), \forall j \neq i)?\frac{f(\xi_i)}{p_i(\xi_i)} : 0. \quad (3.18)$$

- $w_i(x) = \frac{p_i(x)}{\sum_{j=1}^{n} p_j(x)}$, the so-called *balance heuristic*, yields

$$\langle I\rangle_{balance} = \sum_{i=1}^{n}\frac{f(\xi_i)}{\sum_{j=1}^{n} p_j(\xi_i)}. \quad (3.19)$$

  The variance equals

$$\sigma^2_{balance} = \int_0^1 \frac{1}{\sum_{i=1}^{n} p_i(x)} f^2(x)dx - \sum_{i=1}^{n}\left[\int_0^1 \frac{p_i(x)}{\sum_{j=1}^{n} p_j(x)} f(x)dx\right]^2. \quad (3.20)$$

  This choice is optimal in the sense that it minimises the sum of the second-order moments of the individual terms of the estimator, i.e. the first term of the expression for the variance (3.15). For a more general case it is proven in [126] that the variance obtained using other weight functions can only improve the balance heuristic by a limited amount:

$$\sigma^2_{combine} \geq \sigma^2_{balance} - \left(1 - \frac{1}{n}\right) I^2.$$

- $w_i(x) = \frac{p_i^\beta(x)}{\sum_{j=1}^n p_j^\beta(x)}$ is a generalisation of both the maximum heuristic and the balance heuristic, called the *power heuristic*. The estimator becomes

  $$\langle I \rangle_{power} = \sum_{i=1}^{n} \frac{p_i^{\beta-1}(\xi_i)}{\sum_{j=1}^n p_j^\beta(\xi_i)} f(\xi_i). \tag{3.21}$$

  The exponent β can be any number; $\beta = 1$ yields the balance heuristic and $\beta = \infty$ yields the maximum heuristic. The strategy is reported to be more effective than other heuristics if the PDFs fit the integrand particularly well in regions where they are relatively large. The exponent emphasizes the effects of the weights: they become even larger in regions where they were relatively large already and smaller in regions where they were relatively small.

As with importance sampling and other variance reduction techniques we can interpret the combination of estimators as a transformation of the original integral. Formally:

$$\begin{aligned} I &= \int_0^1 f(x)dx \\ &= \sum_{i=1}^{n} \int_0^1 w_i(x) f(x) dx, \end{aligned}$$

on the same condition as above that $\sum_{i=1}^n w_i(x) \equiv 1$. From this point of view combining different estimators of the same integrand with weight functions is actually equivalent to partitioning the integrand with the weight functions and summing the estimators. Sampling each of the n integrals with its corresponding PDF and summing the estimators yields the combined estimator (3.13). The importance sampling corresponds to the following additional transformations:

$$I = \sum_{i=1}^{n} \int_0^1 \frac{w_i(P_i^{-1}(t))}{p_i(P_i^{-1}(t))} f(P_i^{-1}(t)) dt.$$

Figures 3.9 to 3.14 illustrate the partitioning on an example. The integrand is the product of two strongly peaked functions. The second and third graphs show how the integrand is partitioned: into two equal parts (Fig. 3.9), into the function itself and 0 (Fig. 3.10), into 0 and the function itself (Fig. 3.11), into parts corresponding to the maximum heuristic (Fig. 3.12), into parts corresponding to the balance heuristic (Fig. 3.13) and into parts corresponding to the power heuristic (Fig. 3.14). The respective parts are sampled according to the PDFs corresponding to the two original peaked functions. The fourth and fifth graphs of each figure show the functions after the transformations that are implied by the importance sampling. Both transformed integrals are estimated with a single sample and the estimates are summed. This is equivalent to taking a single sample from the two-dimensional function at the bottom of each figure. The variance is also the sum of the variances as the samples are uncorrelated. The more constant both transformed functions are the lower the variance will be. In this example the maximum heuristic yields functions that are much more constant than the original function. In some subintervals the functions drop to 0 though, thereby increasing the variance substantially. The effect is similar to *hit-or-miss sampling* where the integrand is estimated by assigning 1 or a constant to a sample if it lies within the volume to be integrated and 0 otherwise. This technique is very inefficient in general and will also have a negative influence on the maximum heuristic. The balance heuristic yields visibly much more constant functions and therefore better results. The power heuristic with $\beta = 2$ lies somewhere in between both heuristics and has little effect in this case.

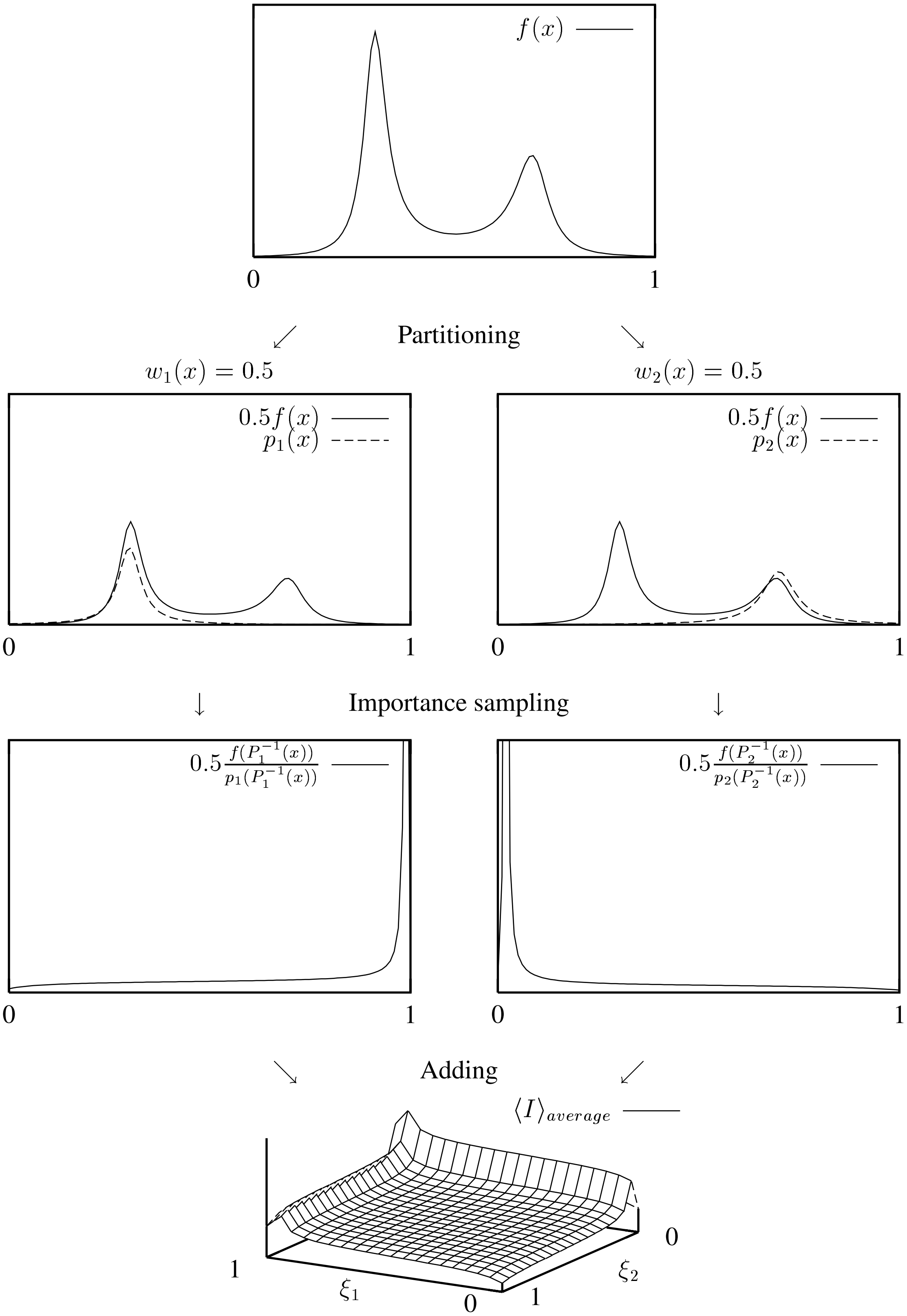


Figure 3.9: An example integrand $f(x)$ that is the product of two strongly peaked functions. The integrand is partitioned by halving it. Both parts are sampled according to their own PDFs. This is equivalent to the transformations shown. Finally the estimates are summed.

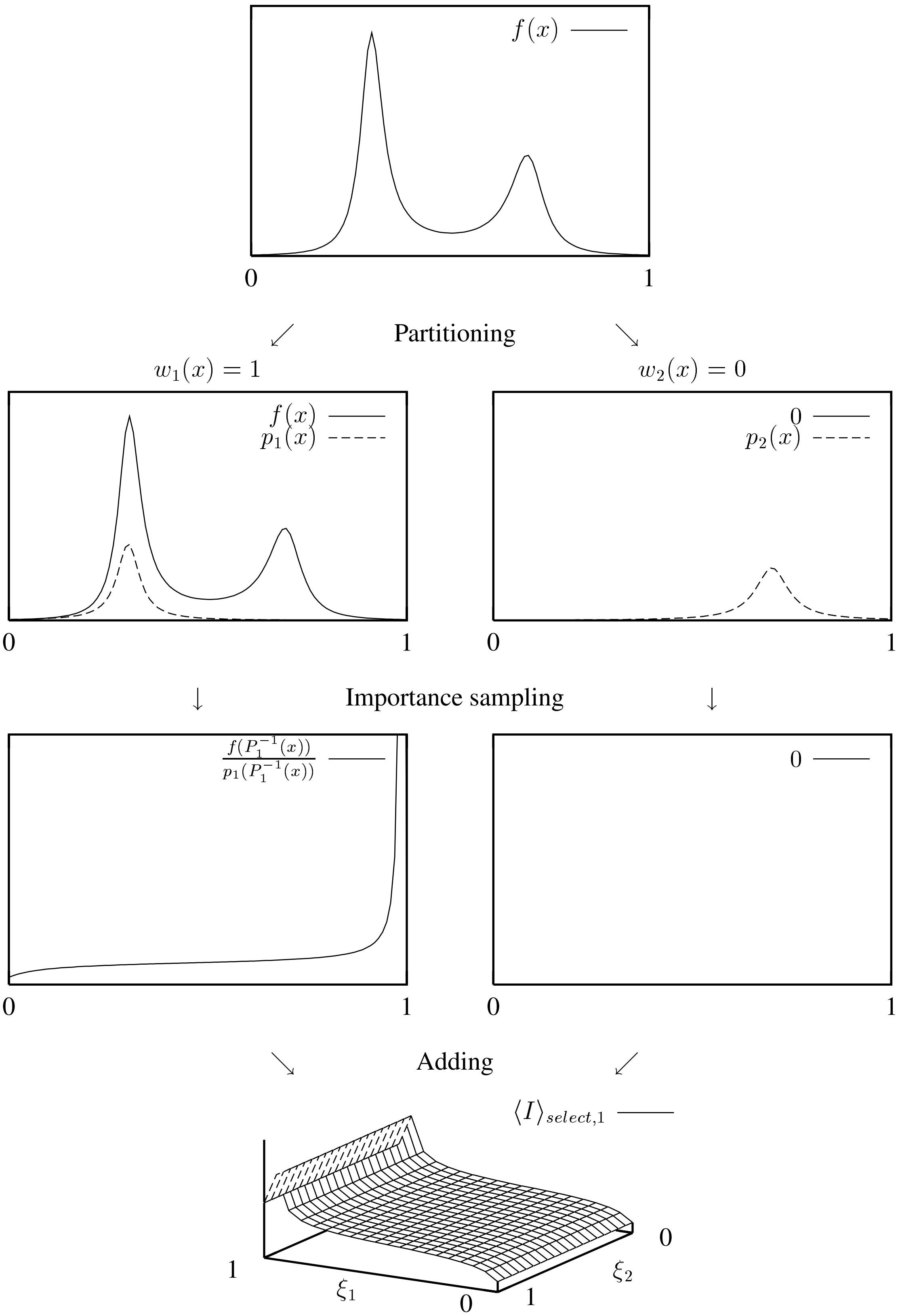


Figure 3.10: The same integrand is now partitioned into itself and 0. Both parts are sampled according to their own PDFs and the estimates are summed. The weight functions thus always select the first estimator.

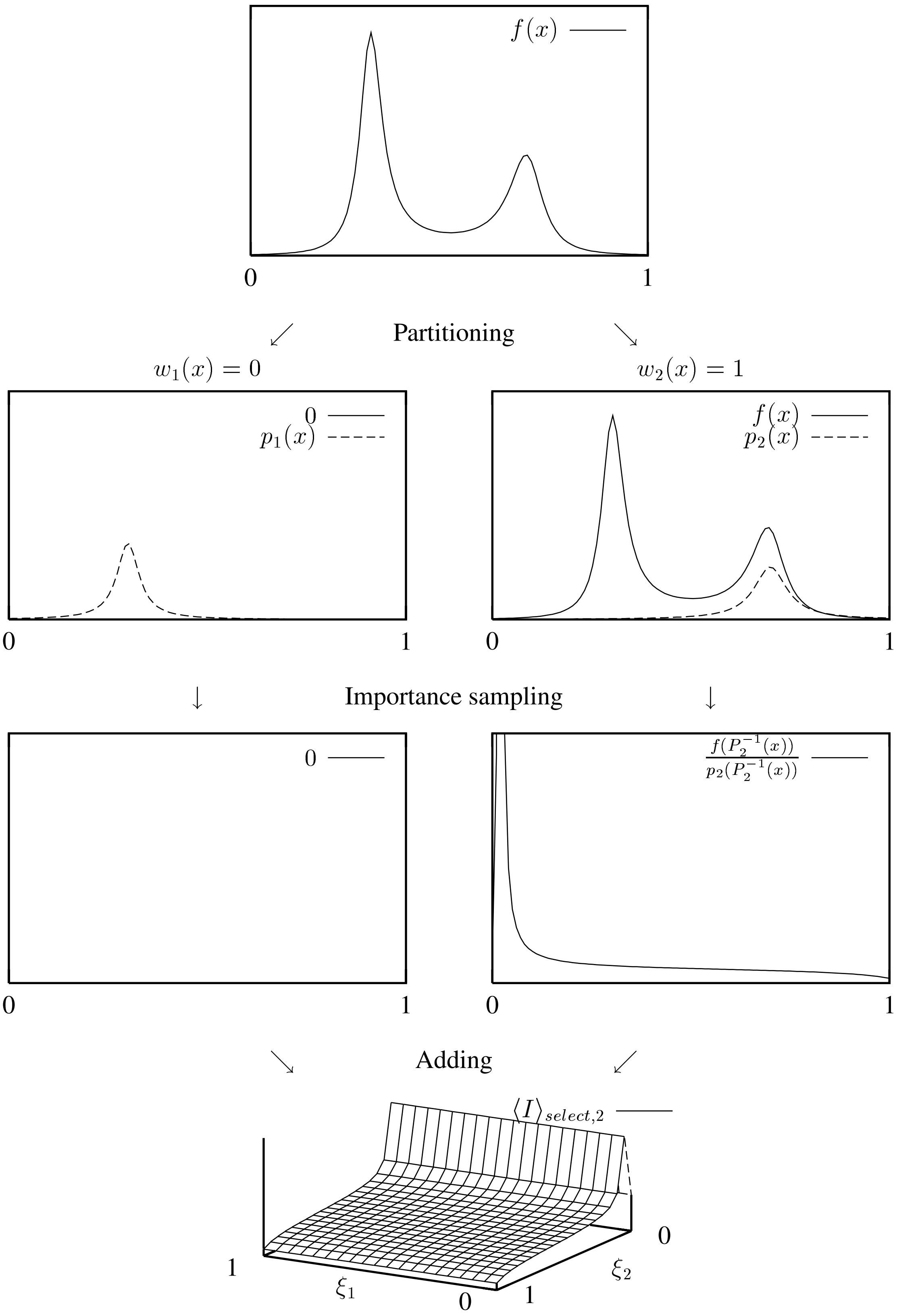


Figure 3.11: The same integrand is now partitioned into 0 and itself. Both parts are sampled according to their own PDFs and the estimates are summed. The weight functions thus always select the second estimator.

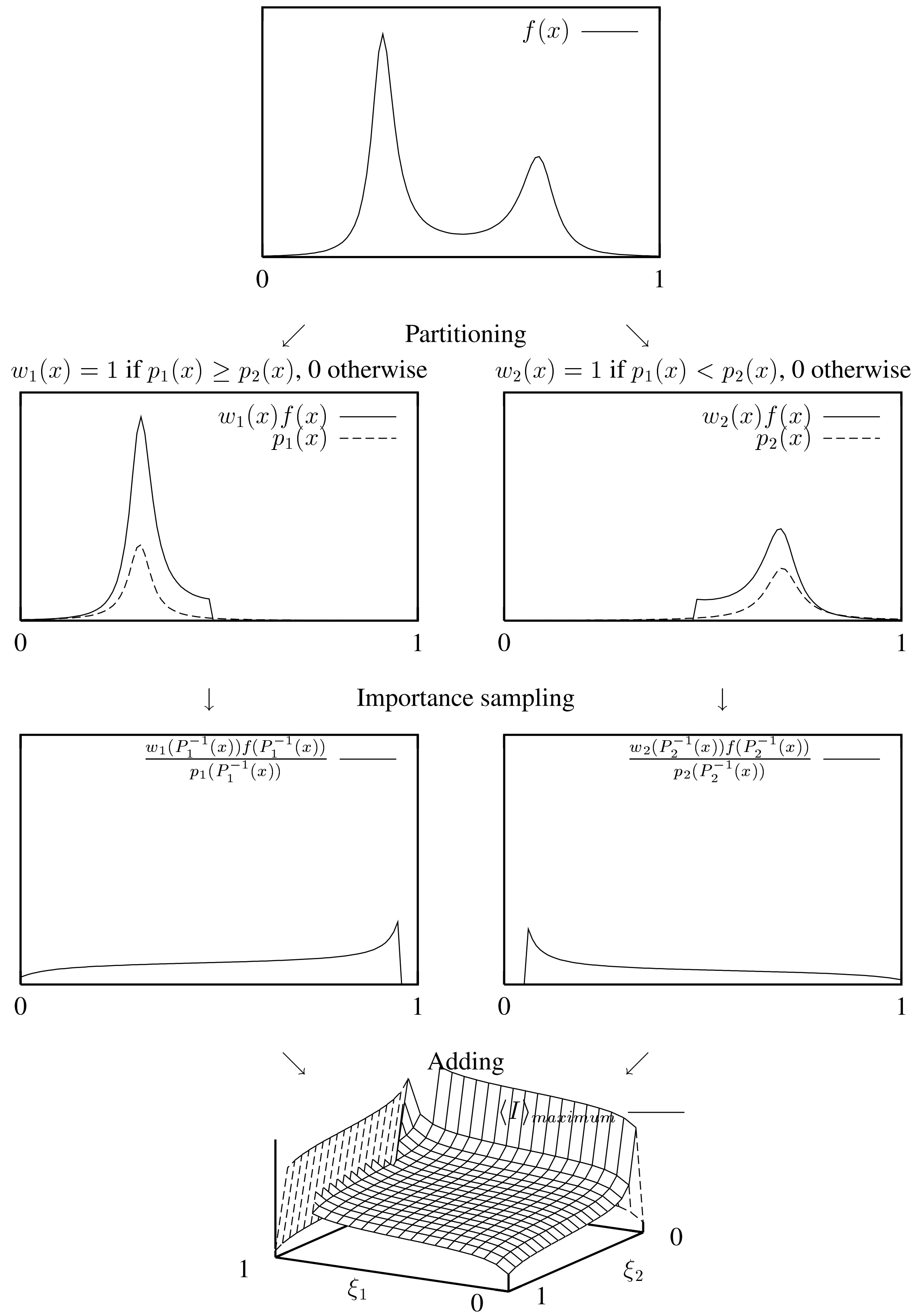


Figure 3.12: The same integrand is now partitioned on the basis of the maximum heuristic. The weight functions are only 1 at points where the corresponding PDFs have the largest value. Both resulting parts are sampled and the estimates are summed.

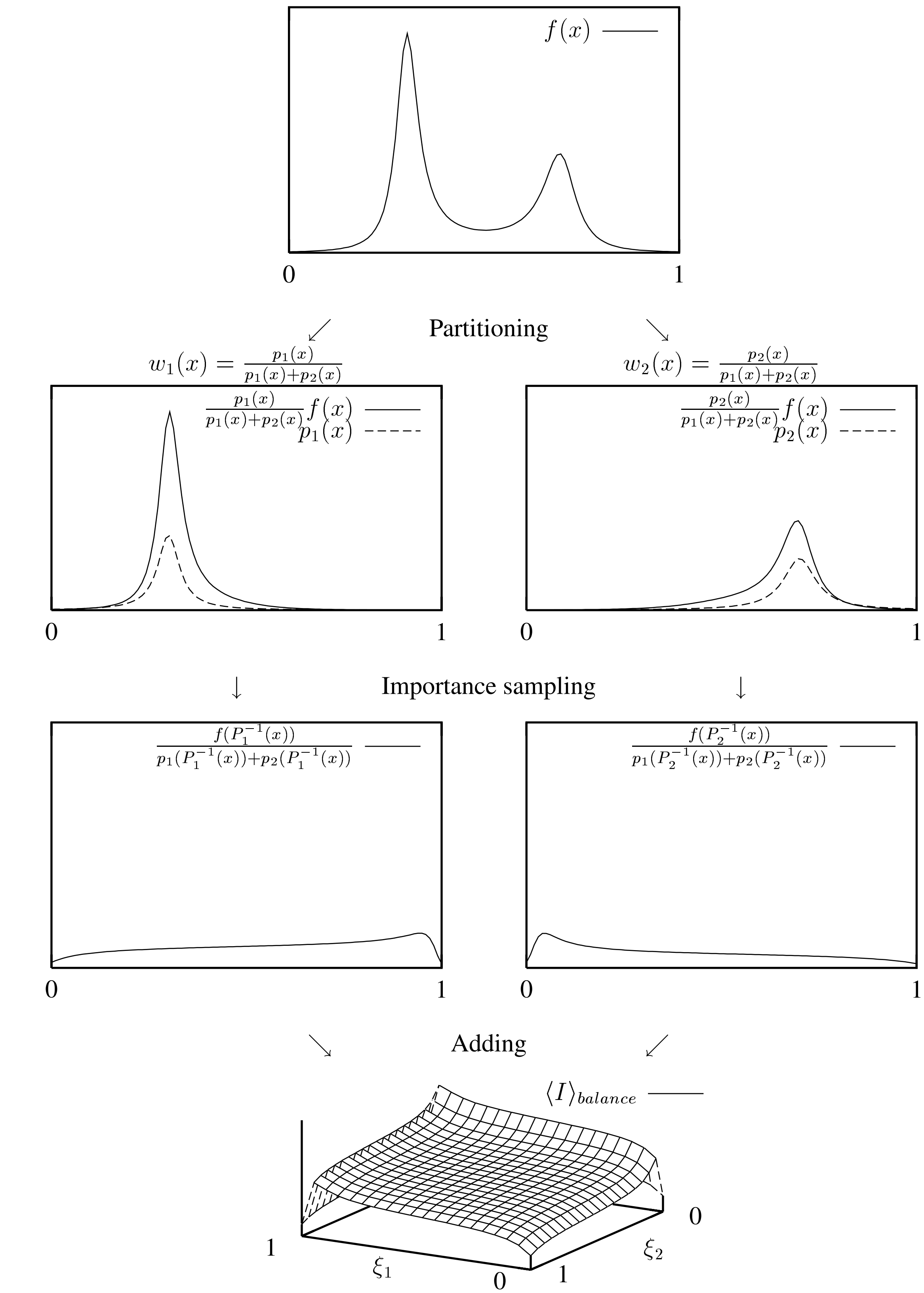


Figure 3.13: The same integrand is now partitioned on the basis of the balance heuristic. The weight functions are large at points where the corresponding PDFs are relatively large. Both resulting parts are sampled and the estimates are summed.

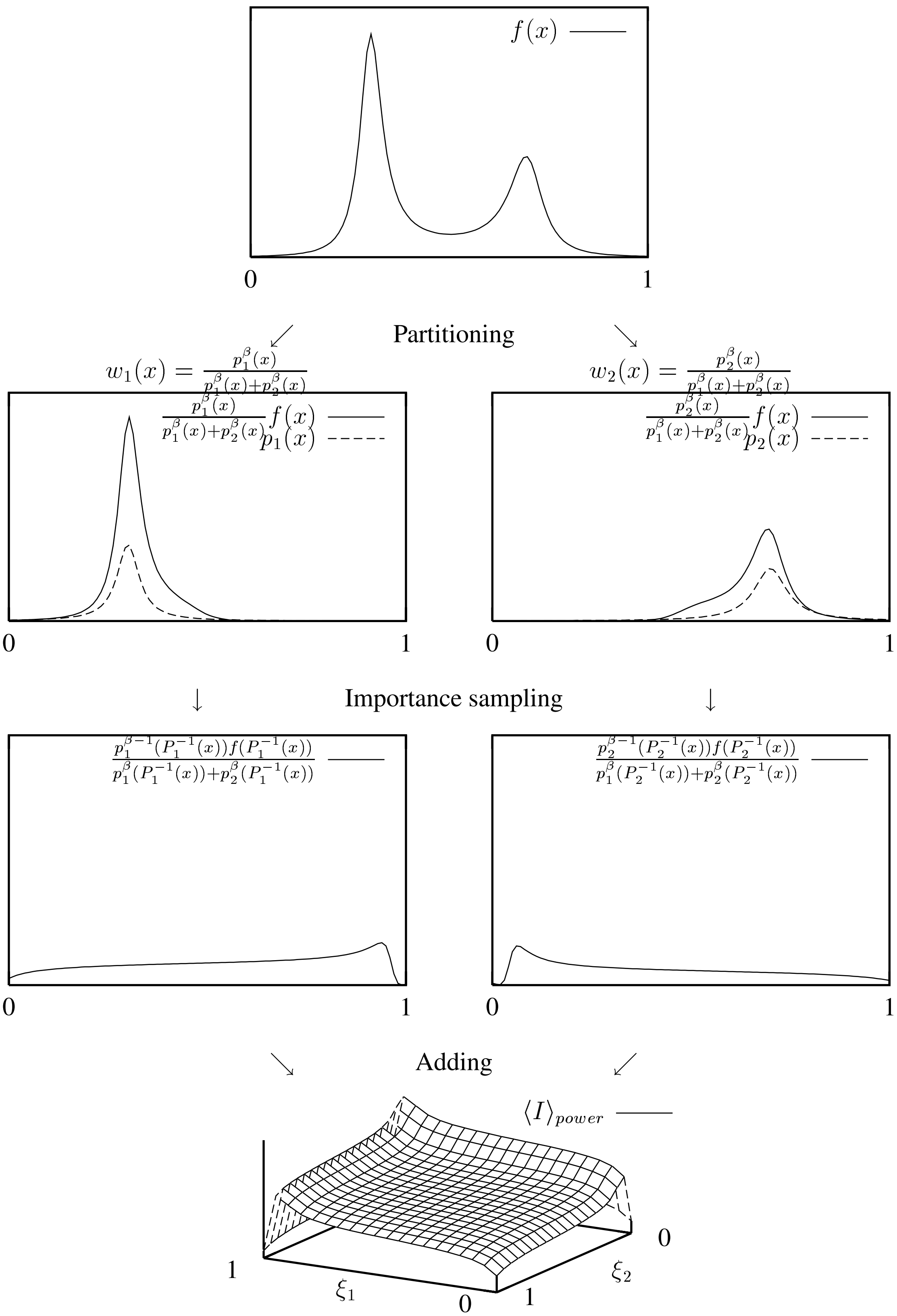


Figure 3.14: The same integrand is now partitioned on the basis of the power heuristic with $\beta = 2$. The weight functions are large at points where the corresponding PDFs are relatively large, slightly more so than with the balance heuristic.

An interesting question is whether these heuristics can be improved without any further specific knowledge about the integrand. For some heuristics this appears to be possible. In his discussion of stochastic quadrature rules Ermakow points out that estimators that are not symmetrical in $\xi_1, \ldots, \xi_n$ can always be replaced by symmetrical estimators with a lower variance [30, pp. 124-125]. A multi-dimensional function $F(x_1, x_2, \ldots, x_n)$ is called symmetrical in this context if it remains invariant when its variables are permuted arbitrarily. Suppose one has an estimator that is a function of n stochastic variables:

$$\langle I\rangle_{asymmetrical} = F(\xi_1, \ldots, \xi_n),$$

where the stochastic variables $\xi_1, \ldots, \xi_n$ are sampled over $[0,1]^n$ according to the n-dimensional PDF $p(x_1, \ldots, x_n)$. If the estimator is not symmetrical a symmetrical estimator and a symmetrical PDF can be constructed as follows:

$$\langle I\rangle_{symmetrical} = \frac{\sum_{P(k_1,\ldots,k_n)} F(\xi_{k_1}, \ldots, \xi_{k_n}) p(\xi_{k_1}, \ldots, \xi_{k_n})}{\sum_{P(k_1,\ldots,k_n)} p(\xi_{k_1}, \ldots, \xi_{k_n})},$$

$$p_{symmetrical}(x_1, \ldots, x_n) = \frac{1}{n!} \sum_{P(k_1,\ldots,k_n)} p(x_{k_1}, \ldots, x_{k_n}).$$

where the summation over $P(k_1, \ldots, k_n)$ denotes a summation over all possible permutations of $1, \ldots, n$ assigned to $k_1, \ldots, k_n$. One can easily show that the expected value of the estimator remains unchanged. The symmetrical PDF only brings about a random permutation of the stochastic variables which has no effect, as the estimator is now symmetrical. The original PDF is therefore equally effective and simpler to use in practice. The variance of the symmetrical estimator however can be shown to be always lower than the original variance. For each $(x_1, \ldots, x_n) \in [0,1]^n$:

$$\begin{aligned}
&\left[\frac{\sum_{P(k_1,\ldots,k_n)} F(x_{k_1}, \ldots, x_{k_n}) p(x_{k_1}, \ldots, x_{k_n})}{\sum_{P(k_1,\ldots,k_n)} p(x_{k_1}, \ldots, x_{k_n})}\right]^2 \frac{1}{n!} \sum_{P(k_1,\ldots,k_n)} p(x_{k_1}, \ldots, x_{k_n}) \\
&= \frac{1}{n!} \frac{\left[\sum_{P(k_1,\ldots,k_n)} F(x_{k_1}, \ldots, x_{k_n}) p(x_{k_1}, \ldots, x_{k_n})\right]^2}{\sum_{P(k_1,\ldots,k_n)} p(x_{k_1}, \ldots, x_{k_n})} \\
&\leq \frac{1}{n!} \frac{\left[\sum_{P(k_1,\ldots,k_n)} F^2(x_{k_1}, \ldots, x_{k_n}) p(x_{k_1}, \ldots, x_{k_n})\right]\left[\sum_{P(k_1,\ldots,k_n)} p(x_{k_1}, \ldots, x_{k_n})\right]}{\sum_{P(k_1,\ldots,k_n)} p(x_{k_1}, \ldots, x_{k_n})} \\
&= \frac{1}{n!} \sum_{P(k_1,\ldots,k_n)} F^2(x_{k_1}, \ldots, x_{k_n}) p(x_{k_1}, \ldots, x_{k_n}),
\end{aligned}$$

where the inequality holds because $[\sum a_i b_i]^2 \leq \sum a_i^2 b_i \sum b_i$ if all $b_i \geq 0$. Integrating both sides of this inequality over the interval $[0,1]$ yields the second order moments of the complete symmetrical and asymmetrical estimators respectively. Therefore always:

$$\sigma^2_{symmetrical} \leq \sigma^2_{asymmetrical}. \tag{3.22}$$

We can now apply these findings to the combined estimator (3.13). The expression shows that the estimator is in general not symmetrical in $\xi_1, \ldots, \xi_n$ if the factors $w_i(x)/p_i(x)$ differ. The n-dimensional PDF is the product of the different PDFs:

$$p(x_1, \ldots, x_n) = p_1(x_1) \ldots p_n(x_n).$$

Beside the artificial estimator that selects a single estimator from the ones available, the combined estimator is not symmetrical for the maximum heuristic and for the power heuristic in general (with $\beta \neq 1$). They can therefore be improved as follows:

- For the maximum heuristic the estimator becomes

$$\langle I \rangle_{maximum,sym} = \frac{\sum_{P(k_1,\ldots,k_n)} \left[\sum_{i=1}^{n} (p_i(\xi_{k_i}) > p_j(\xi_{k_i}), \forall j \neq i) ? \frac{f(\xi_{k_i})}{p_i(\xi_{k_i})} : 0\right] p_1(\xi_{k_1}) \ldots p_n(\xi_{k_n})}{\sum_{P(k_1,\ldots,k_n)} p_1(\xi_{k_1}) \ldots p_n(\xi_{k_n})}. \tag{3.23}$$

  Figure 3.15 shows the symmetrical version of the maximum heuristic for the double-peaked example function. Equation (3.22) guarantees that it is an improvement over the ordinary maximum heuristic shown in the bottom graph of Fig. 3.12.

- For the power heuristic the estimator becomes

$$\langle I \rangle_{power,sym} = \frac{\sum_{P(k_1,\ldots,k_n)} \left[\sum_{i=1}^{n} \frac{p_i^{\beta-1}(\xi_{k_i})}{\sum_{j=1}^{n} p_j^{\beta}(\xi_{k_i})} f(\xi_{k_i})\right] p_1(\xi_{k_1}) \ldots p_n(\xi_{k_n})}{\sum_{P(k_1,\ldots,k_n)} p_1(\xi_{k_1}) \ldots p_n(\xi_{k_n})}. \tag{3.24}$$

  Figure 3.16 shows the symmetrical version of the power heuristic for the example function. Equation (3.22) again guarantees that it is an improvement over the ordinary power heuristic shown in the bottom graph of Fig. 3.14. In this case the visible difference is small.

As mentioned earlier the stochastic variables can safely be sampled using the original non-symmetrical PDF. The resulting estimators after transformation that are shown in the figures are therefore not necessarily symmetrical anymore.

The balance heuristic cannot be improved any further using this technique as its estimator is already symmetrical. Future research should point out if there are other ways to enhance it.

In general, the variance of a combination of n estimators can be better or worse than the variance of n samples taken according to any of the individual PDFs $p_i(x)$. It can also be better or worse than the variance of n samples taken according to any linear combination of the PDFs $\sum_i \alpha_i p_i(x)$. Veach and Guibas already mention that it is not possible to combine several bad estimators into a single good estimator [126]. Moreover, if one estimator is perfect or nearly perfect additional estimators will only increase the variance of the result. The main challenge when applying the technique to actual problems therefore lies in finding PDFs that at least have the potential to sample parts of the integrand efficiently.

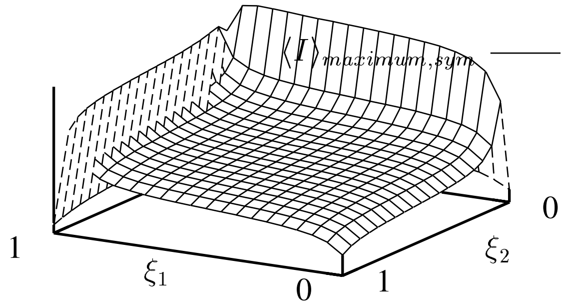


Figure 3.15: The symmetrical estimator derived from the maximum heuristic shown in Fig. 3.12, after transformation on the basis of the original non-symmetrical PDF. The variance is guaranteed to be lower.

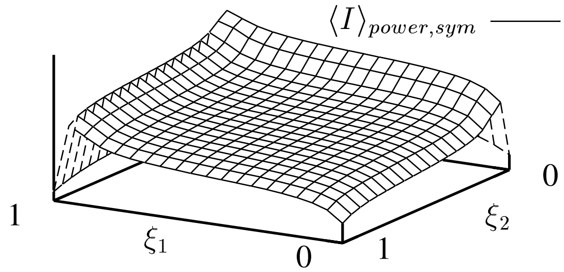


Figure 3.16: The symmetrical estimator derived from the power heuristic with $\beta = 2$ shown in Fig. 3.14, after transformation on the basis of the original non-symmetrical PDF. Again the variance is guaranteed to be lower.

3.6 Control Variates

The requirements for deriving a PDF from a function that approximates the integrand are sometimes difficult to meet. If one knows a function $g(x)$ that approximates the integrand $f(x)$ and that can be integrated analytically then it can be used as a *control variate* (Fig. 3.17). Transforming the original integral (3.1) again yields

$$\begin{aligned} I &= \int_0^1 f(x)dx \\ &= \int_0^1 [f(x)-g(x)]dx + \int_0^1 g(x)dx \\ &= \int_0^1 [f(x)-g(x)]dx + J \\ &= \int_0^1 [f(x)-g(x)+J]dx. \end{aligned}$$

If a random variable ξ is sampled uniformly over $[0,1]$ the primary estimator for the latter expression becomes

$$\langle I \rangle_{con} = f(\xi) - g(\xi) + J. \tag{3.25}$$

The expected value of the estimator still is the actual value of the integral. The variance now becomes

$$\sigma_{con}^2 = \int_0^1 [f(x)-g(x)+J]^2 dx - I^2. \tag{3.26}$$

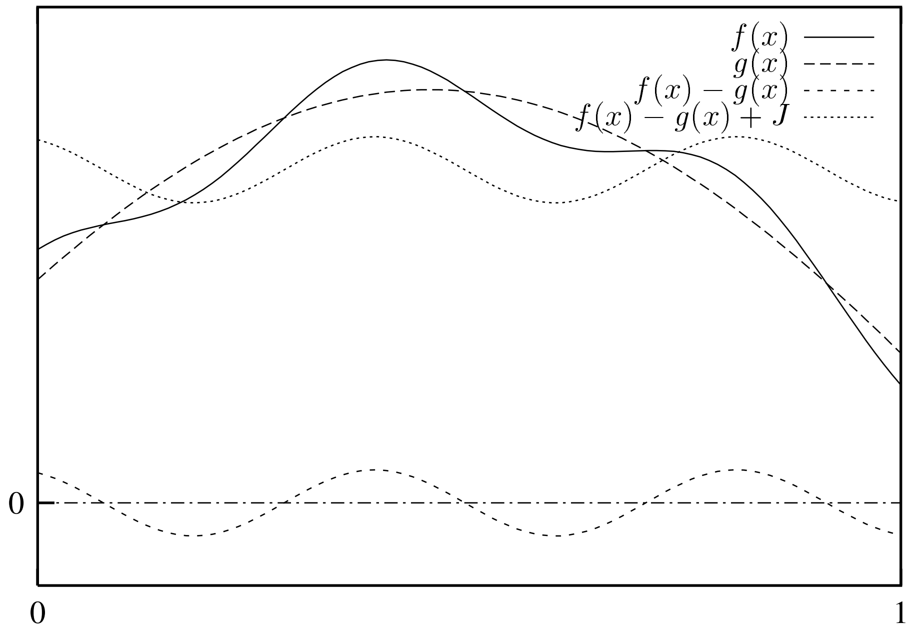


Figure 3.17: A function $g(x)$ that approximates $f(x)$ and that can be integrated analytically can serve as a control variate. It is more efficient to sample the difference between the functions and add the known integral value J than to sample $f(x)$ itself in order to compute an estimate. As with importance sampling the variance reduction results from the transformed function being more constant.

As with importance sampling the ideal case $g(x) = f(x)$ (and therefore $J = I$) would reduce the variance to 0 but would also require the knowledge of the exact value of the integral. Control variates that only approximate the integrand have a smoothing effect on it and thereby reduce the variance of the estimator.

Unlike importance sampling control variates do not require the cumulative function of the approximating function to be invertible. On the other hand the approximation has to be known exactly. With importance sampling it only has to be known to a constant factor since the function is normalised anyway. If a function is used for importance sampling it has no further effect as a control variate. The implicit transformation of the importance sampling would reduce the function to a constant. This constant function is useless as a control variate, as can be seen from the expressions for the estimator (3.25). Halton [39, 38] makes a brief comparison between the effectiveness of importance sampling and that of control variates. The intuitive conclusion is that functions that approximate the integrand with a constant relative error are more effective as a basis for importance sampling; functions that approximate the integrand with a small absolute error are more effective as control variates.

3.7 Monte Carlo Methods to Solve Integral Equations

Consider the following integral equation:

$$f(x) = g(x) + \int_0^1 K(x, y) f(y) dy, \tag{3.27}$$

where $f(x)$ is the unknown function, $g(x)$ is a known function and $K(x, y)$ is the kernel of the integral operator. The equation is known as a *Fredholm equation of the second kind*. If it cannot be solved analytically, which is often the case in practical applications, one can solve it numerically. Finite element methods are a first alternative to compute an approximation to the unknown function $f(x)$, as a sum of basis functions. In some cases it may not be feasible to reconstruct the entire function, because it is too complex to store for instance. In other cases it may not be necessary to know the entire function, if one is only interested in the function values at a few sample points. Then an alternative approach is to use a Monte Carlo method that computes $f(x)$ for a fixed x by evaluating the right hand side as if it were an ordinary integral. Because this in turn requires the evaluation of some $f(y)$ the process is repeated recursively. If all subsequent integrals are evaluated by each time taking a sample ξ_i over $[0, 1]$ according to the PDF $p_i(x)$ the resulting primary estimator looks as follows:

$$\begin{aligned} \langle f(x) \rangle_{recursive} &= g(x) + \frac{K(x, \xi_1)}{p_1(\xi_1)} \langle f(\xi_1) \rangle_{recursive} \\ &= g(x) + \frac{K(x, \xi_1)}{p_1(\xi_1)} \left[g(\xi_1) + \frac{K(\xi_1, \xi_2)}{p_2(\xi_2)} \langle f(\xi_2) \rangle_{recursive} \right] \\ &= g(x) + \frac{K(x, \xi_1)}{p_1(\xi_1)} g(\xi_1) + \frac{K(x, \xi_1)}{p_1(\xi_1)} \frac{K(\xi_0, \xi_2)}{p_2(\xi_2)} g(\xi_2) + \ldots \\ &= \sum_{i=0}^{\infty} \left[\prod_{j=1}^{i} \frac{K(\xi_{j-1}, \xi_j)}{p_j(\xi_j)} \right] g(\xi_i), \end{aligned} \tag{3.28}$$

where $\xi_0 = x$. The series of points $\xi_1, \xi_2, \ldots$ is a so-called *Markov chain* if the PDFs $p_i(x)$ for each point ξ_i only depend on the previous point ξ_{i-1}. It is also called a *random walk*, as one can visualise the process as a series of random hops from one point to the next one over the domain of the integral equation.

The integral equation can be written in short using an integral operator:

$$f = g + Tf.$$

The solution can then be written out as a *Neumann series*:

$$f = g + Tg + T^2 g + T^3 g + \ldots$$

The primary estimator (3.28) can be regarded as a Monte Carlo evaluation of this sum of integrals.

3.8 Russian roulette

The infinite number of terms in the estimator (3.28) presents a practical problem. The series has to be cut off at some point. If the Neumann series converges the terms will converge to 0. Truncating the series at some fixed point however is equivalent to consistently estimating one of the integrals in the recursion as 0. From a theoretical point of view this unavoidably introduces a bias into the result. Even from a practical point of view one is never sure that the neglected terms are not important compared to the computed terms. A theoretically more sound technique is *Russian roulette*. It does not cut off any of the subsequent integrals in a deterministic way, but each of them may be cut off in a probabilistic way. Dividing the estimator by the proper probability to compensate for the probabilistic truncation yields an unbiased estimator. The primary estimator for the simple integral (3.1) can be derived through a simple transformation (Fig. 3.18):

$$
\begin{aligned}
I &= \int_0^1 f(x)dx \\
&= \int_0^P \frac{1}{P} f(\frac{x'}{P})dx' \\
&= \int_0^1 \frac{1}{P} f(\frac{x'}{P}) u(\frac{x'}{P})dx',
\end{aligned}
$$

where:

- P = some arbitrary but fixed number in $(0, 1]$,
- $u(x)$ = the step function that is 1 for $x \leq 1$, and 0 for $x > 1$.

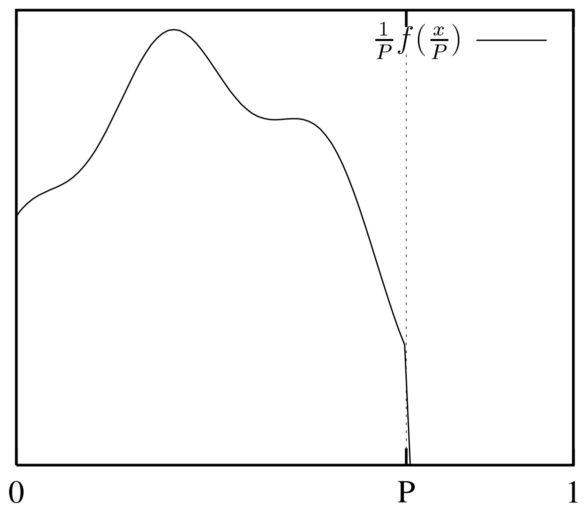


Figure 3.18: Russian roulette can be regarded as a transformation of the original integral. The integrand is scaled down by P on the x-axis and scaled up by $1/P$ on the y-axis. The value of the integral remains the same. The Monte Carlo estimator of the transformed function over the interval $[0, 1]$ therefore has the same expected value, albeit with a larger variance. The technique is mostly used to terminate an otherwise infinite recursion when solving an integral equation.

Straightforward Monte Carlo integration of the latter integral with a single uniform random sample ξ from $[0,1]$ yields the following primary estimator:

$$\langle I\rangle_{Russian} = \begin{cases} \frac{1}{P}f(\frac{\xi}{P}) & \text{if } \xi \leq P, \\ 0 & \text{otherwise.} \end{cases} \tag{3.29}$$

The estimator shows that P is actually the probability that the integrand $f(x)$ has to be evaluated. The probability that the estimator is just 0 is $1-P$. Russian roulette is therefore often implemented by choosing two random variables, the first of which is used to decide whether the estimate is 0 or not and the second of which is used to actually estimate the integral. One can show that the variance of the latter approach is larger once the technique is being applied in combination with stratified sampling (N-rooks sampling). Thus the above estimator is preferable.

The derivation shows that the estimator is unbiased. The variance unavoidably increases compared to the standard primary estimator (3.2) however:

$$\begin{aligned} \sigma^2_{Russian} &= \int_0^1 \left[\frac{1}{P}f(\frac{\xi}{P})u(\frac{\xi}{P})\right]^2 d\xi - I^2 \\ &= \int_0^P \frac{1}{P^2}f^2(\frac{\xi}{P})d\xi - I^2 \\ &= \frac{1}{P}\int_0^1 f^2(\xi)d\xi - I^2 \\ &> \sigma^2_{prim}. \end{aligned} \tag{3.30}$$

Russian Roulette Combined with Importance Sampling and Control Variates

We will now analyse the effects of combining Russian roulette, importance sampling and control variates, as the results will be of practical interest for the algorithms developed in the Chapter 4.

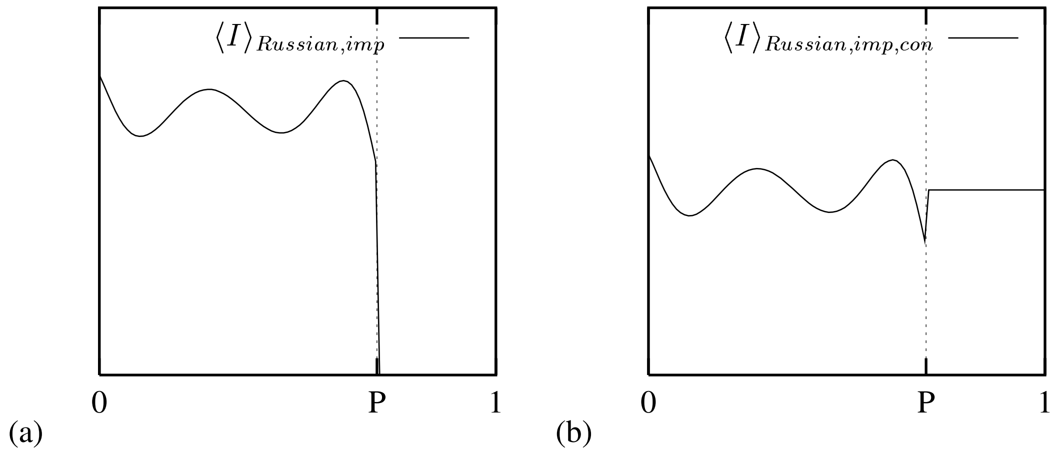


Figure 3.19: (a) Applying the same importance sampling as in Fig. 3.6 after the Russian roulette smoothes the scaled function in a similar way. (b) Using a function proportional to the PDF as a control variate normally does not have any additional effect. When applied before the Russian roulette however, it smoothes the integrand further.

As mentioned before a function that is used as a PDF for importance sampling cannot bring any further benefits as a control variate. If Russian roulette is also applied however, the control variate can reduce the variance further. Consider the function $g(x)$ that approximates the integrand $f(x)$, that can be integrated analytically ($J = \int_0^1 g(x)dx$) and that can be used to derive a PDF ($p(x) = g(x)/J$). If Russian roulette is applied, followed by importance sampling the estimator on the basis of a uniform sample τ over $[0,1]$ becomes a combination of Eq. (3.12) and Eq. (3.18):

$$\langle I \rangle_{Russian,imp} = \begin{cases} \frac{1}{P}\frac{f(P^{-1}(\frac{\tau}{P}))}{p(P^{-1}(\frac{\tau}{P}))} & \text{if } \tau \leq P, \\ 0 & \text{otherwise.} \end{cases} \tag{3.31}$$

Figure 3.19a shows the transformed integrand for the example. It has become more constant in the region where it differs from 0. The variance equals

$$\sigma^2_{Russian,imp} = \frac{1}{P}\int_0^1 \frac{f^2(P^{-1}(\tau))}{p^2(P^{-1}(\tau))}d\tau - I^2. \tag{3.32}$$

If the function $g(x)$ is used as a control variate before applying the Russian roulette the estimator becomes

$$\begin{aligned} \langle I \rangle_{Russian,imp,con} &= \begin{cases} \frac{1}{P}\frac{f(P^{-1}(\frac{\tau}{P}))-g(P^{-1}(\frac{\tau}{P}))}{p(P^{-1}(\frac{\tau}{P}))} + J & \text{if } \tau \leq P, \\ J & \text{otherwise.} \end{cases} \\ &= \begin{cases} \frac{1}{P}\frac{f(P^{-1}(\frac{\tau}{P}))}{p(P^{-1}(\frac{\tau}{P}))} - (\frac{1}{P} - 1)J & \text{if } \tau \leq P, \\ J & \text{otherwise.} \end{cases} \end{aligned} \tag{3.33}$$

Figure 3.19b shows the eventual integrand. The function has become more constant because the part differing from 0 has been translated downwards and the constant part has been translated upwards. Even in case the original function is not evaluated the estimate will differ from 0. The variance can be shown to be equal to

$$\sigma^2_{Russian,imp,con} = \frac{1}{P}\int_0^1 \frac{f^2(P^{-1}(\tau))}{p^2(P^{-1}(\tau))}d\tau - (\frac{1}{P} - 1)(2I - J)J - I^2. \tag{3.34}$$

Comparing Eq. (3.32) and Eq. (3.34) shows that the variance will be smaller if the integral of the approximating function lies close enough to the actual integral. More precisely:

$$\sigma^2_{Russian,imp,con} < \sigma^2_{Russian,imp} \Leftrightarrow 0 < J < 2I. \tag{3.35}$$

Russian Roulette Applied to Integral Equations

Russian roulette is usually only applied when the integral is actually part of an integral equation. If it is applied at each level of the recursion the random walk is guaranteed to terminate, albeit in a non-deterministic time. It becomes a finite series $\xi_1, \xi_2, \ldots, \xi_k$ because the integral is estimated as 0 at some point ξ_k. If each ξ_i is sampled according to the PDF $p_i(x)$ with a probability P_i the primary estimator (3.28) in the case of integral equation (3.27) becomes a finite sum:

$$\begin{aligned}
\langle f(x)\rangle_{Russian,recursive} &= g(x) \\
&+ \frac{K(x,\xi_1)}{P_1 p_1(\xi_1)} g(\xi_1) \\
&+ \frac{K(x,\xi_1)}{P_1 p_1(\xi_1)} \frac{K(\xi_1,\xi_2)}{P_2 p_2(\xi_2)} g(\xi_2) \\
&+ \ldots \\
&+ \frac{K(x,\xi_1)}{P_1 p_1(\xi_1)} \frac{K(\xi_1,\xi_2)}{P_2 p_2(\xi_2)} \cdots \frac{K(\xi_{k-1},\xi_k)}{P_k p_k(\xi_k)} g(\xi_k) \\
&= \sum_{i=0}^{k} \left[\prod_{j=1}^{i} \frac{K(\xi_{j-1},\xi_j)}{P_j p_j(\xi_j)} \right] g(\xi_i),
\end{aligned} \tag{3.36}$$

where $\xi_0 = x$.

In many physical applications the normalisation value of the kernel $K(x,y)$ is smaller than 1 for each $x \in [0,1]$:

$$\int_0^1 K(x,y)dy < 1.$$

The kernel can then be used as a so-called *subcritical PDF*. This is equivalent to sampling according to its corresponding normalised PDF and using the normalisation constant as the probability for Russian roulette:

$$P_i = \int_0^1 K(\xi_{i-1},y)dy,$$

$$p_i(x) = \frac{K(\xi_{i-1},x)}{P_i}.$$

A subcritical PDF is therefore equally valid in estimators as a regular PDF. In this case the primary estimator simplifies to the elegant expression

$$\langle f(x)\rangle_{subcriticalPDF} = \sum_{i=1}^{k} g(\xi_i). \tag{3.37}$$

This approach entirely follows the spirit of importance sampling. The integral is likely to give only a small additional contribution if the kernel is small. By setting a low probability for continuing the random walk at this point more work can be put in solving other –hopefully more important– integrals and integral equations. Ermakow discusses alternative estimators in [30].

3.9 Next Event Estimation

In our applications the estimator (3.37) suffers from a high variance because the source function $g(x)$ will be mostly 0 over the domain. The estimate will only spuriously get large contributions, which is an unfavourable hit-or-miss sampling effect. In that case *next event estimation* offers great advantages. The original integral of integral equation (3.27) can be split up into two separate integrals each of which is solved in a different way:

$$\begin{aligned} f(x) &= g(x) + \int_0^1 K(x,y)f(y)dy \\ &= g(x) + h(x), \end{aligned}$$

where

$$\begin{aligned} h(x) &= \int_0^1 K(x,y)f(y)dy \\ &= \int_0^1 K(x,y)[g(y)+h(y)]dy \\ &= \int_0^1 K(x,y)g(y)dy + \int_0^1 K(x,y)h(y)dy. \end{aligned}$$

The former integral is preferably estimated by sampling according to a PDF that is shaped similar to the dominant factor $g(y)$ of the integrand. In the latter integral $h(y)$ is unknown so it is sampled as before. Formally, if at each level of the recursion ζ_i is sampled according to the PDF $p_i(z)$ and each ξ_i according to the subcritical PDF $K(\xi_{i-1}, x)$ the primary estimator becomes

$$\langle f(x)\rangle_{nextevent} = g(x) + \langle h(x)\rangle_{nextevent},$$

where

$$\begin{aligned} \langle h(x)\rangle_{nextevent} &= \frac{K(x,\zeta_1)g(\zeta_1)}{p_1(\zeta_1)} + \langle h(\xi_1)\rangle_{nextevent} \\ &= \frac{K(x,\zeta_1)g(\zeta_1)}{p_1(\zeta_1)} + \frac{K(\xi_1,\zeta_2)g(\zeta_2)}{p_2(\zeta_2)} + \langle h(\xi_2)\rangle_{nextevent} \\ &= \ldots \\ &= \sum_{i=1}^{k} \frac{K(\xi_{i-1},\zeta_i)g(\zeta_i)}{p_i(\zeta_i)}. \end{aligned}$$

Combining the two equations yields

$$\langle f(x)\rangle_{nextevent} = g(x) + \sum_{i=1}^{k} \frac{K(\xi_{i-1},\zeta_i)g(\zeta_i)}{p_i(\zeta_i)}. \tag{3.38}$$

3.10 Summary

In this chapter we have presented the basic principles of Monte Carlo methods that are important for the global illumination algorithms presented in Chapter 4. All Monte Carlo methods are essentially techniques for numerical integration. They compute an estimate for a definite integral on the basis of random samples. The primary estimator expresses the estimate as a function of a single sample. An estimator is unbiased if its expected value is equal to the actual value of the integrand. Secondary estimators average several uncorrelated results from the primary estimator into a more reliable estimator. The uncertainty of the result is quantified by the variance, which is inversely proportional to the number of samples.

As the variance decreases only slowly when increasing the number of samples, research on Monte Carlo methods is mainly directed towards finding other variance reducing techniques. Most of these techniques can be regarded as transformations of the integral that try to smooth the integrand, in the sense of making it more constant. Smoother functions are easier to integrate numerically. For Monte Carlo methods this means a lower variance for the same number of samples, or the same variance for a smaller number of samples.

A first important optimisation is stratification of the samples, to keep them from clumping together. Samples that are spread out more or less evenly over the integration domain will in general be better at capturing the shape of the integrand. The most straightforward unbiased technique is to subdivide the integration domain into subdomains, then take a single sample in each of the subdomains to estimate the partial integrals and finally sum the estimates. In one dimension the subdomains are usually equally sized intervals. This can be extended to an arbitrary or even an infinite number of dimensions using N-rooks sampling.

Importance sampling is intended to select more samples in regions where the integrand is large, by using an appropriate probability density function (PDF). The estimator then has to be adapted accordingly. Ideally the PDF should be proportional to the integrand. Importance sampling corresponds to uniformly sampling the integrand after a transformation. The transformation is based on the inverse of the probability distribution function, which therefore has to be practically computable.

If several potentially good PDFs are available they can be combined using heuristics such as the balance heuristic that have provable good properties. This combining of estimators is equivalent to partitioning the integrand and summing the estimators for the different parts. We have shown how heuristics with non-symmetrical estimators can be further improved by deriving symmetrical estimators.

Control variates are functions that approximate the integrand and that unlike the integrand can be integrated analytically. Sampling the difference between the control variate and the integrand and adding the analytically computed integral yields a new primary estimator with a lower variance.

Monte Carlo methods can handle integral equations as well. The actual solution of the integral equation is only estimated in a specific point by treating the integral equation as a recursive definite integral. The infinite recursion can be stopped using Russian roulette. We have shown how Russian roulette and importance sampling can be combined with control variates, and under which condition this reduces the variance further. A final variance reduction technique, specifically for integral equations, is next event estimation. It always estimates the source term one step ahead in the recursion, in order to sample its shape more efficiently.

In general, we have shown explicitly how all the variance reducing techniques discussed can be seen as transformations of the original integral. The variance reduction results from the transformed integrand being smoother and therefore easier to integrate. This observation also has its practical use in that it shows that different variance reduction techniques, e.g. stratified sampling and importance sampling, can be combined by consecutively applying the corresponding transformations.

The transformations also show that all these variance reduction techniques are unbiased: averaging an increasing number of primary estimates is guaranteed to converge to the desired solution. Still, a large number of samples is often required to reduce the uncertainty to an acceptable level. In practice Monte Carlo methods often only have an accuracy of 0.1% or less. A primary rule is therefore to compute quantities analytically wherever possible. Somewhat ironically all variance reducing techniques also work this way. The more information one has about the problem the further the probabilistic Monte Carlo aspect and its associated variance can be reduced. As Trotter and Tukey put it: "The only good Monte Carlos are dead Monte Carlos" [123] (quoted in [38]).

The question remains whether deterministic methods are not preferable over probabilistic methods. The inherent uncertainty on the results of Monte Carlo methods may feel awkward. Deterministic methods on the other hand often have strict error bounds. However, these bounds only hold under certain conditions on the integrand, which can rarely be met in practical applications. Physical quantities for instance often have complex discontinuities and oscillations in space and in time. Furthermore, deterministic integration techniques typically suffer from the curse of dimensionality, which is far less of a problem for Monte Carlo techniques. Shreider suggests that Monte Carlo methods may be interesting for smooth integrands in 5 or more dimensions, for piece-wise smooth integrands in 2 or more dimensions and for less well-behaved discontinuous integrands in any number of dimensions [114, pp. 124–125]. Progress in deterministic integration techniques have undoubtedly shifted these boundaries considerably. Davis and Rabinowitz [23, pp. 416–417] give an overview of classes of deterministic integration techniques for integrals of dimensions up to 2, 7, 15 and even greater. Monte Carlo methods may still be a viable alternative for problems with complex and high-dimensional integrands.

Other research has investigated the possibilities of biased estimators that still have a lower error overall and the possibilities of quasi-Monte Carlo methods. Quasi-Monte Carlo methods apply the same techniques as Monte Carlo methods, only substituting the random numbers by deterministic sequences of numbers that have desirable properties. They allow to put more rigid bounds on the integration error as with deterministic methods, also under similar conditions. Although their convergence rate is claimed to be far superior for some problems, some technical difficulties still remain. Their application to more complex problems such as the global illumination problem largely remains to be explored so far.

In Chapter 4 we will discuss the application of the standard Monte Carlo techniques explained in this chapter to the global illumination problem and to physically-based rendering in particular.

Chapter 4

Monte Carlo Methods Applied to the Global Illumination Problem

In this chapter we will apply the generic Monte Carlo methods discussed in Chapter 3 to the mathematical models of the global illumination problem discussed in Chapter 2. While the basic principles are straightforward almost all variance reduction techniques require more information about the problem at hand. We will discuss various alternatives for the global illumination problem and evaluate their performance.

Although the mathematical models for global illumination are completely general we will restrict ourselves to computing the radiant flux or average radiance for pixels, as shown in Fig. 2.5, rather than for patches or other sets of points and directions in the scene, as shown in Fig. 2.4. The algorithms will therefore be ray tracing-like algorithms rather than radiosity-like algorithms. They produce photo-realistic images of a scene rather than radiance values for all points and directions in the scene. The advantage of this approach is flexibility; the relative simplicity of the desired output, if only because of the data structure required to represent the results, allows for more complex problems and more versatile algorithms.

The choice for a Monte Carlo approach is not self-evident. In the case of physically-based rendering at least a few arguments plead in favour of this approach however:

- The integrals that describe the problem are high-dimensional. In general some integrand has to be sampled over each pixel to obtain anti-aliasing, which accounts for two dimensions. If depth of field is to be simulated the integrand also has to be sampled over the camera aperture, accounting for another two dimensions. Light tracing techniques on the other hand sample over surfaces and hemispherical directions of the light sources, rather than over the pixels and the camera aperture. This also corresponds to four dimensions. Some algorithms combine both sampling techniques. If motion blur is to be simulated the integrand has to be sampled over time, adding yet another dimension. The actual global illumination problem itself is defined by integral equations over the two-dimensional hemisphere. Monte Carlo methods are less sensitive to the dimension of the problem than deterministic techniques.

- The integrands in all global illumination equations are complex. The radiance function and the potential function can be shown to be bounded because of physical reasons, but they are not necessarily of bounded variation. They usually have an unwieldy number of first-order and higher-order discontinuities. Radiosity methods can still determine these discontinuities explicitly, as they consider a simplified problem with perfectly diffuse surfaces. The radiance functions only have discontinuities in the 2D space of the surfaces as the functions are assumed constant along the directional dimensions, i.e. over the hemisphere. Discontinuity meshing algorithms can thus subdivide the finite elements, e.g. [98, 44, 45, 74]. While the associated computational cost is already high for diffuse surfaces, it becomes prohibitive for general radiance or potential functions in the 4D or 5D space of positions and directions. This presents problems to any numerical technique that relies on the integrands being smooth or well-behaved. Monte Carlo methods are generally more robust in this respect.

- The required accuracy of the results is very small compared to many other applications. The computed images are generally intended as an end product, for viewing purposes. The human eye has a great dynamic range but only a limited absolute sensitivity. This is partially reflected in the accuracy of "true colour" displays that are typically limited to 8 bits per colour channel. An accuracy of only 0.1 to 1% will therefore be perfectly acceptable in general.

As discussed in Chapter 3 these arguments support the choice of Monte Carlo methods over deterministic approaches for solving the global illumination problem. Their versatility allows to attack the problem in its most general shape without making gross simplifications a priori. We are interested in physically based rendering without restraining the geometry and optical properties of the scenes or the optical effects to be rendered. Monte Carlo methods seem most suitable for this purpose as they only point-sample the functions that define the geometry and the optical properties. We attempt not to resort to shortcuts for specific cases such as perfectly diffuse or perfectly specular surfaces. The rendering algorithms should be generally applicable.

The most obvious cost of this versatility is a slower convergence. Cook [19] notes that any remaining stochastic noise in the images is less objectionable to our visual system than the aliasing effects of deterministic techniques. In spite of this and of the low accuracy that is required the slow convergence remains the major obstacle for Monte Carlo rendering techniques to be practical. Reducing their errors to an acceptable level with acceptable computation times calls for a detailed study of their application to physically-based rendering.

A pragmatic point of view is that Monte Carlo methods still play an important role in many algorithms for physically-based rendering, in spite of their disadvantages. Chief examples are the computation of form-factors in radiosity methods, e.g. [128, 41] and especially the final rendering steps in many rendering algorithms, e.g. [127, 117, 11, 137]. It may therefore be worthwhile to study their application to this domain more thoroughly.

We will stress the unbiasedness of the Monte Carlo methods we present. This is an aspect that is often neglected in practice. Many earlier Monte Carlo rendering algorithms make approximations that are intuitive and yield visually pleasing results. Unfortunately they sometimes lack a theoretical basis. Especially in physically-based rendering we should only have to resort to shortcuts when all standard mathematical techniques are exhausted. Kirk and Arvo [56] noted the undesired effects of biased adaptive sampling techniques in computer graphics. Unbiasedness provides a minimal rule by which to develop Monte Carlo algorithms. It guarantees that the algorithms will eventually converge to the correct solution. Even if this remains a rather unhelpful argument in practice, recent progress, which we will discuss later, seems to indicate that there is room for improvement in this direction. This makes unbiased techniques worthwhile to explore.

In this chapter we will show how the different mathematical models for global illumination can lead to different rendering algorithms by straightforward Monte Carlo integration. The most commonly known model is the rendering equation as the basis of stochastic ray tracing algorithms. In a similar way the potential equation naturally gives rise to various light tracing algorithms. We will discuss their specific advantages and disadvantages and apply standard variance reduction techniques to improve their efficiency. Finally, we will demonstrate how the global reflectance distribution function and its corresponding integral equations lead to a new algorithm, called bidirectional path tracing. This algorithm effectively combines the favourable properties of previous algorithms. Test examples will illustrate the practical performance of all algorithms and optimisations.

4.1 Monte Carlo Methods Applied to the Rendering Equation – Path Tracing

4.1.1 Basic Algorithm

A first approach to solve the global illumination problem is based on the rendering equation. It leads to various stochastic ray tracing algorithms. Ray tracing was first introduced by Whitted [135] as an empirical and deterministic algorithm for simulating various illumination effects. Cook [20] extended the algorithm with stochastic sampling techniques [20, 19]. Kajiya [53] was the first to recognise that the algorithm actually tries to solve the rendering equation. Many other researchers have developed this idea further to construct optimised physically based rendering algorithms, e.g [134, 106, 99, 69]. They have proposed various techniques such as stratified sampling, adaptive sampling and filtering. Some of these techniques are biased and some of them are not. In the following sections we will present an overview of common unbiased optimisations in the notation of our framework and discuss their merits.

The derivation of the basic algorithm starts from the expression of the radiant flux in terms of radiance (2.6) and the rendering equation (2.11), i.e. the integral equation that describes the radiance function:

$$\Phi = \int_A \int_{\Omega_x} L(x, \Theta_x) W_e(x, \Theta_x) \left|\Theta_x \cdot N_x\right| d\omega_x d\mu_x, \tag{2.6}$$

$$L(x, \Theta_x) = L_e(x, \Theta_x) + \int_{\Omega_x^{-1}} L(y, \Theta_y) f_r(x, \Theta_y, \Theta_x) \left|\Theta_y \cdot N_x\right| d\omega_y. \tag{2.11}$$

We would like to compute the flux or average radiance Φ through each of the pixels in turn. Sampling stochastic variables x_0 and Θ_{x_0} according to some PDF $p_0(x, \Theta_x)$ over all surface points and corresponding hemispheres yields the following primary estimator for the integral of Eq. (2.6):

$$\langle \Phi \rangle_{path} = \frac{W_e(x_0, \Theta_{x_0}) \left|\Theta_{x_0} \cdot N_{x_0}\right|}{p_0(x_0, \Theta_{x_0})} \langle L(x_0, \Theta_{x_0}) \rangle_{path}.$$

For image-based rendering algorithms the self-emitted potential for all points and directions depends on the eye point and on the pixel and is therefore only known indirectly. It depends on the camera model, as explained in some more detail in Appendix A. In practice a point x_{-1} on the camera aperture and the direction Θ_{x_0} pointing through the pixel are sampled according to a PDF $p_{-1}(x, \Theta_x)$. Point x_0 is then determined by tracing a ray to the nearest surface: $x_0 = r(x_{-1}, \Theta_{x_0}^{-1})$. The PDF $p_0(x, \Theta_x)$ that is implicitly used for sampling x_0 and Θ_{x_0} can easily be evaluated a posteriori:

$$p_0(x_0, \Theta_{x_0}) = \frac{p_{-1}(x_{-1}, \Theta_{x_0}) \left|\Theta_{x_0} \cdot N_{x_0}\right|}{\|x_{-1} - x_0\|^2}.$$

The radiance function is unknown, but the particular value $L(x_0, \Theta_{x_0})$ can be estimated by treating the rendering equation (2.11) as an infinitely recursive integral, as explained in Section 3.7. Applying Russian roulette as in Section 3.8 ensures that the recursion terminates after a finite number of steps without introducing a bias. If the probability for continuing the random walk at each level of the recursion is P_i and each direction Θ_{x_i} is sampled according to a PDF $p_i(\Theta_x)$ over the hemisphere the general primary estimator (3.36) becomes

$$\begin{aligned}\langle\Phi\rangle_{path} &= \frac{W_e(x_0,\Theta_{x_0})\,|\Theta_{x_0}\cdot N_{x_0}|}{p_0(x_0,\Theta_{x_0})} \\ &\quad\times\sum_{i=0}^{l}\left[\prod_{j=1}^{i}\frac{f_r(x_{j-1},\Theta_{x_j},\Theta_{x_{j-1}})\left|\Theta_{x_j}\cdot N_{x_{j-1}}\right|}{P_j p_j(\Theta_{x_j})}\right]L_e(x_i,\Theta_{x_i}),\end{aligned} \tag{4.1}$$

where the length of the random walk $x_0,\ldots,x_l$ defines l. Each point $x_i = r(x_{i-1},\Theta_{x_i}^{-1})$, as already specified in the rendering equation (2.11). This is implemented by tracing a ray from point x_{i-1} along direction $\Theta_{x_i}^{-1}$ and finding the nearest point x_i (Fig. 4.1). The resulting algorithm obviously is a generic form of path tracing, without explicit sampling of the light sources (Fig. 4.2). The algorithm was first presented by Kajiya [53] as a Monte Carlo solution for physically-based rendering. Rays are sent out from the virtual eye point through a pixel. They bounce through the scene and contribute to the estimate of the flux every time they hit a light source. As such it is a so-called light *gathering* approach. Figures 4.3 and 4.4 demonstrate the principles on an example scene.

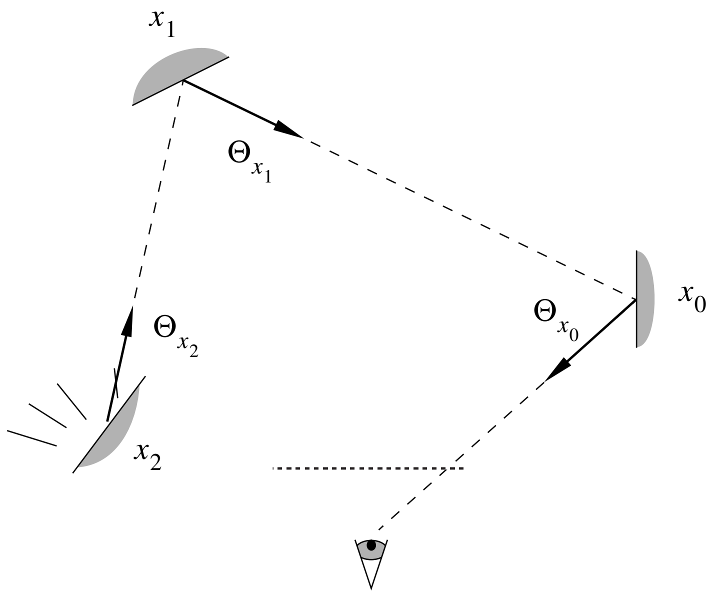


Figure 4.1: Explanation of the symbols used in the path tracing algorithm.

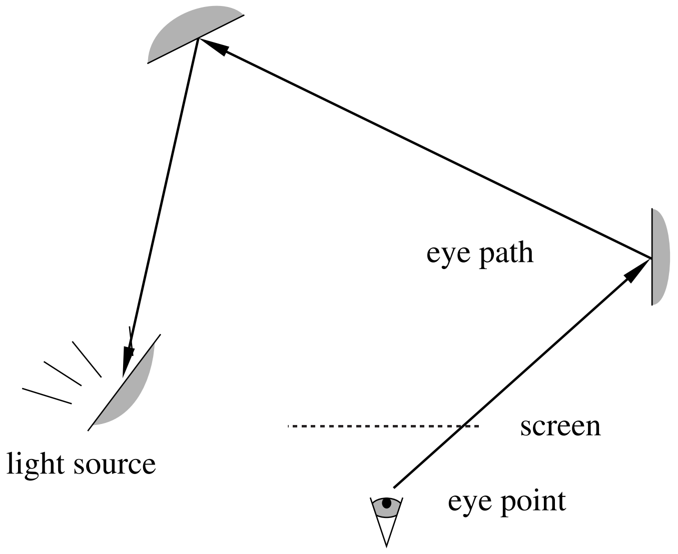


Figure 4.2: Schematic overview of the path tracing algorithm. The radiant flux through a pixel has to be estimated. The tracing of a primary ray from the virtual eye point through a pixel corresponds to sampling the expression for the flux. The subsequent random walk through the scene corresponds to recursively estimating the radiance values. Each time a light source is hit a contribution is added to the estimate.

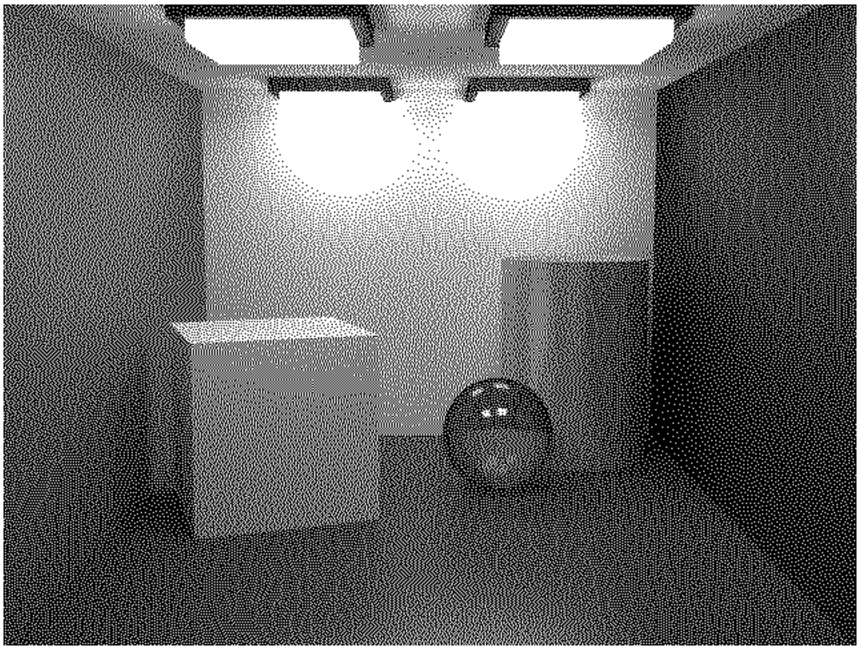


Figure 4.3: An example scene rendered with the path tracing algorithm. The algorithm is best suited for scenes with mostly specular surfaces and large diffuse luminaires. This scene contains large light boxes and a few objects with different geometries and optical characteristics. Note the typical global illumination effects: indirect illumination, glossy reflections, soft shadows and "colour bleeding" of the coloured objects on the white surfaces.

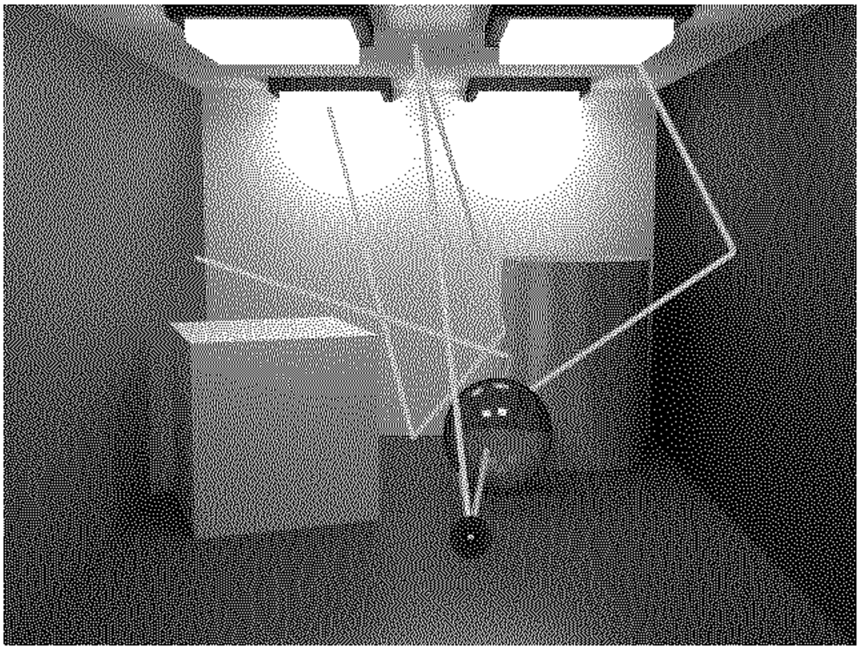


Figure 4.4: A few examples of random walks bouncing through the scene (blue rays), taken from an actual simulation. In this case all walks start from the same pixel in the lower part of the image. Whenever a ray hits an emitting surface a contribution is added to the pixel. If the light sources are small, few rays will hit them. These few but possibly large contributions will result in a large variance.

4.1.2 Stratified Sampling

Multi-dimensional stratified sampling as presented in Section 3.3 can also be applied to the expression of the radiant flux (2.6) and to the rendering equation (2.11). In the context of stochastic ray tracing several researchers discuss the use of sampling techniques that try to distribute the samples uniformly, such as jittered sampling and Poisson disc sampling, in one and two dimensions [25, 71, 19]. In his PhD dissertation Shirley [106] compares two-dimensional N-rooks sampling to Poisson disc sampling and jittered sampling. On the basis of practical test results and a theoretical equidistribution metric called *discrepancy* [136, 108] N-rooks sampling compares favourably to the alternative methods. Heinrich and Keller [46] include several quasi-Monte Carlo methods in the comparison. Their tests indicate that the convergence rates are of the same order of magnitude. Chiu *et al.* [12] present a combination of N-rooks and grid sampling in two dimensions that combines the properties of both.

For higher dimensions Cook [19] extends the stratification by constructing two-dimensional random prototype patterns, specifically for sampling the pixel, the lens, the reflections, etc. Mitchell [79], Shirley [106] and Schlick [99] discuss multi-dimensional N-rooks sampling. This approach is particularly attractive because of its simplicity. Mitchell [80] compares several stratification techniques in two and three dimensions. He shows that quasi-Monte Carlo distributions such as Zaremba-Hammersley points can have a better theoretical asymptotic behaviour than N-rooks sampling. Unfortunately the error bounds derived for quasi-Monte Carlo techniques are not applicable, because the integrands of the global illumination problem are rarely of limited variation; they are most often highly discontinuous. Heinrich and Keller [47] and Keller [55] experiment with Halton and Hammersley sequences of quasi-random points for solving the rendering equation for a simple scene. The results look promising, although it is not obvious if they can be extrapolated to more complex scenes. A practical problem is that the sequences can be constructed for any fixed number of dimensions but to our knowledge not for an arbitrary and a priori unknown number, as we require.

One can wonder whether stratified sampling is worth bothering with, as it derives its variance reduction from coherence in the integrand. The integrand is highly discontinuous over the hemisphere, because of the discontinuous incoming radiance function. After even a single step in the recursion, i.e. after one reflection on the ray path, there will often be little coherence left. However, stratified sampling can at least be worthwhile for sampling the initial expression for the flux which usually has some coherence. The primary rays will have a better spatial distribution for sampling the surfaces visible through the pixel. Even if they hit a single surface the fraction of rays reflected after Russian roulette is a better approximation of the desired fraction. Each of the reflected rays yields a contribution to the estimate. Reducing the randomness of this fraction therefore significantly reduces the randomness of the result. This reasoning can be extended further. If subsequently reflected rays are sampled according to a highly specular BRDF the following integrands may still be coherent, which can again be exploited by stratified sampling.

Overall, stratified sampling has the potential to improve the results and this at little cost. It should therefore be considered a worthwhile addition to the set of variance reduction techniques for path tracing as well. A specific advantage of N-rooks sampling is that it offers a perfect stratification of the individual dimensions, even if one or more dimensions fold into a point, e.g. when rendering images without depth of field or motion blur.

As noted in Sections 3.3 and 3.4, the effectiveness of any stratification technique depends a lot on the parametrisation of the integrals. It therefore also depends on the way any importance sampling scheme is applied, as one scheme may be implemented using several alternative transformations or parametrisations. A transformation that maps a "compact" region of the unit square onto some distorted region of the hemisphere will be just as effective as any other equivalent transformation as far as importance sampling is concerned, but it will decrease the effectiveness of the stratification.

4.1.3 Importance Sampling

As discussed in Section 3.4 importance sampling should attempt to select more samples in important regions of the integrand. In the context of ray tracing it is commonly used to select reflection directions at surfaces, such that more rays are sent in the specular reflection directions of the hemisphere. Deterministic, non-physical ray tracing algorithms even send a single reflection ray in the perfect reflection direction. This eliminates the uncertainty on the result and yields noiseless images. It is however a very crude approximation, which is definitely biased. Since the introduction of the rendering equation physically more correct approaches have been proposed, e.g. [53, 106, 57, 113]. Numerous alternative sampling techniques are possible. We will discuss importance sampling in a slightly more generic way in the context of our framework.

The integral of Eq. (2.6) has a known part $W_e(x, \Theta_x)\,|\Theta_x \cdot N_x|$, which is different for each pixel. The most natural choice is to sample the integral according to a PDF that is proportional to this part. The PDF for the stochastic variables x_0 and Θ_{x_0} then equals

$$p_0(x, \Theta_x) = \frac{W_e(x, \Theta_x)\,|\Theta_x \cdot N_x|}{W}, \tag{4.2}$$

where W is the normalisation factor:

$$W = \int_A \int_{\Omega_x} W_e(x, \Theta_x)\,|\Theta_x \cdot N_x|\, d\omega_x d\mu_x.$$

This PDF ensures that points and directions are sampled that contribute to the pixel for which the flux is being computed, proportionally to their importance for the flux. Depending on the camera model, which determines the importance $W_e(x, \Theta_x)$ (cfr. Appendix A.2), this may or may not be

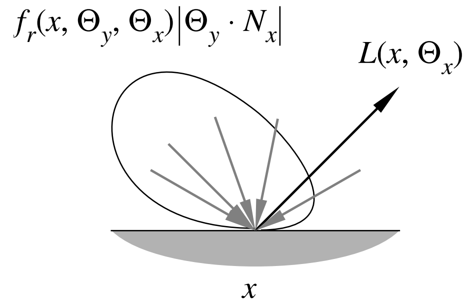


Figure 4.5: Importance sampling of the rendering equation is usually based on the local reflectance properties of the surface. The optimisation selects more samples along incoming directions for which the BRDF $f_r(x, \Theta_y, \Theta_x)$ times the cosine of the incoming direction with the normal $|\Theta_y \cdot N_x|$ is large. The path tracing algorithm then traces more rays along these important directions. Importance sampling will mainly reduce the variance of the estimator for specular surfaces, where the BRDF largely determines the shape of the integrand.

perfectly feasible. If not, it is in general possible to construct an approximating PDF and modify the estimators accordingly.

The rendering equation (2.11) has a known part $f_r(x_{i-1}, \Theta_x, \Theta_{x_{i-1}}) \left|\Theta_x \cdot N_{x_{i-1}}\right|$ at each point in the recursion. The integral of this factor over the hemisphere is the directional-hemispherical reflectance $\rho(x_{i-1}, \Theta_{x_{i-1}})$. According to the law of conservation of energy (2.9) it is always smaller than 1. The function can therefore in principle be used as a subcritical PDF for sampling Θ_{x_i} over the hemisphere (Fig. 4.5):

$$P_i\, p_i(\Theta_x) = f_r(x_{i-1}, \Theta_x, \Theta_{x_{i-1}}) \left|\Theta_x \cdot N_{x_{i-1}}\right|. \tag{4.3}$$

As in Eq. (3.37) the primary estimator (4.1) now simplifies to

$$\langle\Phi\rangle_{path,imp} = W \sum_{i=0}^{l} L_e(x_i, \Theta_{x_i}). \tag{4.4}$$

The importance sampling based on the BRDFs ensures that more rays are sent in directions that will have large contributions, at least according to the local reflective properties. Unfortunately it is rarely possible to sample perfectly according to the proposed PDFs, because they cannot be integrated and then inverted. One usually has to sample according to approximate PDFs instead and apply the more general expression of Eq. (4.1). The estimator is unbiased but may have a larger variance. In [66] we discuss various aspects of importance sampling according to a modified Phong BRDF.

More importantly, the PDFs may not be ideal because of the unknown radiance function in the integrals. Simple importance sampling ignores this factor, although it may have a large influence on the shape of the integrand. More specifically directions towards light sources and bright areas in general are important. Additional information about the radiance function can therefore help to construct better PDFs, as we will show in Section 4.1.7.

Importance sampling is essential for path tracing as it is the only way to efficiently capture the possibly sharp peaks of specular BRDFs that dominate the integrand. The radiance function will then be relatively constant compared to the BRDF so that the latter is a well-suited basis for a PDF. The algorithm as a whole therefore performs best on scenes with specular surfaces and diffuse illumination and large diffuse luminaires.

4.1.4 Next Event Estimation

In many common scenes the luminaires are relatively small. Rays then have a low probability of hitting a light source and most random walks will not contribute to the estimate of the flux. The effect is similar to hit-or-miss sampling, which also has a large variance. As discussed in Section 3.9, it can be remedied by applying next event estimation to the sampling of the rendering equation (2.11):

$$\begin{aligned} L(x,\Theta_x) &= L_e(x,\Theta_x) + \int_{\Omega_x^{-1}} L(y,\Theta_y) f_r(x,\Theta_y,\Theta_x) \left|\Theta_y \cdot N_x\right| d\omega_y \\ &= L_e(x,\Theta_x) + L_r(x,\Theta_x), \end{aligned}$$

where the latter term L_r physically corresponds to the radiance resulting from incident light that is reflected:

$$L_r(x,\Theta_x) = \int_{\Omega_x^{-1}} L(y,\Theta_y) f_r(x,\Theta_y,\Theta_x) \left|\Theta_y \cdot N_x\right| d\omega_y$$

$$
\begin{aligned}
&= \int_{\Omega_x^{-1}} [L_e(y,\Theta_y) + L_r(y,\Theta_y)] f_r(x,\Theta_y,\Theta_x)\, |\Theta_y \cdot N_x|\, d\omega_y \\
&= \int_{\Omega_x^{-1}} L_e(y,\Theta_y) f_r(x,\Theta_y,\Theta_x)\, |\Theta_y \cdot N_x|\, d\omega_y \\
&\quad + \int_{\Omega_x^{-1}} L_r(y,\Theta_y) f_r(x,\Theta_y,\Theta_x)\, |\Theta_y \cdot N_x|\, d\omega_y \\
&= \int_A L_e(y,\Theta_y) f_r(x,\Theta_{y\to x},\Theta_x) v(x,y) \frac{|\Theta_{y\to x} \cdot N_x|\, |\Theta_{y\to x} \cdot N_y|}{\|x-y\|^2} d\mu_y \\
&\quad + \int_{\Omega_x^{-1}} L_r(y,\Theta_y) f_r(x,\Theta_y,\Theta_x)\, |\Theta_y \cdot N_x|\, d\omega_y \\
&= L_d(x,\Theta_x) + L_i(x,\Theta_x).
\end{aligned}
$$

The former integral expresses the radiance L_d resulting from direct illumination. The latter integral expresses the radiance L_i resulting from indirect illumination. The directly reflected radiance is usually estimated by sampling a point on a light source, hence the transformation to a more convenient expression similar to Eq. (2.13). A simple option for sampling point y in this integral is a PDF that is proportional to the self-emitted radiance integrated over the hemisphere at each point:

$$
p(y) = \frac{L(y)}{L}, \tag{4.5}
$$

where:

- $L(y) = \int_{\Omega_y} L_e(y,\Theta_y)\, |\Theta_y \cdot N_y|\, d\omega_y$, the *self-emitted radiosity* or *radiant exitance* at point y $[W/m^2]$,
- $L = \int_A \int_{\Omega_y} L_e(y,\Theta_y)\, |\Theta_y \cdot N_y|\, d\omega_y d\mu_y$, the total self-emitted flux in the scene $[W]$.

If all other integrals are sampled as in the previous section, according to the PDF (4.2) and subcritical PDF (4.3), the estimator (4.4) changes into

$$
\begin{aligned}
\langle \Phi \rangle_{path,imp,nextevent} &= W \Bigg[L_e(x_0,\Theta_{x_0}) + \frac{L}{L(y)} \sum_{i=0}^{l} L_e(y,\Theta_{y\to x_i}) \\
&\quad \times f_r(x_i,\Theta_{y\to x_i},\Theta_{x_i}) v(x_i,y) \frac{|\Theta_{y\to x_i} \cdot N_{x_i}|\, |\Theta_{y\to x_i} \cdot N_y|}{\|x_i - y\|^2} \Bigg].
\end{aligned} \tag{4.6}
$$

Figure 4.6 gives an overview of the symbols used in the expressions.

The resulting algorithm is path tracing with explicit sampling of the light sources to compute direct illumination (Fig. 4.7). The algorithm traces rays from the eye point that reflect through the scene. The visibility function is evaluated by means of a shadow ray at each intersection point.

The PDF for sampling the direct illumination (4.5) does not take into account any singularities of the integral. They are evaluated in the estimator (4.6) instead. This is an unfortunate deficiency, as mentioned in Section 3.4. The estimate can become arbitrarily large as a result of the division by $\|x_i - y\|^2$. These large estimates can easily overwhelm the other primary estimates with which they are averaged and eventually show up as bright pixels in the image.

As before, the algorithm is best-suited for scenes with mostly specular surfaces. Only now the next event estimation is most efficient if the light sources are small and if they illuminate most of the scene directly. The PDFs for sampling the light sources then dominate the integrand (Fig. 4.8 and 4.9).

In this particular instantiation the same point on one of the light sources is used at all levels of the recursion. Alternatively a different point can be sampled each time. It is then also possible to use more sophisticated sampling techniques that take into account different factors than just self-emitted radiance, such as distance or subtended solid angle. Shirley *et al.* [112, 113] present several of these techniques for different types of luminaires. Zimmerman and Shirley [137] present a more complex approach to select between different sampling techniques depending on the characteristics of the light sources. In their particular algorithm the light sources have a vast range of characteristics as they comprise all the patches of the scene after a radiosity preprocessing step.

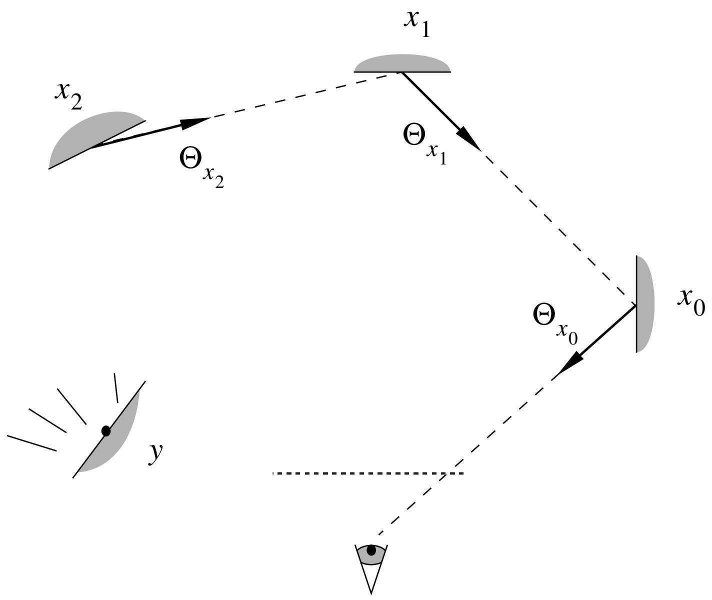


Figure 4.6: Explanation of the symbols used in the path tracing algorithm with event estimation. The random walk remains the same as without next event estimation (Fig. 4.1) but an additional point y is now sampled on a light source.

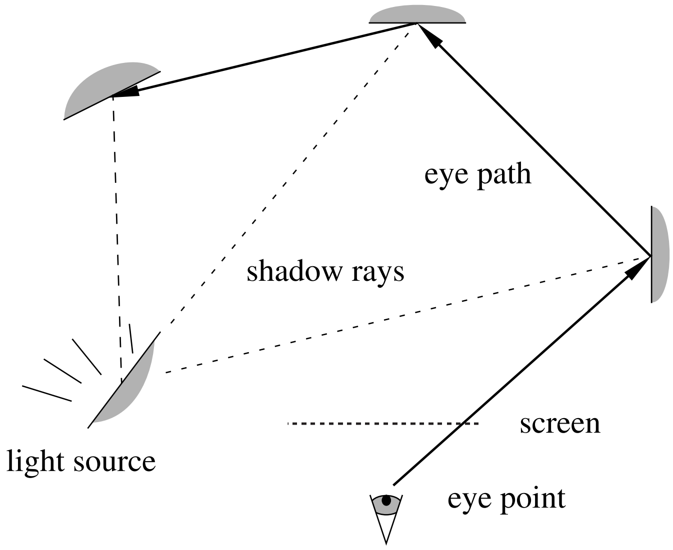


Figure 4.7: Schematic overview of the path tracing algorithm with next event estimation. Direct illumination is now computed explicitly at each point on the random walk by sampling the light sources.

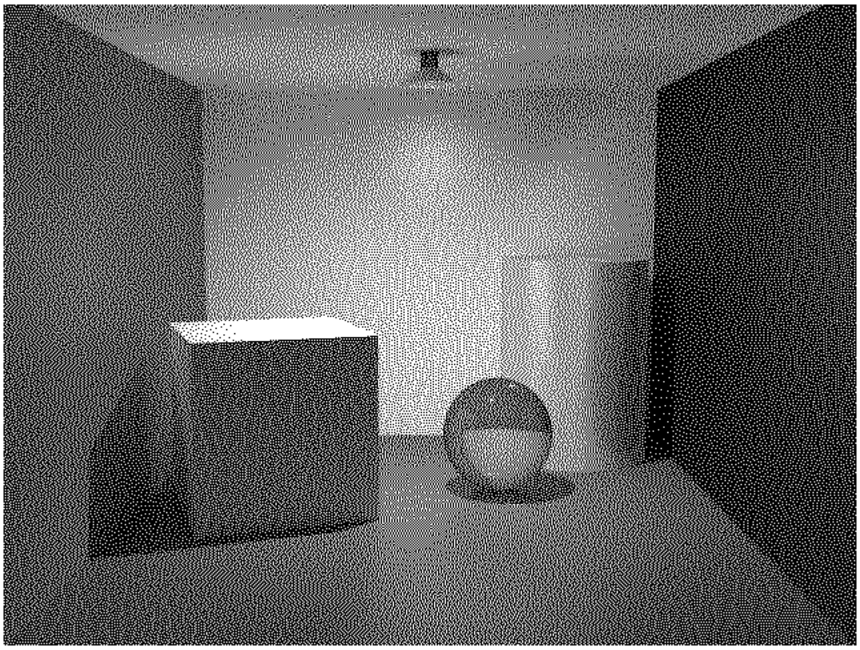


Figure 4.8: An example scene rendered with the path tracing algorithm with next event estimation. The algorithm is best suited for scenes with mostly specular surfaces and small diffuse luminaires. In this case there is a single light spot against the ceiling.

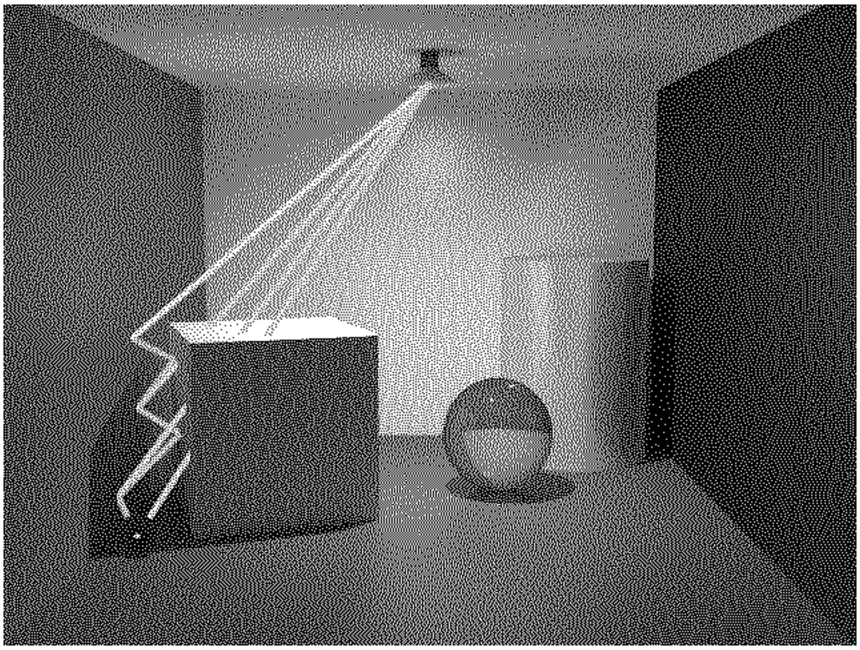


Figure 4.9: An example random walk (blue rays) starting from a pixel in the lower left corner of the image. From each intersection point a shadow ray is cast to the light source. If it does not hit an object (green ray) an illumination contribution is added to the pixel. If it does hit an object (red rays) no contribution is added.

4.1.5 Combining Estimators

One of the applications presented by Veach and Guibas [126] for their heuristics to combine different estimators is the computation of direct illumination. In pure path tracing direct illumination at each point on the eye path is estimated by sampling the hemisphere, preferably according to the BRDF. In path tracing with next event estimation it is estimated by sampling the light sources, preferably according to their radiance or their radiosity. The hemisphere is still sampled however, in order to estimate the indirect illumination. As a ray is already cast for this purpose the sample offers an alternative estimator for the direct illumination as well, without much additional work.

For this purpose the samples have to be specified in the same parameter space, for instance the space of all points on the surfaces. When the algorithm samples a direction and then traces a ray to determine a point the probability density for sampling that point can be computed using a simple transformation. If $p(\Theta)$ is a PDF for sampling Θ_x over the hemisphere Ω_x and if $y = r(x, \Theta_x)$ then the implicit probability density for sampling y is

$$p(y) = p(\Theta_x)\frac{|\Theta_x \cdot N_y|}{\|x-y\|^2}.$$

This value can easily be computed a posteriori. Assuming for instance all samples are taken as in path tracing with next event estimation, according to PDF (4.5), PDF (4.2) and subcritical PDF (4.3), the estimator eventually becomes

$$\begin{aligned}\langle\Phi\rangle_{path,imp,nextevent,comb} &= W\Big[L_e(x_0,\Theta_{x_0}) \\ &+ L\sum_{i=0}^{l}\frac{L_e(y,\Theta_{y\to x_i})v(x_i,y)f_r(x_i,\Theta_{y\to x_i},\Theta_{x_i})\left|\Theta_{y\to x_i}\cdot N_{x_i}\right|\left|\Theta_{y\to x_i}\cdot N_y\right|}{L(y)\|x_i-y\|^2+Lf_r(x_i,\Theta_{y\to x_i},\Theta_{x_i})\left|\Theta_{y\to x_i}\cdot N_{x_i}\right|\left|\Theta_{y\to x_i}\cdot N_y\right|} \\ &+ L\sum_{i=1}^{l}\frac{L_e(y,\Theta_{x_i})f_r(x_{i-1},\Theta_{x_i},\Theta_{x_{i-1}})\left|\Theta_{x_i}\cdot N_{x_{i-1}}\right|\left|\Theta_{x_i}\cdot N_y\right|}{L(y)\|x_i-y\|^2+Lf_r(x_{i-1},\Theta_{x_i},\Theta_{x_{i-1}})\left|\Theta_{x_i}\cdot N_{x_{i-1}}\right|\left|\Theta_{x_i}\cdot N_y\right|}\Big].\end{aligned} \tag{4.7}$$

Apart from the interesting properties discussed in Section 3.5 the estimator is particularly striking because the combination of a bounded but generally inefficient and a more efficient but unbounded estimator results in an estimator that is in general bounded. Practical tests presented in [126] and in Chapter 5 strongly indicate its efficiency for a wide range of illumination conditions.

4.1.6 Control Variates

Control variates as discussed in Section 3.6 can also be applied to the mathematical models of the global illumination problem. An approximating function that is already used as a PDF for importance sampling cannot improve the results any further as a control variate. However, in Section 3.8 we have shown that it can improve the results if the PDF is subcritical. This is the case when sampling the rendering equation (2.11) using the PDF of Eq. (4.3). In [64] we have therefore proposed to derive a simple control variate, proportional to the PDF, by assuming that the incoming radiance is equal to some constant ambient radiance L_a (Fig. 4.10). Rewriting the rendering equation yields

$$\begin{aligned} L(x,\Theta_x) &= L_e(x,\Theta_x) + \int_{\Omega_x^{-1}} L(y,\Theta_y) f_r(x,\Theta_y,\Theta_x) \left|\Theta_y \cdot N_x\right| d\omega_y \\ &= L_e(x,\Theta_x) + \int_{\Omega_x^{-1}} [L(y,\Theta_y) - L_a] f_r(x,\Theta_y,\Theta_x) \left|\Theta_y \cdot N_x\right| d\omega_y \\ & \quad + \int_{\Omega_x^{-1}} L_a f_r(x,\Theta_y,\Theta_x) \left|\Theta_y \cdot N_x\right| d\omega_y \\ &= L_e(x,\Theta_x) + \int_{\Omega_x^{-1}} [L(y,\Theta_y) - L_a] f_r(x,\Theta_y,\Theta_x) \left|\Theta_y \cdot N_x\right| d\omega_y \\ & \quad + \rho(x,\Theta_x) L_a. \end{aligned}$$

If the integrals are sampled according to the same PDFs, estimator (4.4) becomes

$$\begin{aligned} \langle\Phi\rangle_{path,imp,con} &= W \Big[L_e(x_0,\Theta_{x_0}) + \rho(x_0,\Theta_{x_0}) L_a \\ & \quad + \sum_{i=1}^{l} [L_e(x_i,\Theta_{x_i}) - L_a + \rho(x_i,\Theta_{x_i}) L_a] \Big] \\ &= W \left[L_a + \sum_{i=0}^{l} [L_e(x_i,\Theta_{x_i}) - (1-\rho(x_i,\Theta_{x_i})) L_a] \right]. \end{aligned} \tag{4.8}$$

The control variates add some approximate contribution to the estimate, even if Russian roulette terminates the random walk at some random point. The result remains unbiased by only estimating the difference with the approximation whenever the random walk continues.

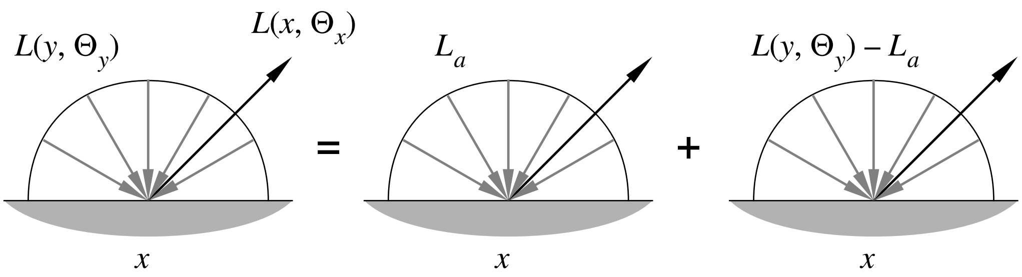


Figure 4.10: A simple control variate for the rendering equation can be derived by selecting a constant ambient incoming radiance L_a. The resulting reflected radiance is equal to $\rho(x,\Theta_x)L_a$, which can often be computed analytically. Instead of sampling $L(y,\Theta_y)$ the path tracing algorithm now sums this approximation with the result from sampling the difference $L(y,\Theta_y) - L_a$, to obtain an unbiased estimate for the actual reflected radiance $L(x,\Theta_x)$. If L_a is chosen appropriately the estimator will have a lower variance.

Control variates can also be combined with next event estimation. It then only approximates the indirect incoming radiance as the self-emitted radiance is computed separately. Estimator (4.6) becomes

$$\langle\Phi\rangle_{path,imp,nextevent,con} = W\left[L_e(x_0,\Theta_{x_0}) + L_a + \sum_{i=0}^{l}\left[L\frac{L_e(y,\Theta_{y\to x_i})}{L(y)}\right.\right. \tag{4.9}$$
$$\left.\left.\times\ f_r(x_i,\Theta_{y\to x_i},\Theta_{x_i})v(x_i,y)\frac{|\Theta_{y\to x_i}\cdot N_{x_i}|\,|\Theta_{y\to x_i}\cdot N_y|}{\|x_i-y\|^2} - (1-\rho(x_i,\Theta_{x_i}))L_a\right]\right].$$

Eq. (3.35) expresses formally how close the integrated control variate should approximate the actual integral value to reduce the variance of the sampling process. For estimator (4.8) without next event estimation this means that L_a should be such that at each point x_i

$$0 < \rho(x_i,\Theta_{x_i})L_a < 2L_r(x_i,\Theta_{x_i}),$$

where L_r is the actually reflected radiance, as defined in Section 3.9.

For estimator (4.9) with next event estimation it means that L_a should be such that

$$0 < \rho(x_i,\Theta_{x_i})L_a < 2L_i(x_i,\Theta_{x_i}),$$

where L_i is the reflected radiance resulting from indirect illumination.

The functions $L_r(x,\Theta_x)/\rho(x,\Theta_x)$ and even more $L_i(x,\Theta_x)/\rho(x,\Theta_x)$ are relatively constant for common scenes as the illumination becomes smoother after each reflection. The conditions can thus be satisfied in both cases by choosing an appropriate ambient radiance L_a. The value can be conservatively small if necessary; a value of 0 is equivalent to not using a control variate at all. A more sophisticated approach could be to apply different values for different points and directions. The improvements that are proposed in the next section do exactly this, based in additional information about the illumination.

4.1.7 Improved Importance Sampling and Control Variates

In [67] we have presented further improvements to the importance sampling and control variates of Sections 4.1.3 and 4.1.6. These variance reduction techniques are typically based on the local reflectance characteristics of the surfaces, which represent an important factor of the integrand. The field radiance, which is the other factor in the integrand, is simply approximated as a constant. More detailed information about the field radiance could therefore help to improve the importance sampling and the control variates.

Information about the field radiance throughout the scene has to be stored in an appropriate data structure. Various data structures to store information about the illumination in a scene have been proposed in the literature. Boundary element methods such as the radiosity method store information for each of the elements of the discretised surfaces. The information is usually restricted to the diffuse component of the illumination, although extensions to store directional information as well have been proposed, e.g. by discretising the hemisphere [49] or by using spherical harmonics [116]. Ward [134, 133] uses an octree that is independent from the surfaces to store diffuse illumination along with information about the gradients. Environment mapping techniques such as the one proposed by Reinhard *et al.* [91] store approximate incoming radiance values for a set of discrete directions. Finite element methods for rendering participating media discretise the space into regular grids [82, 70, 76] or hierarchical structures [115], mostly without storing directional information.

We propose a 5D tree which is a straightforward extension of the 1D binary tree, the 2D quadtree, the 3D octree, etc. The dimensions are the three regular spatial dimensions and two additional angular dimensions. Arvo and Kirk [5] already presented a similar tree structure in the context of accelerating ray intersection tests. For convenience a 5D point corresponding to a point (x, y, z) and a direction (θ, ϕ) is represented in coordinates that are normalised to lie between 0 and 1:

$$(x_n, y_n, z_n, s_n, t_n)$$

where:

$$\begin{aligned}
x_n &= (x - x_{min})/(x_{max} - x_{min}), \\
y_n &= (y - y_{min})/(y_{max} - y_{min}), \\
z_n &= (z - z_{min})/(z_{max} - z_{min}), \\
s_n &= (\cos\theta + 1)/2, \\
t_n &= \phi/(2\pi)
\end{aligned}$$

This parametrisation ensures a uniform subdivision: equally sized 5D volumes at any 5D position correspond to equal 3D volumes and equal spatial angles. The branching factor of the tree is $2^5 = 32$. It may seem that this would give rise to an explosive growth of the storage requirements, but it is not a problem in practice as only nodes that are effectively needed to store values are created.

The data stored in the tree are averaged field radiance values. Throughout the simulation, the path tracing algorithm estimates field radiance values at different levels of the recursion. Each estimate is averaged with the values already stored in the appropriate nodes of the tree. Whenever the number of estimates averaged in a node exceeds a given maximum threshold the node is split; a new node is created if necessary. The building of the tree is in a sense importance-driven; more detailed representations are constructed in regions where more rays are traced.

The best estimate for any point and direction can be retrieved from the tree at any time by descending to the lowest node corresponding to the given coordinates. We impose a minimum threshold on the number of averaged values for the estimate to be reliable. While the representation is not as accurate as representations that are based in the geometry of the surfaces we now avoid any meshing problems. The tree automatically makes abstraction of the complexity of the scene. Large groups of small geometric and optical details (e.g. a desk cluttered with books, pencils and paper clips) are treated as a whole by storing only averaged incoming radiance values. The tree is simple to maintain, adaptive and versatile.

The information can now be used to construct PDFs for importance sampling and control variates. The original normalised PDFs $p_i(\Theta_x)$ of Eq. (4.3), on the basis of the BRDFs, implicitly entail a transformation to a different parameter space, as discussed in general in Section 3.4. We will write this parameter space explicitly using $(\xi_1, \xi_2) \in [0, 1] \times [0, 1]$. Rewriting the rendering equation:

$$\begin{aligned}
L(x, \Theta_x) &= L_e(x, \Theta_x) + \int_{\Omega_x^{-1}} L(y, \Theta_y) f_r(x, \Theta_y, \Theta_x) \left|\Theta_y \cdot N_x\right| d\omega_y \\
&= L_e(x, \Theta_x) + P \int_{\Omega_x^{-1}} L(y, \Theta_y) \frac{f_r(x, \Theta_y, \Theta_x) \left|\Theta_y \cdot N_x\right|}{P\, p(\Theta_y)} p(\Theta_y) d\omega_y
\end{aligned}$$

$$= \quad L_e(x,\Theta_x) + \rho(x,\Theta_x) \int_0^1 \int_0^1 L(y,\Theta_y) d\xi_1 d\xi_2$$

where Θ_y now depends on ξ_1 and ξ_2. Sampling according to the original PDFs corresponds to sampling uniformly in this new parameter space. In order to take into account the additional information about the field radiance we construct an additional non-uniform PDF and a control variate in this parameter space. For this purpose we resample the stored field radiance values on a regular grid. This yields a piece-wise constant approximation $\tilde{L}(\xi_1,\xi_2)$ of the field radiance. The approximation can then be used as follows to construct a PDF:

$$p'(\xi_1,\xi_2) = \frac{\tilde{L}(\xi_1,\xi_2)}{\tilde{L}}, \tag{4.10}$$

where:

- $\tilde{L} = \int_0^1 \int_0^1 \tilde{L}(\xi_1,\xi_2) d\xi_1 d\xi_2$, the normalisation factor for the PDF, which can be computed analytically because it is a piece-wise constant function.

The net effect of this importance sampling strategy is that more rays are sent in highly reflective directions, as before, and now also in bright directions. Both factors can be important for the resulting radiance estimates. If this process is repeated at each step of the recursion, with the same Russian roulette strategy as before, based on the reflectivity, the entire estimator for the flux becomes

$$\langle\Phi\rangle_{path,betterimp} = W \sum_{i=0}^{l} \frac{\tilde{L}_i}{\tilde{L}_i(\xi_{1,i},\xi_{2,i})} L_e(x_i,\Theta_{x_i}). \tag{4.11}$$

The expression can be adapted to include next event estimation, similarly to Eq. (4.6).

Jensen [50] independently developed a similar technique, albeit on the basis of a different data structure. He stores all individual "photons" from a light tracing preprocessing step. During the actual simulation he retrieves all neighbouring photons to improve the importance sampling in a virtually identical way as presented here. Dutré and Willems [29] also presented a similar technique for the light tracing algorithm, which will be discussed in Section 4.2.

The piece-wise constant approximation is also useful as a control variate. With the same Russian roulette strategy as before the estimator for the flux eventually becomes

$$\begin{aligned} \langle\Phi\rangle_{path,betterimp,con} &= W \Big[L_e(x_0,\Theta_{x_0}) + \rho(x_0,\Theta_{x_0})\tilde{L}_0 \\ &\quad + \sum_{i=1}^{l} \left[\frac{\tilde{L}_i}{\tilde{L}_i(\xi_{1,i},\xi_{2,i})} [L_e(x_i,\Theta_{x_i}) - \tilde{L}_i(\xi_{1,i},\xi_{2,i})] + \rho(x_i,\Theta_{x_i})\tilde{L}_i \right] \Big] \\ &= W \left[\tilde{L}_0 + \sum_{i=0}^{l} \left[\frac{\tilde{L}_i}{\tilde{L}_i(\xi_{1,i},\xi_{2,i})} L_e(x_i,\Theta_{x_i}) - (1-\rho(x_i,\Theta_{x_i}))\tilde{L}_i \right] \right]. \end{aligned} \tag{4.12}$$

Again the expression can be adapted to include next event estimation, similarly to Eq. (4.9).

4.2 Monte Carlo Methods Applied to the Potential Equation – Light Tracing

4.2.1 Basic Algorithm

A second approach to solve the global illumination problem starts from the expression of the radiant flux in terms of potential (2.16) and the potential equation (2.18):

$$\Phi = \int_A \int_{\Omega_x} L_e(x, \Theta_x) W(x, \Theta_x) \left|\Theta_x \cdot N_x\right| d\omega_x d\mu_x, \tag{2.16}$$

$$W(x, \Theta_x) = W_e(x, \Theta_x) + \int_{\Omega_y} W(y, \Theta_y) f_r(y, \Theta_x, \Theta_y) \left|\Theta_y \cdot N_y\right| d\omega_y. \tag{2.18}$$

As before we would like to compute the flux or average radiance Φ through each of the pixels. Sampling stochastic variables x_0 and Θ_{x_0} according to some PDF $p_0(x, \Theta_x)$ over all surface points and corresponding hemispheres yields the following primary estimator for the integral of Eq. (2.16):

$$\langle\Phi\rangle_{light} = \frac{L_e(x_0, \Theta_{x_0}) \left|\Theta_{x_0} \cdot N_{x_0}\right|}{p_0(x_0, \Theta_{x_0})} \langle W(x_0, \Theta_{x_0})\rangle_{light}.$$

In this case the potential function is unknown, but the particular value $W(x_0, \Theta_{x_0})$ can be estimated by treating the potential equation (2.18) as an infinitely recursive integral. Applying Russian roulette with probability P_i at each level of the recursion and sampling each direction Θ_{x_i} according to a PDF $p_i(\Theta_x)$ over the hemisphere would yield the following primary estimator:

$$\begin{aligned} \langle\Phi\rangle_{light} &= \frac{L_e(x_0, \Theta_{x_0}) \left|\Theta_{x_0} \cdot N_{x_0}\right|}{p_0(x_0, \Theta_{x_0})} \\ &\quad \times \sum_{i=0}^{l} \left[\prod_{j=1}^{i} \frac{f_r(x_j, \Theta_{x_{j-1}}, \Theta_{x_j}) \left|\Theta_{x_j} \cdot N_{x_j}\right|}{P_j p_j(\Theta_{x_j})} \right] W_e(x_i, \Theta_{x_i}), \end{aligned} \tag{4.13}$$

where the length of the random walk $x_0, \ldots, x_{l+1}$ defines l. Each point $x_{i+1} = r(x_i, \Theta_{x_i})$ is found by tracing a ray (Fig. 4.11). The resulting generic algorithm as shown in Fig. (4.12) is entirely dual to simple path tracing. It is a light tracing algorithm where rays are shot from the light sources, again bouncing through the scene. Whenever they pass through the frustum of the pixel a contribution is added (Fig. 4.2). As such it is a so-called light *shooting* algorithm.

There are a few essential differences with path tracing though. First of all the random walks are independent of the pixels. Instead of considering each pixel in turn the pixels can therefore all be dealt with at the same time. A contribution can be added to the estimate of any of the pixels whenever a ray passes through it.

More importantly though, the measure of the regions where the self-emitted potential differs from 0 is usually much smaller than the measure of the regions where the self-emitted radiance differs from 0. Intuitively: there are far less points and directions covered by a pixel than points and directions on light sources. The hit-or-miss effect is therefore even worse; most random walks will not contribute to any pixel at all. If the camera model does not simulate depth of field the measure of the former sets is even infinitesimally small. The probability of a ray going through the pixel is then 0. The algorithm is thus quite useless for computing fluxes for pixels, unless the camera aperture is excessively large. In their system to assess the visibility of a road in a foggy atmosphere Rozé *et al.* [92] apply a light tracing algorithm. In order to obtain acceptable results they model the eye as a disc with a $0.8m$ radius.

Pattanaik [84, 83, 88, 87, 85, 89] does use an instance of this light tracing algorithm to compute fluxes leaving diffuse elements from a finite element representation. The approach is feasible in this case because a patch with its hemisphere in general occupies a larger fraction of the space of positions and directions than a pixel. Rays therefore have a larger probability of leaving any particular patch. Moreover, the union of all patches and their hemispheres spans the entire space of positions and directions that is considered. Each ray contributes to exactly one patch as a result, so that less computational effort is wasted. Similar problems for estimating the fluxes of the individual patches occur however if the patches are subdivided into very small elements.

The same optimisations applied to path tracing in Section 4.1 can also be applied to the sampling of Eq. (2.16) and Eq. (2.18). The primary optimisation for making the algorithm remotely useful is next event estimation. For the sake of completeness we will first mention the application of importance sampling however.

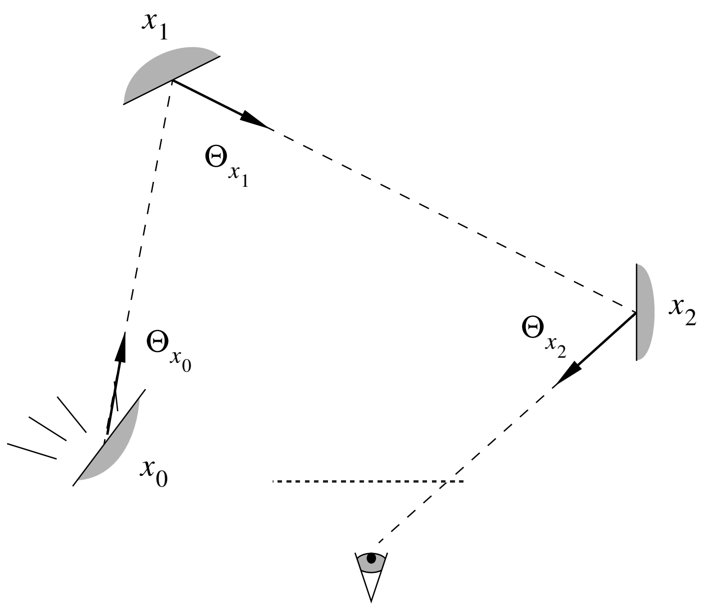


Figure 4.11: Explanation of the symbols used in the light tracing algorithm.

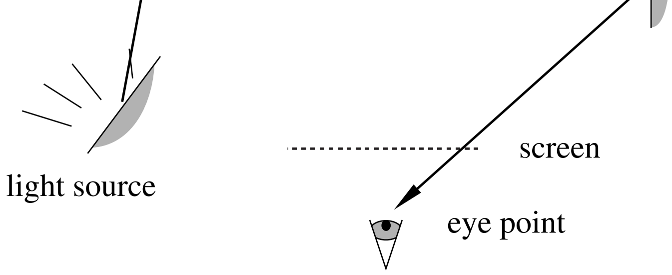


Figure 4.12: Schematic overview of the light tracing algorithm. Again the radiant flux through each pixel has to be estimated. The tracing of a primary ray from a light source corresponds to sampling the expression for the flux in terms of potential. The subsequent random walk through the scene corresponds to recursively estimating the potential values. Each time a ray passes through a pixel a contribution is added to the estimate.

4.2.2 Importance Sampling

Entirely analogous to Eq. (4.2) an appropriate PDF for sampling the stochastic variables x_0 and Θ_{x_0} in the light tracing algorithm is

$$p_0(x, \Theta_x) = \frac{L_e(x, \Theta_x)\,|\Theta_x \cdot N_x|}{L}, \tag{4.14}$$

where L is the same normalisation factor as in Eq. (4.5), the total self-emitted flux. This PDF ensures that points and directions are sampled proportionally to their self-emitted radiance. This means that more random walks will start from bright light sources and in bright directions, which are important for the illumination of the scene.

Likewise analogous to Eq. (4.3) the stochastic variables Θ_{x_i} that determine the rest of the random walk can be sampled according to the BRDF times a cosine factor. Formally the subcritical PDF equals

$$P_i\ p_i(\Theta_x) = f_r(x_i, \Theta_x, \Theta_{x_{i-1}})\,|\Theta_x \cdot N_{x_i}|\,. \tag{4.15}$$

The primary estimator (4.13) now simplifies to the dual expression of Eq. (4.4):

$$\langle \Phi \rangle_{light,imp} = L \sum_{i=0}^{l} W_e(x_i, \Theta_{x_i}). \tag{4.16}$$

Dutré and Willems suggest further improvements to the importance sampling of the light sources [28] and of the reflected directions [29]. The basic ideas are dual to the ones presented in Section 4.1.7. The optimisations send more rays in directions with a high potential. The required information about the potential is gathered and stored during the simulation. Their scheme is based on different data structures and does not take into account the BRDFs in the importance sampling however.

4.2.3 Next Event Estimation

In [27] we explain how the extension of the basic algorithm with next event estimation makes it possible to actually render images. In summary, a similar derivation as for path tracing can be made, this time starting from the potential equation (2.18):

$$\begin{aligned} W(x, \Theta_x) &= W_e(x, \Theta_x) + \int_{\Omega_y} W(y, \Theta_y) f_r(y, \Theta_x, \Theta_y)\,|\Theta_y \cdot N_y|\,d\omega_y \\ &= W_e(x, \Theta_x) + W_r(x, \Theta_x), \end{aligned}$$

where the latter term now corresponds to the potential resulting from incident potential that is reflected:

$$\begin{aligned} W_r(x, \Theta_x) &= \int_{\Omega_y} W(y, \Theta_y) f_r(y, \Theta_x, \Theta_y)\,|\Theta_y \cdot N_y|\,d\omega_y \\ &= \int_{\Omega_y} [W_e(y, \Theta_y) + W_r(y, \Theta_y)] f_r(y, \Theta_x, \Theta_y)\,|\Theta_y \cdot N_y|\,d\omega_y \\ &= \int_{\Omega_y} W_e(y, \Theta_y) f_r(y, \Theta_x, \Theta_y)\,|\Theta_y \cdot N_y|\,d\omega_y \\ &\quad + \int_{\Omega_y} W_r(y, \Theta_y) f_r(y, \Theta_x, \Theta_y)\,|\Theta_y \cdot N_y|\,d\omega_y \end{aligned}$$

$$
\begin{aligned}
&= \int_{A_{aperture}} W_e(y,\Theta_{y\to z}) f_r(y,\Theta_x,\Theta_{y\to z}) \frac{|\Theta_{y\to z}\cdot N_y|\,|\Theta_{y\to z}\cdot V|}{\|y-z\|^2} d\mu_z \\
&\quad + \int_{\Omega_y} W_r(y,\Theta_y) f_r(y,\Theta_x,\Theta_y)\,|\Theta_y \cdot N_y|\, d\omega_y,
\end{aligned}
$$

where V is the general viewing direction of the camera. For a simple camera model the point z in the former integral can be sampled uniformly over the camera aperture:

$$
p(z) = \frac{1}{A_{aperture}}. \tag{4.17}
$$

For a perfect pin-hole camera model point z is always the pin-hole or the eye point. As with simple light tracing each random walk is used to estimate the fluxes through all pixels at the same time. Determining to which pixel or pixels the random walk contributes is usually straightforward.

If all other integrals are sampled as before, according to the PDF (4.14) and subcritical PDF (4.15), the estimator (4.16) changes into

$$
\begin{aligned}
\langle \Phi \rangle_{light,imp,nextevent} &= L \Bigg[W_e(x_0,\Theta_{x_0}) + A_{aperture} \sum_{i=0}^{l} W_e(x_{i+1},\Theta_{x_{i+1}\to z}) & (4.18)\\
&\quad \times f_r(x_{i+1},\Theta_{x_i},\Theta_{x_{i+1}\to z}) & (4.19)\\
&\quad \times\, v(x_{i+1},z) \frac{\left|\Theta_{x_{i+1}\to z}\cdot N_{x_{i+1}}\right| \left|\Theta_{x_{i+1}\to z}\cdot V\right|}{\|x_{i+1}-z\|^2} \Bigg].
\end{aligned}
$$

Figure 4.13 gives an overview of the symbols used in the expressions. The algorithm as shown in Fig. 4.14 is now the dual version of path tracing with next event estimation. Next event estimation somewhat removes the effect of hit-or-miss sampling, to the extent that it is now possible to render noisy images. The effect still plays however, because the number of contributions added to each pixel is highly random. The division by $\|x_{i+1} - z\|^2$ in the estimator is less detrimental than in path tracing; the distance to the eye point is in general bounded from below so that the estimator cannot become arbitrarily large.

Apart from these problems the algorithm is only well-suited to render reflections from small directional light sources via specular surfaces on diffuse surfaces. These reflections are known as caustics. They are difficult to render with other algorithms but most suited for light tracing, as the PDFs used for importance sampling are the ones dominating the integrands in this case. Dutré and Willems [28, 29] propose alternative PDFs on the basis of information about the distribution of potential at the light sources and at each patch in a discretised scene. In his PhD dissertation [26] Dutré discusses various theoretical and practical aspects of light tracing algorithms in more detail.

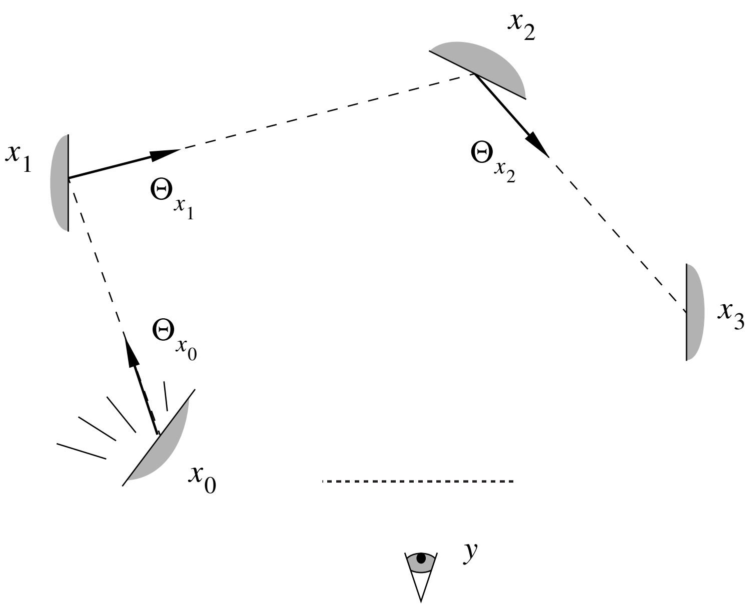


Figure 4.13: Explanation of the symbols used in the light tracing algorithm with event estimation. The random walk remains the same as without next event estimation (Fig. 4.11) but an additional point y is now sampled over the camera aperture.

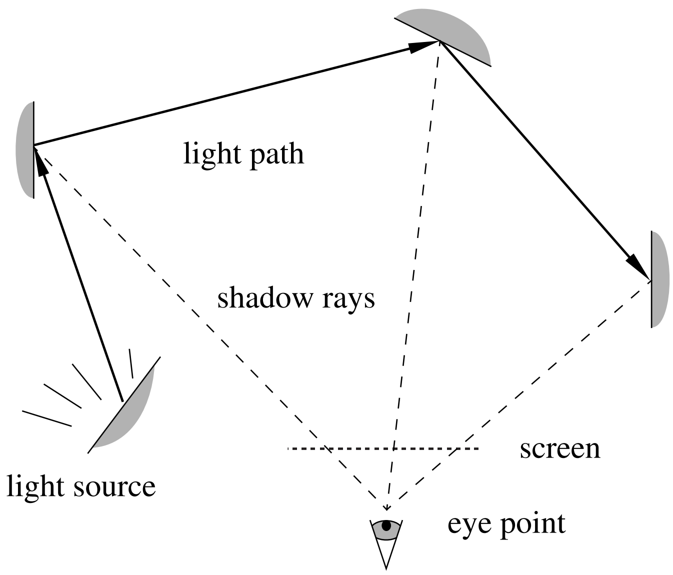


Figure 4.14: Schematic overview of the light tracing algorithm with next event estimation. Direct contributions of potential are now computed at each point on the random walk by sampling the relevant pixels, if any.

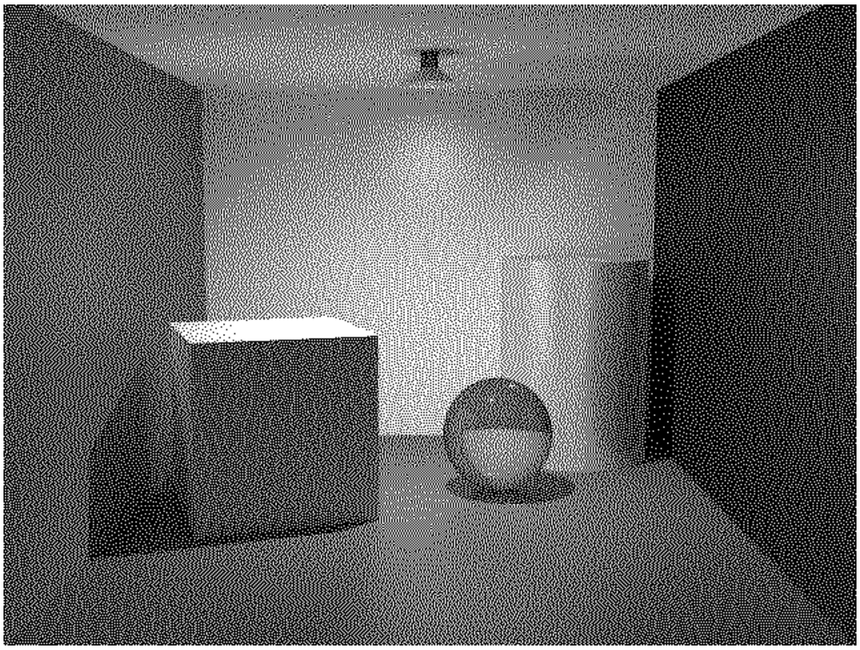


Figure 4.15: The same example scene as in Fig. 4.8. The light tracing algorithm is only really suited to render caustics on diffuse surfaces from small directional luminaires via mostly specular surfaces. Even rendering a simple scene like this one would take an impractical amount of time.

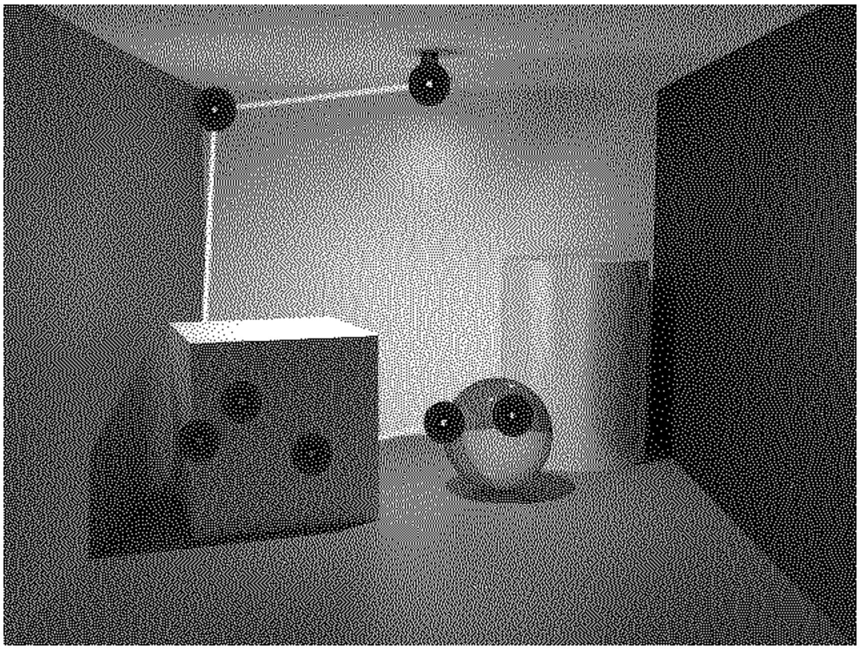


Figure 4.16: An example random walk (white rays) starting from the light source. From each intersection point a ray is cast to the eye and a contribution is added to the relevant pixel if there is no intervening object.

4.3 Monte Carlo Methods Applied to the Integral Equations of the GRDF – Bidirectional Path Tracing

4.3.1 Basic Algorithm

Section 4.1 and Section 4.2 present two alternative Monte Carlo approaches for physically-based rendering. Both the gathering and the shooting approach have their advantages and disadvantages. This begs for an algorithm that integrates both ideas and hopefully combines their strengths. In [62] we have presented a third alternative that does exactly this. In [65] we have explained its theoretical background, which we will also discuss here. The algorithm can be derived by starting from the expression of the radiant flux in terms of the global reflectance distribution function (GRDF) (2.22) and integral equations (2.23) and (2.24) defining the GRDF:

$$\begin{aligned}\Phi &= \int_A \int_{\Omega_x} \int_A \int_{\Omega_y} L_e(x,\Theta_x) W_e(y,\Theta_y) F_r(x,\Theta_x,y,\Theta_y) \\ &\qquad\qquad |\Theta_x \cdot N_x|\, |\Theta_y \cdot N_y|\, d\omega_y d\mu_y d\omega_x d\mu_x,\end{aligned} \tag{2.22}$$

$$\begin{aligned}F_r(x,\Theta_x,y,\Theta_y) &= \delta(x,\Theta_x,y,\Theta_y) \\ &\quad + \int_{\Omega_y^{-1}} f_r(y,\Theta_z,\Theta_y) F_r(x,\Theta_x,z,\Theta_z)\, |\Theta_z \cdot N_y|\, d\omega_z,\end{aligned} \tag{2.23}$$

$$\begin{aligned}F_r(x,\Theta_x,y,\Theta_y) &= \delta(x,\Theta_x,y,\Theta_y) \\ &\quad + \int_{\Omega_z} f_r(z,\Theta_x,\Theta_z) F_r(z,\Theta_z,y,\Theta_y)\, |\Theta_z \cdot N_z|\, d\omega_z.\end{aligned} \tag{2.24}$$

We would like to compute the flux or average radiance Φ through each of the pixels in turn. Suppose we sample stochastic variables x_0 and Θ_{x_0} according to some PDF $p_0(x,\Theta_x)$ and y_0 and Θ_{y_0} according to some PDF $q_0(y,\Theta_y)$ over all surface points and corresponding hemispheres. If we disregard the special nature of the Dirac impulse for a moment, these PDFs yield the following primary estimator for the flux Φ as expressed by Eq. (2.22):

$$\langle\Phi\rangle_{bipath} = \frac{L_e(x_0,\Theta_{x_0})\, |\Theta_{x_0} \cdot N_{x_0}|}{p_0(x_0,\Theta_{x_0})} \frac{W_e(y_0,\Theta_{y_0})\, |\Theta_{y_0} \cdot N_{y_0}|}{q_0(y_0,\Theta_{y_0})} \langle F_r(x_0,\Theta_{x_0},y_0,\Theta_{y_0})\rangle_{bipath}.$$

As in path tracing the sampling of a point y and direction Θ_y involves tracing a primary ray from the virtual eye point through the virtual pixel.

The GRDF is unknown, but the particular value $F_r(x_0,\Theta_{x_0},y_0,\Theta_{y_0})$ can be estimated by sampling any of the equivalent integral equations (2.23) or (2.24) recursively. We would like to use both of them at the same time, so we will combine them into a single expression. We will simply use a linear combination of them. With the operator notation of Eq. (2.25) and Eq. (2.26) and omitting the variable names this yields

$$F_r = \delta + w^* T^* F_r + w T F_r,$$

where $w + w^* = 1$. Filling in this expression recursively with different weights each time yields the following Neumann series:

$$F_r = \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} w_{ij} T^{*i} T^j \delta,$$

where the weights w_{ij} are arbitrary, as long as

$$\sum_i w_{i,N-i} = 1, \text{ for } N = 0, 1, \ldots \tag{4.20}$$

The i-th integral of operator T^* can be estimated as in light tracing: the direction Θ_{x_i} is sampled according to a PDF $p_i(\Theta_x)$ over the hemisphere with a probability P_i for Russian roulette. The j-th integral of operator T can be estimated as in path tracing: the direction Θ_{y_j} is sampled according to a PDF $q_j(\Theta_y)$ over the hemisphere with a probability Q_j for Russian roulette. Overall, the generic estimator for the flux would become

$$\begin{aligned}\langle\Phi\rangle_{bipath} \quad = \quad & \frac{L_e(x_0, \Theta_{x_0}) \left|\Theta_{x_0} \cdot N_{x_0}\right|}{p_0(x_0, \Theta_{x_0})} \frac{W_e(y_0, \Theta_{y_0}) \left|\Theta_{y_0} \cdot N_{y_0}\right|}{q_0(y_0, \Theta_{y_0})} \\ & \times \sum_{i=0}^{l^*} \sum_{j=0}^{l} w_{ij} \ \prod_{k=1}^{i} \frac{f_r(x_k, \Theta_{x_{k-1}}, \Theta_{x_k}) \left|\Theta_{x_j} \cdot N_{x_j}\right|}{P_k p_k(\Theta_{x_k})} \\ & \qquad\qquad\quad \prod_{k=1}^{j} \frac{f_r(y_{k-1}, \Theta_{y_k}, \Theta_{y_{k-1}}) \left|\Theta_{y_k} \cdot N_{y_{k-1}}\right|}{Q_k q_k(\Theta_{y_k})} \delta(x_i, \Theta_{x_i}, y_j, \Theta_{y_j}),\end{aligned}$$

where the lengths of the random walks $x_0, \ldots, x_{l^*+1}$ and $y_0, \ldots, y_l$ define l^* and l respectively. Each point $x_{i+1} = r(x_i, \Theta_{x_i})$ and each point $y_j = r(y_{j-1}, \Theta_{y_j}^{-1})$ as shown in Fig. 4.19. The probability of having a non-zero contribution is 0 though, because of the Dirac impulse. The solution lies in applying next event estimation, which evaluates each Dirac impulse at an earlier stage in the sampling process

4.3.2 Next Event Estimation

In path tracing and light tracing, next event estimation is derived by filling in the integral expressions of the corresponding equations in the right hand sides and subsequently sampling the two resulting integrals in different ways. In this case the filling in is necessary to remove the Dirac impulses. The actual evaluation does not require any additional samples anymore, because of the definitions of the Dirac impulse. Evaluating each Dirac impulse of the Neumann series at an earlier stage in the sampling process requires transformations to the appropriate parameter spaces, as with path tracing and light tracing. While the random walk remains the same, the generic estimator eventually becomes

$$\langle\Phi\rangle_{bipath,nextevent} = \sum_{i=0}^{l} \sum_{j=0}^{l^*} w_{ij} C_{ij}, \tag{4.21}$$

where a distinction has to be made between the individual contributions:

- $i = 0, j = 0$: The flux resulting from this very first term can be estimated by only using the samples y_0 and Θ_{y_0} and at this point evaluating the corresponding integral, divided by the probability density of the samples:

  $$C_{0,0} = \frac{W_e(y_0, \Theta_{y_0}) \left|\Theta_{y_0} \cdot N_{y_0}\right|}{q_0(y_0, \Theta_{y_0})} L_e(y_0, \Theta_{y_0}).$$

  From a physical point of view this term is an estimate for the flux received from a light source that is directly seen through the pixel under consideration. It is estimated in the same way as it is in path tracing.

- $i = 0, j > 0$: The fluxes resulting from these terms can be estimated by only using the samples x_0 and $y_0, \Theta_{y_0}, \ldots, \Theta_{y_{j-1}}$ and at this point evaluating the estimate. Filling in any of the two equivalent integral equations eventually yields

$$\begin{aligned} C_{0,j} &= \frac{L_e(x_0, \Theta_{x_0 \to y_j})}{p_0'(x_0)} \frac{W_e(y_0, \Theta_{y_0}) \left|\Theta_{y_0} \cdot N_{y_0}\right|}{q_0(y_0, \Theta_{y_0})} \\ &\quad \times \prod_{k=1}^{j-1} \frac{f_r(y_{k-1}, \Theta_{y_k}, \Theta_{y_{k-1}}) \left|\Theta_{y_k} \cdot N_{y_{k-1}}\right|}{Q_k q_k(\Theta_{y_k})} \\ &\quad \times f_r(y_j, \Theta_{x_0 \to y_j}, \Theta_{y_{j-1}}) v(x_0, y_{j-1}) \frac{\left|\Theta_{x_0 \to y_{j-1}} \cdot N_{x_0}\right| \left|\Theta_{x_0 \to y_{j-1}} \cdot N_{y_{j-1}}\right|}{\|x_0 - y_{j-1}\|^2}, \end{aligned}$$

where $p'(x)$ is the PDF that is explicitly or implicitly used to sample point x_0 over the light sources:

$$p'(x) = \int_{\Omega_x} p_0(x, \Theta_x) d\omega_x.$$

These terms are estimates for the flux resulting from light that is reflected j times on the eye path. They are estimated in the same way as in path tracing.

- $i > 0, j > 0$: The fluxes resulting from these terms can be estimated by using the samples $x_0, \Theta_{x_0}, \ldots, \Theta_{x_{i-1}}$ and $y_0, \Theta_{y_0}, \ldots, \Theta_{y_{j-1}}$ and at that point evaluating the estimate:

$$\begin{aligned} C_{ij} &= \frac{L_e(x_0, \Theta_{x_0}) \left|\Theta_{x_0} \cdot N_{x_0}\right|}{p_0(x_0, \Theta_{x_0})} \frac{W_e(y_0, \Theta_{y_0}) \left|\Theta_{y_0} \cdot N_{y_0}\right|}{q_0(y_0, \Theta_{y_0})} \\ &\quad \times \prod_{k=1}^{i-1} \frac{f_r(x_k, \Theta_{x_{k-1}}, \Theta_{x_k}) \left|\Theta_{x_k} \cdot N_{x_k}\right|}{P_k p_k(\Theta_{x_k})} \\ &\quad \times \prod_{k=1}^{j-1} \frac{f_r(y_{k-1}, \Theta_{y_k}, \Theta_{y_{k-1}}) \left|\Theta_{y_k} \cdot N_{y_{k-1}}\right|}{Q_k q_k(\Theta_{y_k})} \\ &\quad \times f_r(x_i, \Theta_{x_{i-1}}, \Theta_{x_i} \to y_{j-1}) f_r(y_{j-1}, \Theta_{x_i \to y_{j-1}}, \Theta_{y_{j-1}}) \\ &\quad \times v(x_i, y_{j-1}) \frac{\left|\Theta_{x_i \to y_{j-1}} \cdot N_{x_i}\right| \left|\Theta_{x_i \to y_{j-1}} \cdot N_{y_{j-1}}\right|}{\|x_i - y_{j-1}\|^2}. \end{aligned}$$

These terms are estimates for the flux resulting from light that is reflected i times on the light path and j times on the eye path.

The resulting algorithm is called bidirectional path tracing. It was introduced by us in [62], while Veach and Guibas later independently presented a similar algorithm in [125]. To our knowledge it is the first pure Monte Carlo rendering algorithm that combines the ideas of shooting light from the light sources and gathering light through the pixels.

Each term C_{ij} is a different estimate for the flux resulting from light that is reflected $i + j$ times. This provides a physical explanation for the sums of the weights w_{ij} with equal depths of reflection $i + j$ having to be 1, as expressed by Eq. (4.20).

Other estimators

While the estimators presented above are already usable there are still some alternatives left unmentioned. We will name them using negative subscripts. The sums in Eq. (4.21) have to be extended accordingly to start from -1 rather than from 0.

- $i > 0, j = -1$: The fluxes resulting from these terms can also be estimated by only using the samples $x_0, \Theta_{x_0}, \ldots, \Theta_{x_{i-1}}$ and at this point evaluating the estimate as in pure light tracing:

$$\begin{aligned} C_{i,-1} &= \frac{L_e(x_0, \Theta_{x_0}) \left|\Theta_{x_0} \cdot N_{x_0}\right|}{p_0(x_0, \Theta_{x_0})} \\ &\times \left[\prod_{k=1}^{i-1} \frac{f_r(x_k, \Theta_{x_{k-1}}, \Theta_{x_k}) \left|\Theta_{x_k} \cdot N_{x_k}\right|}{P_k p_k(\Theta_{x_k})} \right] W_e(x_{i-1}, \Theta_{x_{i-1}}). \end{aligned}$$

  As discussed in Section 4.2 these estimators are not practical for image-based rendering as hardly any rays will pass through any of the pixels.

- $i = -1, j > 0$: The fluxes resulting from these terms can also be estimated by only using the samples $y_0, \Theta_{y_0}, \ldots, \Theta_{y_j}$:

$$\begin{aligned} C_{-1,j} &= \frac{W_e(y_0, \Theta_{y_0}) \left|\Theta_{y_0} \cdot N_{y_0}\right|}{q_0(y_0, \Theta_{y_0})} \\ &\times \left[\prod_{k=1}^{j-1} \frac{f_r(y_{k-1}, \Theta_{y_k}, \Theta_{y_{k-1}}) \left|\Theta_{y_k} \cdot N_{y_{k-1}}\right|}{Q_k q_k(\Theta_{y_k})} \right] L_e(y_{j-1}, \Theta_{y_{j-1}}). \end{aligned}$$

  These estimators are those of pure path tracing. Estimator $C_{-1,1}$ actually corresponds to the $C_{0,0}$ defined earlier, which we will redefine in the next paragraph. The other estimators were missing in our previous work but were applied in [125]. As discussed in Section 4.1 this type of estimator is mostly efficient for rendering scenes with specular surfaces and large light sources.

- $i \geq 0, j = 0$: The fluxes resulting from these terms can also be estimated by only using the samples $x_0, \Theta_{x_0}, \ldots, \Theta_{x_j}$ and the point on the camera aperture y_{-1} and then evaluating the estimator:

$$C_{0,0} = \frac{L_e(x_0, \Theta_{x_0 \to y_{-1}})}{p'(x_0)} \frac{W_e(x_0, \Theta_{x_0 \to y_{-1}})}{q'(y_{-1})} v(x_0, y_{-1}) \frac{\left|\Theta_{x_0 \to y_{-1}} \cdot N_{x_0}\right| \left|\Theta_{x_0 \to y_{-1}} \cdot V\right|}{\|x_0 - y_{-1}\|^2},$$

  and for $i > 0$:

$$\begin{aligned} C_{i,0} &= \frac{L_e(x_0, \Theta_{x_0}) \left|\Theta_{x_0} \cdot N_{x_0}\right|}{p_0(x_0, \Theta_{x_0}) q'(y_{-1})} W_e(y_j, \Theta_{x_i \to y_0}) \\ &\times \prod_{k=1}^{i-1} \frac{f_r(x_k, \Theta_{x_{k-1}}, \Theta_{x_k}) \left|\Theta_{x_k} \cdot N_{x_k}\right|}{P_k p_k(\Theta_{x_k})} \\ &\times f_r(x_i, \Theta_{x_{i-1}}, \Theta_{x_i \to y_{-1}}) v(x_i, y_{-1}) \frac{\left|\Theta_{x_i \to y_{-1}} \cdot N_{x_i}\right| \left|\Theta_{x_i \to y_{-1}} \cdot V\right|}{\|x_i - y_{-1}\|^2}, \end{aligned}$$

where $q'(y)$ is the PDF that is explicitly or implicitly used to sample point y_{-1} over the camera aperture. The estimators are those of light tracing with next event estimation. As discussed in Section 4.2 they are not particularly useful in general, but well-suited to render caustics.

Figure 4.17 gives a complete overview of the alternative contributions and the samples they depend on. Figure 4.18 then gives a visual representation of the same contributions. Their respective weights must sum to 1 along the diagonal with equal numbers of reflections $i + j$. The three positions in the top left corner do not correspond to valid contributions as at least a position and a direction have to be sampled to estimate a flux. Figure 4.19 summarises the symbols used for the variables that determine the random walk. Figure 4.20 gives a schematic overview of the final algorithm as it has been implemented, i.e. without the light tracing contributions.

Comparison with Previous Algorithms

The algorithms discussed in the previous sections select a single contribution from each diagonal of Figure 4.19. Pure path tracing corresponds to the contributions from the first column ($w_{i,j} = 1$ for $i = -1$, and 0 otherwise). Path tracing with next event estimation corresponds to the second column, except for the first element, which estimates direct illumination. It is replaced by the first element from the first column ($w_{-1,1} = 1$, $w_{i,j} = 1$ for $i = 0, j > 0$, and 0 for the other weights). Pure light tracing corresponds to the first row ($w_{i,j} = 1$ for $j = -1$, 0 otherwise). Light tracing with next event estimation corresponds to the second row ($w_{i,j} = 1$ for $j = 0$, 0 otherwise).

The potential strength of bidirectional path tracing lies in handling scenes with complex indirect illumination. A pure gathering approach performs best if the light sources are directed to large parts of the scene, even if the viewer only looks at a small part of the scene. The eye paths originate at the eye point and thus easily find their way to the light sources. A pure shooting approach on the other hand works best if the viewer sees most of the scene, even if the light sources are directed away. The light paths originate at the light sources and can then bounce through the scene to contribute to pixels of the screen. The integrated shooting and gathering approach of bidirectional path tracing combines these advantages. The light paths and the eye paths originate at the important ends of the illumination transport chain and meet halfway. This is specifically interesting for complex illumination situations where the light sources illuminate the scene indirectly and the viewer sees only a part of the scene. Figure 4.21 provides an example scene, for which Fig. 4.22 visualises a light path and an eye path and the corresponding shadow rays. Such a pair of random walks makes up a primary estimate in our implementation; the secondary estimator averages the results of many of these pairs. Alternatively, one could link a set of several light paths with a set of several eye paths, which would still be unbiased.

Note that bidirectional path tracing puts more computational effort in estimating indirect illumination by nature. The deeper rays on both paths and especially the additional shadow rays only serve this purpose. In path tracing and light tracing a larger fraction of the traced rays caters for lower levels of reflection. The shadow rays of next event estimation already put a greater emphasis on indirect illumination. For any of the algorithms even more effort can be put in the deeper rays, by choosing higher probabilities for reflection in the Russian roulette. Pattanaik notes that the results of his light tracing algorithm improve by applying this *absorption suppression* technique [83, p. 91]. This effect is related to the question of how many rays should be traced at each reflection. Each of the recursive integrals can be estimated by any number of samples. Arvo and Kirk [6] already mention that it is difficult to determine an optimal number a priori. In many practical scenes sampling the objects seen through a pixel and the camera aperture is a much simpler problem than sampling reflections over the hemisphere. This would suggest that a tree of samples would be more effective than a simple random walk. For scenes containing a large number of tiny objects or complex textures on the other hand the primary rays may again be of great importance for sampling the pixels adequately. The algorithms presented here therefore stick to the most natural solution of single random walks with Russian roulette based on the directional-hemispherical reflectance. Further research could present heuristics or rules to improve this aspect.

As in the path tracing and light tracing algorithms the next event estimation introduces a division by $\|x_i - y_{j-1}\|^2$ into some of the contributions. As before, this is an unfavourable situation, since they can become arbitrarily large as a result. In path tracing the position of the light sources in a particular scene can bound the factor. In light tracing the viewing parameters will even bound the factor in most practical cases. In bidirectional path tracing the effect can occur for a pair of paths anywhere throughout the scene, e.g. if the eye path and the light path almost meet in a corner. It is therefore unavoidable for most practical scenes. As with the other algorithms an appropriate combination of the various estimators largely solves this problem, which will be discussed in Section 4.3.4.

Other Bidirectional Algorithms

There have been many earlier attempts to combine the ideas of shooting and gathering light. The fundamental difference with our solution lies in the way the illumination transport paths are linked. Earlier algorithms invariably store the results from a shooting step in some kind of data structure; they only use these results in a subsequent gathering step. This approach yields a host of two-pass or multi-pass algorithms. An early example is *backward ray tracing*, introduced by Arvo [1] in '86. It has evolved into various bidirectional or multi-pass ray tracing algorithms, e.g. presented by Heckbert [43], by Chen *et al.* [11] and by Collins [18]. They all use different types of data structures to store the diffuse illumination from a stochastic light shooting step. Jensen en Christensen [52] and Shirley *et al.* [111] store all possible information from such a step, but only to perform density estimation afterwards. As with the other techniques, this smoothes out the illumination. Source selection methods as proposed by Kok and Jansen [58, 60] and similar final rendering steps, e.g. [93, 11], gather light from a radiosity solution, which is often computed in a shooting step. Again, the information stored is either diffuse or coarsely directional, because of storage considerations. The accuracy of the final results depends not only on the Monte Carlo simulation, but also on the granularity of the data structures. The algorithms are therefore biased from a strict Monte Carlo point of view.

Our bidirectional path tracing algorithm does not use any such data structures. For each sample it immediately links the original light path and eye path. Working with the original, unfiltered samples thus ensures that the method remains unbiased.

Reducing the Number of Shadow Rays

In [68] we have shown how the large number of shadow rays can be reduced by introducing an additional Russian roulette process. This is an extension to the principle presented by Shirley [106, 107] and by Shirley and Wang [112]. The lighting contributions typically vary over several orders of magnitude. Yet the algorithm traces a shadow ray for each contribution, representing the same amount of work. In order to spend less computational effort on smaller contributions and more effort on larger contributions, shadow rays can be grouped. By stochastically selecting the more important contributions and only tracing their corresponding shadow rays, the number of shadow rays can be reduced, without increasing the variance to the same degree. For simplicity we have not applied this technique for producing the results of Chapter 5.

Contributions that depend on …		samples on the light path up to …				
			x_0	Θ_{x_0}	Θ_{x_1}	Θ_{x_2}
				$C_{1,-1}$	$C_{2,-1}$	$C_{3,-1}$
samples on	y_{-1}		$C_{0,0}$	$C_{1,0}$	$C_{2,0}$	$C_{3,0}$
the eye path	Θ_{y_0}	$C_{-1,1}$	$C_{0,1}$	$C_{1,1}$	$C_{2,1}$	$C_{3,1}$
up to …	Θ_{y_1}	$C_{-1,2}$	$C_{0,2}$	$C_{1,2}$	$C_{2,2}$	$C_{3,2}$

Figure 4.17: An overview of the stochastic samples on the light path and the eye path and their corresponding illumination contributions in bidirectional path tracing.

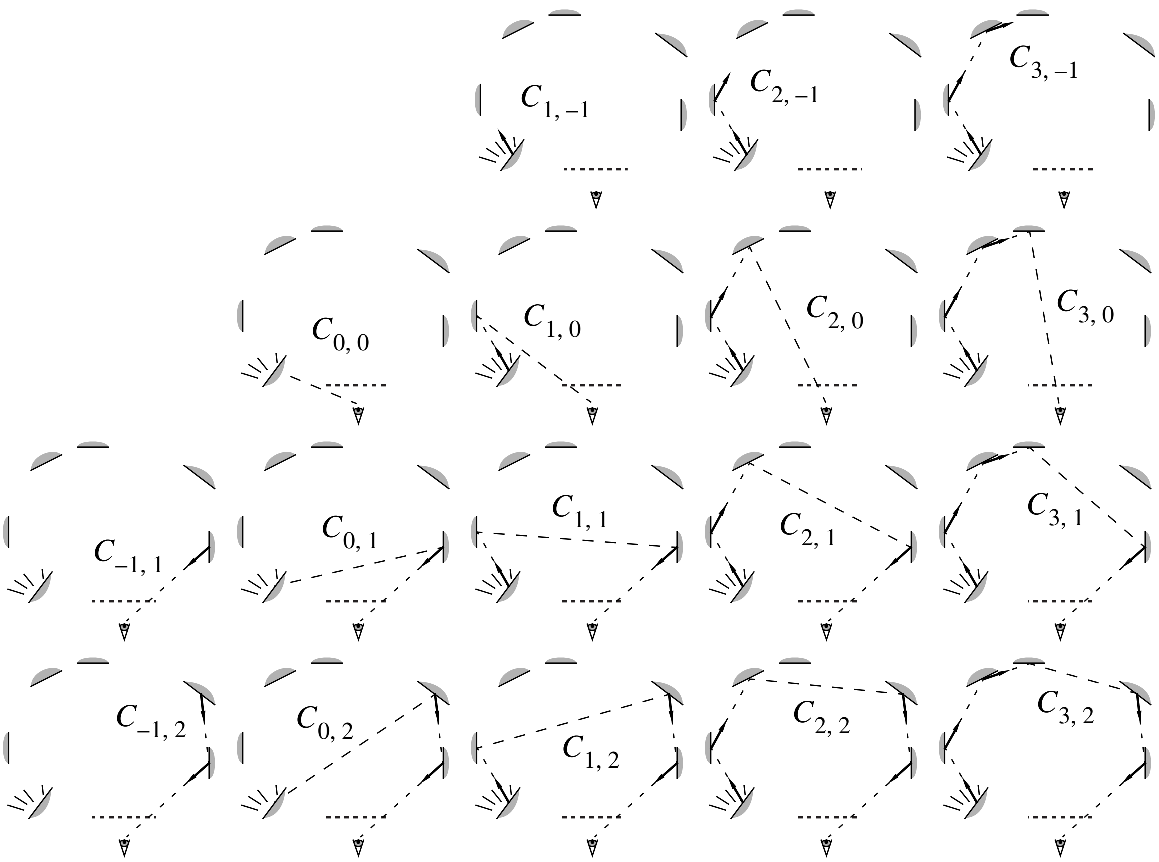


Figure 4.18: A schematic representation of the contributions. The left-most column corresponds to pure path tracing. The first column (except for the first contribution) corresponds to path tracing with next event estimation. The first row corresponds to pure light tracing, which is fairly useless in this context. The second row corresponds to light tracing with next event estimation. The other contributions are unique to bidirectional path tracing.

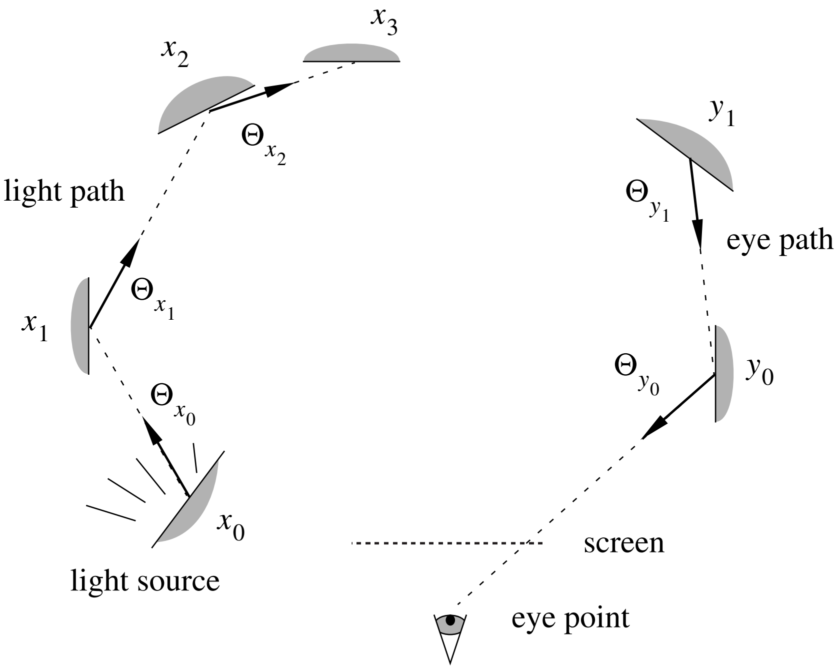


Figure 4.19: Explanation of the symbols used in the bidirectional path tracing algorithm.

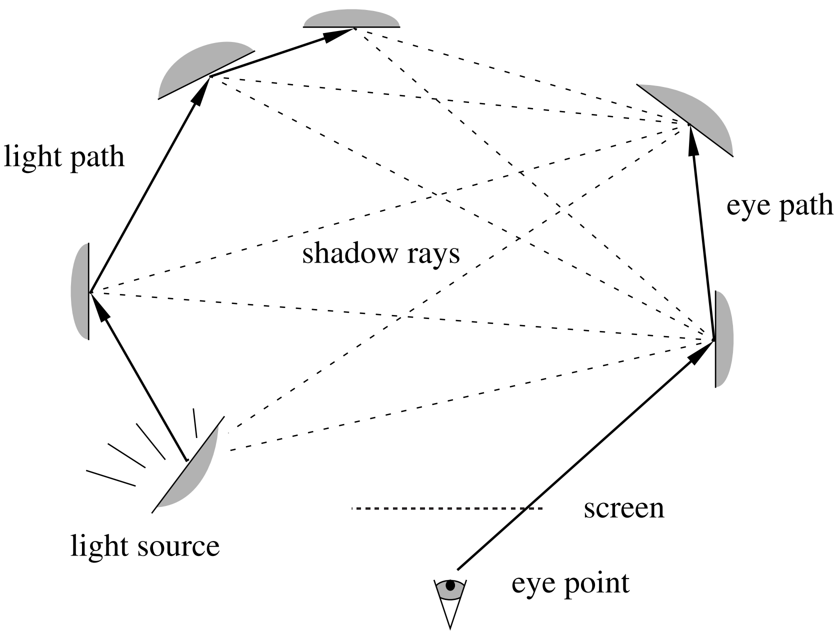


Figure 4.20: Schematic overview of the bidirectional path tracing algorithm. The flux through a pixel is estimated by tracing a random walk from a light source and a random walk from the eye point and by then casting shadow rays between points on the respective paths. A contribution is added for each shadow ray that does not hit any objects.

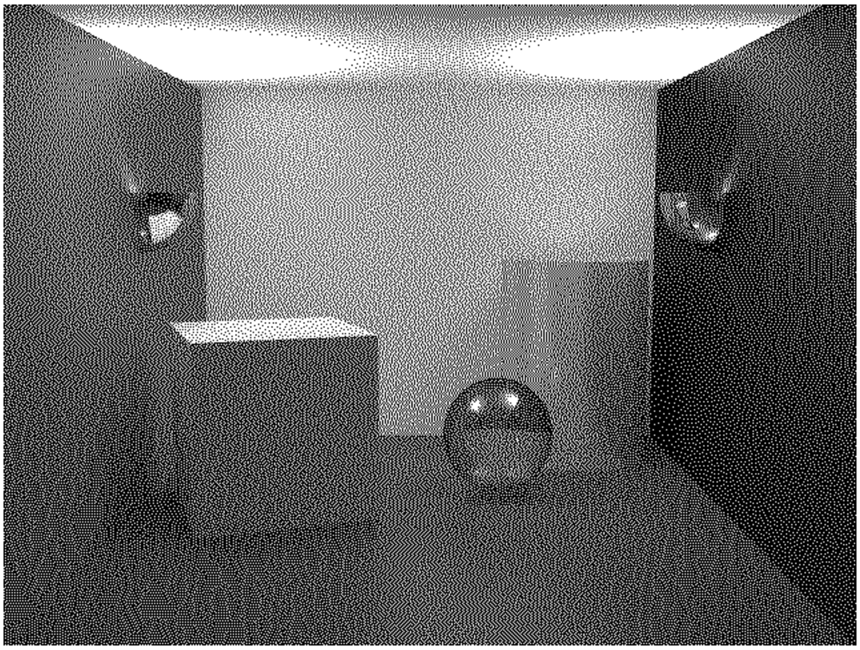


Figure 4.21: An example scene rendered with the bidirectional path tracing algorithm. The algorithm is best suited to render scenes with complex and mostly indirect illumination. In this case the scene is illuminated by lamp shades against the walls.

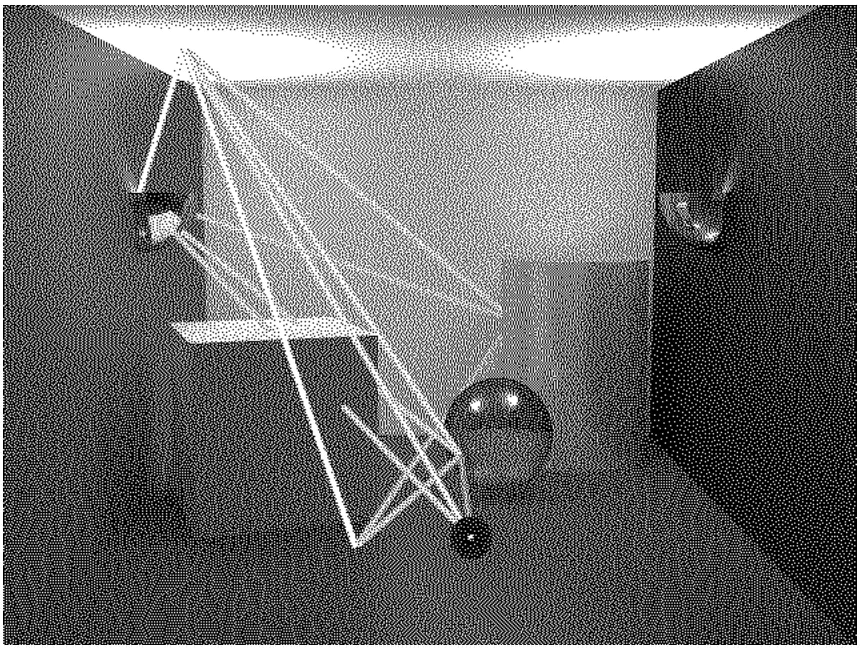


Figure 4.22: An example light path starting from a light source, bouncing to the ceiling and to the floor (white rays) and an eye path going through a pixel in the lower part of the image (blue rays). All intersection points on the respective paths are connected by means of shadow rays. Shadow rays that do not hit any objects (green rays) add an illumination contribution to the pixel. Shadow rays that do hit an object (red rays) do not add a contribution.

4.3.3 Importance Sampling

Clearly the same basic importance sampling techniques as applied for path tracing and for light tracing are suitable for bidirectional path tracing. On the basis of the local emittance and reflectance information the stochastic variables $x_0, \Theta_{x_0}, \Theta_{x_1}, \ldots$ that determine the light path are ideally sampled according to the PDFs (4.14) and (4.15), as in light tracing. Likewise the stochastic variables $y_0, \Theta_{y_0}, \Theta_{y_1}, \ldots$ that determine the eye path are ideally sampled according to the PDFs (4.2) and (4.3), as in path tracing. The resulting estimator is

$$\langle \Phi \rangle_{bipath,nextevent,imp} = \sum_{i=-1}^{l} \sum_{j=-1}^{l^*} w_{ij} C'_{ij}, \tag{4.22}$$

where the relevant contributions simplify to

- $i = -1, j > 0$:

$$C_{-1,j} = L_e(y_j, \Theta_{y_j}).$$

- $i = 0, j > 0$:

$$\begin{aligned} C'_{0,j} &= \frac{L_e(x_0, \Theta_{x_0 \to y_j})}{L(x_0)} W_e(y_j, \Theta_{y_{j-1}}) \\ &\quad \times f_r(y_j, \Theta_{x_0 \to y_j}, \Theta_{y_{j-1}}) v(x_0, y_{j-1}) \frac{\left|\Theta_{x_0 \to y_{j-1}} \cdot N_{x_0}\right| \left|\Theta_{x_0 \to y_{j-1}} \cdot N_{y_{j-1}}\right|}{\|x_0 - y_{j-1}\|^2}. \end{aligned}$$

- $i > 0, j > 0$:

$$\begin{aligned} C'_{ij} &= f_r(x_i, \Theta_{x_{i-1}}, \Theta_{x_i} \to y_{j-1}) f_r(y_{j-1}, \Theta_{x_i \to y_{j-1}}, \Theta_{y_{j-1}}) \\ &\quad \times v(x_i, y_{j-1}) \frac{\left|\Theta_{x_i \to y_{j-1}} \cdot N_{x_i}\right| \left|\Theta_{x_i \to y_{j-1}} \cdot N_{y_{j-1}}\right|}{\|x_i - y_{j-1}\|^2}. \end{aligned}$$

- $i = 0, j = 0$:

$$C'_{0,0} = \frac{L_e(x_0, \Theta_{x_0 \to y_{-1}})}{L(x_0)} \frac{W_e(x_0, \Theta_{x_0 \to y_{-1}})}{q'(y_{-1})} v(x_0, y_{-1}) \frac{\left|\Theta_{x_0 \to y_{-1}} \cdot N_{x_0}\right| \left|\Theta_{x_0 \to y_{-1}} \cdot V\right|}{\|x_0 - y_{-1}\|^2}.$$

- $i > 0, j = 0$:

$$\begin{aligned} C'_{i,0} &= W_e(y_j, \Theta_{x_i \to y_0}) \\ &\quad \times f_r(x_i, \Theta_{x_{i-1}}, \Theta_{x_i \to y_{-1}}) v(x_i, y_{-1}) \frac{\left|\Theta_{x_i \to y_{-1}} \cdot N_{x_i}\right| \left|\Theta_{x_i \to y_{-1}} \cdot V\right|}{\|x_i - y_{-1}\|^2}. \end{aligned}$$

The expressions obviously only present an ideal situation. As mentioned before it may not always be perfectly possible to select samples according to the suggested PDFs. In that case the more general estimator (4.21) has to be applied as an approximation of the above ideal.

4.3.4 Combining Estimators

Section 4.3.2 presented different sets of alternative estimators for the different contributions to the flux. Any properly weighted sum yields an unbiased estimator for the radiant flux. We already mentioned that the various path tracing and light tracing algorithms can be regarded as specific instantiations of the generic bidirectional path tracing algorithm by choosing simple sets of weights.

These weights do not fully exploit the potential of bidirectional path tracing however. In [62, 65] we presented some heuristics to compute weights on the basis of the reflectance characteristics of the surfaces on the eye path. Intuitively one can imagine that directions sampled at specular surfaces are more reliable than directions sampled at diffuse surfaces, because they are sampled according to a BRDF that determines the shape of the integrand. One can take this observation into account by relatively increasing the weights of paths reflecting off specular surfaces. The resulting estimators proved to have a lower variance than previous estimators. Especially images of scenes with complex illumination contain visibly less noise. Even though the estimators are unbiased, the heuristics lack a certain elegance and symmetry, and most of all some more fundamental theoretical support.

The heuristics presented by Veach and Guibas [126], which we have discussed in the context of general Monte Carlo methods in Section 3.5, offer better theoretical and practical qualities. Each contribution C_{ij} corresponding to an illumination transport path $x_0, \ldots, x_i, y_{j-1}, \ldots, y_{-1}$ can be regarded as a single sample with a given probability density. The probability density $p_{ij}(x_0, \ldots, x_i, y_{j-1}, \ldots, y_{-1})$ of this sample can be computed as the product of the probability densities of the individual points, in the correct parameter space. Without writing out the expressions in full the estimator based on the balance heuristic looks like

$$\langle \Phi \rangle_{bipath,nextevent,comb} = \sum_{i=0}^{l} \sum_{j=0}^{l^*} w_{ij} C_{ij}, \tag{4.23}$$

where

$$w_{ij} = \frac{p_{ij}(x_0, \ldots, x_i, y_{j-1}, \ldots, y_{-1})}{\sum_{k=0}^{l} \sum_{m=0}^{l^*} \underset{k+m=i+j}{} p_{km}(x_0, \ldots, x_i, y_{j-1}, \ldots, y_{-1})}.$$

Each weight of the balance heuristic expresses the ratio between

- the probability density of the sampling strategy with which the transport path was actually constructed,

and

- the sum of the a posteriori probability densities of the alternative strategies, with which the path could have been constructed.

Contributions of paths that are relatively improbable get low weights as a result and vice versa. As with the combined estimator (4.7) for path tracing the written out contributions C_{ij} times their weights w_{ij} have sums of factors containing squared distances in the denominator, instead of a single factor $\|x_i - y_{j-1}\|^2$. They are also bounded in most practical cases as a result, which is a valuable property.

4.4 Summary

This chapter has shown how Monte Carlo rendering algorithms are nothing but the standard Monte Carlo methods, explained in Chapter 3, applied to the different mathematical models for the global illumination problem, explained in Chapter 2. Even though the theoretical models are mathematically equivalent, straightforward Monte Carlo evaluation of their integrals gives rise to vastly different algorithms.

The rendering equation naturally leads to stochastic ray tracing, or more specifically to path tracing. The algorithm estimates the flux through each pixel in turn, starting from the expression of the flux in terms of radiance. For this purpose it first samples the surfaces and directions seen through the pixel. It then recursively estimates the integral of the rendering equation by sampling reflection directions. Each time a point and direction on a light source are selected a contribution is added to the estimate. The algorithm therefore only works efficiently if the light sources are large compared to the rest of the scene.

Several variance reduction techniques, some of which were already applied in the earliest empirical ray tracing algorithms, can further improve the efficiency of the estimators. Various stratified sampling techniques attempt to spread the samples more evenly over the aperture time, the lens area, the surface areas and the hemispheres. Importance sampling is mainly applied to select directions according to the reflective characteristics of the surfaces. This is a great gain for efficiently simulating specular reflections. If the light sources are relatively small, as in most common scenes, next event estimation of the integral equations greatly improves the efficiency, by sampling the light sources separately at each step of the recursion. It provides an additional estimator to which the theory of combining estimators can be applied. We have introduced simple control variates to reduce the variance of the estimator further. We have then proposed a data structure to store more detailed information about illumination in the scene. The information is useful to construct improved PDFs and control variates.

The potential equation, adjoint to the rendering equation, naturally leads to a light tracing algorithm. It is dual to path tracing. Starting from the expression for the flux in terms of potential, it samples points and directions at light sources. It then recursively estimates the integral equation by sampling the reflection directions. Each time a point and direction going through a pixel are sampled a contribution is added to the estimate. The basic algorithm is only practical for computing the fluxes of patches, not for the fluxes of pixels, as far too few samples would contribute to a pixel. Next event estimation helps a little by computing a direct contribution to a pixel at each point on the random walk. The algorithm still has a large variance, but it may be useful for computing caustics.

The global reflectance distribution function and its integral equations give rise to a new algorithm, which we have called bidirectional path tracing. It combines the ideas of shooting and gathering light from the previous algorithms. The light path and the eye path are random walks that are created as in light tracing and in path tracing respectively. Connecting the points on the random walks by means of shadow rays yields a set of estimators for the different parts of the illumination transport. The estimators can be weighted based on their expected qualities. The light path and the eye path already trace the most important parts of the illumination transport. The sets of alternative estimators are therefore more flexible in handling complex illumination situations than the single estimators of previous algorithms. Practical tests in Chapter 5 will show that the algorithm has a markedly lower variance when rendering scenes with a lot of indirect illumination.

Chapter 5

Test Results

In this chapter we will discuss some practical results we have obtained with the algorithms of Chapter 4. We will examine the influence of the various variance reduction techniques and of the different algorithms for a set of example scenes. The example scenes are selected specifically so as to span a wide range of illumination conditions.

5.1 Implementation

We have implemented the different variants of path tracing and bidirectional path tracing. The program computes and outputs images one pixel at a time. It therefore does not include the light tracing estimator, for which the entire image would need to be stored. The implementation is based on the publicly available ray tracer *Rayshade*, of which the set of input and output routines and the optimised ray intersection routines are used as a black box. These routines can handle various geometric primitives and more complex constructive solid geometry.

The program furthermore supports area light sources and spot light sources. All area light sources are perfect Lambertian emitters, while the spot lights are point light sources with a Phong-like angular distribution. The BRDFs are modelled as modified Phong BRDFs that are physically plausible [72, 73, 66]. Image texture maps and procedural texture maps make it possible to add small optical details.

The camera model computes average radiance values (cfr. Appendix A.2). It allows for a finite camera aperture, which enables to simulate depth of field, and for a finite exposure time, which enables to simulate motion blur.

For our tests we have rendered images at a resolution of 100×100 pixels. They are stored in the 4-byte "RGB+common exponent" format introduced by Ward [130]. We have rendered reference images of all scenes using 5,000 or 10,000 samples per pixel. In the following sections we will compare all results with these reference images, on the basis of the RMS error:

$$\text{RMS} = \sqrt{\frac{\sum_{pixels\ p}[\overline{L}_{ref,p} - \overline{L}_p]^2}{N}}$$

where N is the number of pixels in the images. The RMS error does not reflect the visual quality of the image very well, because the latter is largely subjective and therefore difficult to measure. Rushmeier *et al.* [96] present some alternative perceptually based metrics to compare synthetic and real images. The RMS error does however present a suitable measure for the variance of the Monte Carlo algorithms, by averaging the errors of the pixels over the image.

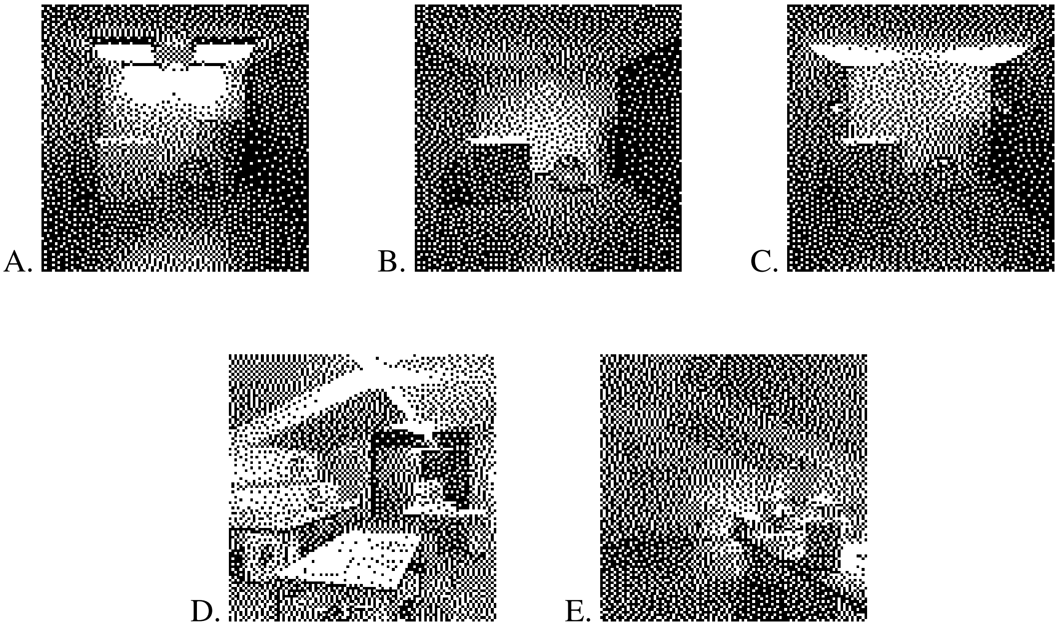


Figure 5.1: The test scenes used to evaluate the variance reduction techniques and different algorithms in practice.

5.2 Test Scenes

Figure 5.1 shows the example scenes with which we have experimented. Each scene has been chosen to represent a different illumination situation. While they may be representative for a large number of scenes, they are doubtlessly *not* representative for an even larger number of scenes. The test results should be considered as such.

A. **Test scene, illuminated directly by light boxes.** The scene loosely resembles the famous Cornell box, which was introduced in [34, 16] and has been popular as a test scene since. In this case the walls and the ceiling are mostly diffuse and the floor has a glossy reflectance. The objects are a diffuse cube, a transparent sphere and a slightly specular cylinder. The scene is illuminated by four light boxes against the ceiling. This scene was also used to illustrate pure path tracing in Fig. 4.3.

B. **Test scene, illuminated directly by a spot light.** The same scene, now illuminated by a single spot light source against the ceiling. It was also used to illustrate path tracing with next event estimation in Fig. 4.8.

C. **Test scene, illuminated indirectly by lamp shades.** The same scene, now illuminated by two chrome lamp shades against the left wall and the right wall respectively. The illumination is mostly indirect by light reflected diffusely off the ceiling. Because of this the scene was also used to illustrate bidirectional path tracing in Fig. 4.21.

D. **Office scene, illuminated by neon lighting.** A view of our office K00.10 at night. Only the overhead neon lights are lit. This is a complex scene, containing 4 area light sources and over a 1000 primitives with various reflectance characteristics and textures. Most of the room is illuminated directly. Figure 5.2 shows a larger view of the scene.

E. **Office scene, illuminated by spot lights.** A different view of the same scene, now illuminated by the 4 spot lights. The illumination of a large part of the room is indirect. Figure 5.3 shows a larger view of the scene.

The scenes mostly have diffuse and glossy surfaces, which are in general difficult to render with ray tracing-like methods. Unless specified otherwise the tests have been performed using the same algorithms for each given scene:

A. Pure path tracing,

B. Path tracing with next event estimation and combined estimators,

C. Bidirectional path tracing,

D. Path tracing with next event estimation and combined estimators,

E. Bidirectional path tracing.

By default, i.e. except for testing the optimisations themselves, stratified sampling, Russian roulette and importance sampling according to the subcritical BRDFs discussed in Chapter 4 have been applied, but no control variates.

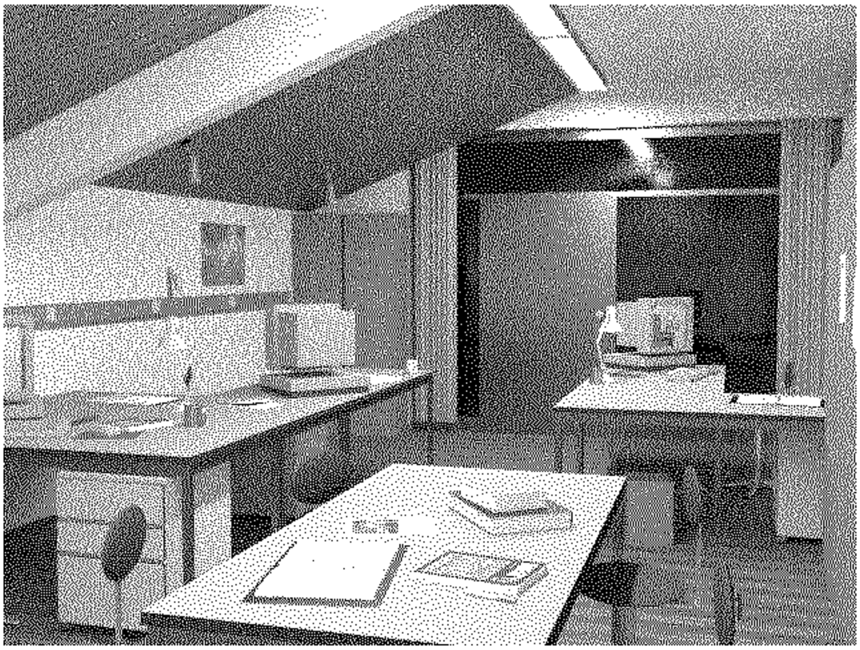


Figure 5.2: A larger view of test scene D, the office scene illuminated by neon lighting.

Figure 5.3: A larger view of test scene E, the office scene illuminated by spot lights.

5.3 Stratified Sampling

We have rendered all test scenes with the algorithms mentioned above, both without stratified sampling and with stratified sampling. As discussed in Sections 4.1.2, we have chosen for N-rooks sampling because of its flexibility. The overhead for generating the strata is small. It consists mainly of generating the permutations for the dimensions, which we have done in a preprocessing step.

The sample rates in our experiments range from 10 samples per pixel up to a 1,000 samples per pixel. Figure 5.4 shows the RMS errors of test scene A on a linear scale, while Fig. 5.5 shows them on a logarithmic scale. The standard deviations of ordinary secondary estimators always decrease as $O(1/\sqrt{N})$, as indicated by Eq. (3.5). The plots of the RMS errors of the test images have a similar shape, since the images are large enough to average out most of the stochastic uncertainty on the errors. Note that a reduction of the standard deviation or of the RMS error by 10% for the same number of samples corresponds to a reduction of the variance by almost 20%. Looking at it from a different point of view, the same results can then be obtained using 20% less samples.

Equation (3.7), which expresses the variance of a secondary estimator with stratified sampling, shows that the decrease of the variance with an increasing number of samples depends on the integrand. In practice the RMS errors have a similar $O(1/\sqrt{N})$ profile, as can be seen in Figures 5.4 and 5.5. Stratified sampling is guaranteed to have a lower variance. It therefore produces lower RMS errors on average, although the RMS error of specific images may sometimes be higher due to the stochastic uncertainty. Table 5.1 presents an overview of the improvements thanks to stratified sampling for the different test scenes. The RMS errors are reduced by 10-20% on average. For test scene C the reduction is larger, probably because stratified sampling selects between the two light sources in a more balanced way.

Test scene	Reduction of the RMS errors
A	10 – 15 %
B	0 – 25 %
C	25 – 50 %
D	0 – 15 %
E	10 – 20 %

Table 5.1: Overview of the reduction of the RMS errors due to stratified sampling, for the same numbers of samples. The different test scenes have been rendered with the specific rendering algorithms that have been listed above. The percentages indicate average improvements for the different numbers of samples. For these examples they vary between 0 and 50%.

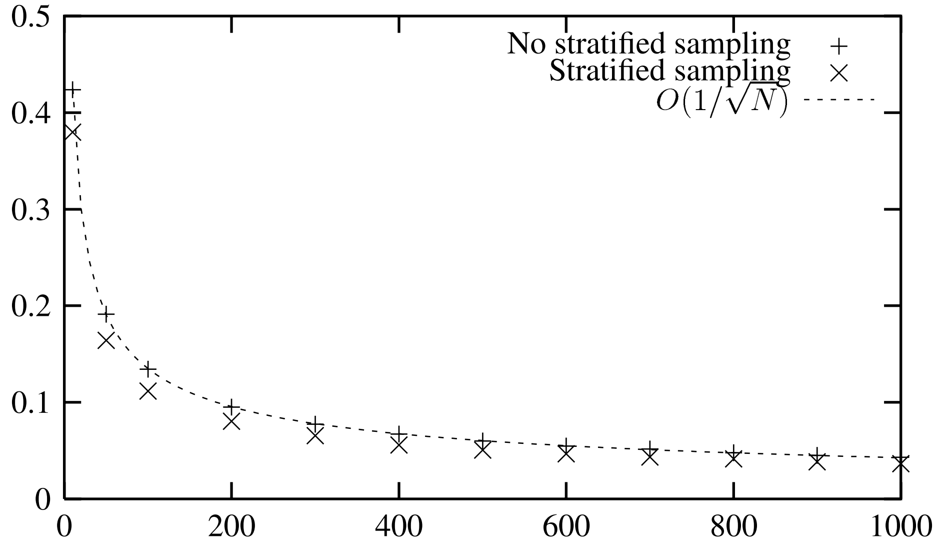


Figure 5.4: The RMS errors for renderings of test scene A, without and with stratified sampling, as functions of the number of samples per pixel. The ordinary secondary estimator has a typical $O(1/\sqrt{N})$ convergence rate. In this case stratified sampling yields a more or less consistent reduction of the RMS error by 10–15%.

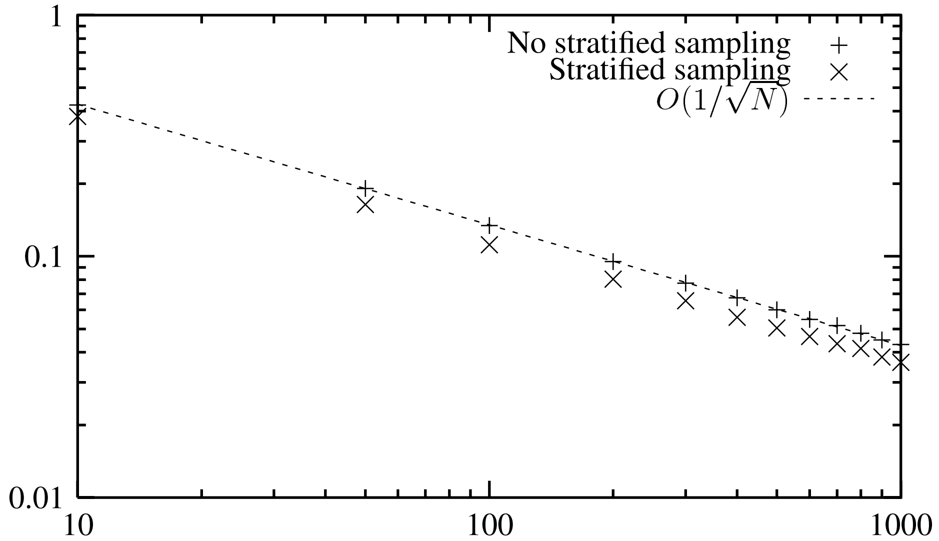


Figure 5.5: The same graphs on a logarithmic scale, clearly showing the $O(1/\sqrt{N})$ convergence rate.

5.4 Importance Sampling

As discussed in Sections 4.1.3 and 4.3.3, importance sampling mainly improves the sampling of reflection directions for path tracing and bidirectional path tracing. We have tested the effects of three common techniques. No importance sampling corresponds to sampling the hemisphere above a surface uniformly. The normalised PDF for the direction Θ_{x_i} then is

$$p_i(\Theta_x) = \frac{1}{2\pi}.$$

This technique can easily be improved by also taking the cosine factor into account:

$$p_i(\Theta_x) = \frac{\left|\Theta_x \cdot N_{x_{i-1}}\right|}{\pi}.$$

The optimal PDF on the basis of local information also includes the BRDF:

$$p_i(\Theta_x) = \frac{f_r(x_{i-1}, \Theta_x, \Theta_{x_{i-1}}) \left|\Theta_x \cdot N_{x_{i-1}}\right|}{\rho(x_{i-1}, \Theta_{x_{i-1}})}.$$

Sampling according to this ideal PDF is computationally expensive for the modified Phong BRDF. We have therefore used a comparable sampling approach, which is explained in [66]. Moreover, each PDF is only an averaged version of the PDFs for each colour component, as we only select a single reflection direction at each intersection point. For coloured surfaces the averaged PDF is thus rarely ideal for any of the colour components.

Obviously, the improvements from importance sampling strongly depend on the reflectance characteristics of the scene. Still, to get an impression, Table 5.2 lists the improvements for the RMS errors for the test scenes. The results show that taking into account the cosine in the sampling function reduces the RMS errors by 5 to 30%. Additionally taking into account the BRDF reduces the RMS errors by 40% or more for the Cornell scene, even if it is mostly diffuse, and by about 30% for the office model, which is even more diffuse. Note that this optimisation is standard for almost all ray tracing algorithms. They are therefore better suited for rendering specular scenes than diffuse scenes. The example scenes are not particularly easy in this respect.

Test scene	Reduction of the RMS errors	
	Cosine	BRDF × cosine
A	30 %	80 %
B	10 %	40 %
C	10 %	40 %
D	5 %	30 %
E	5 %	30 %

Table 5.2: Overview of the average reduction of the RMS errors due to importance sampling, for the same numbers of samples. The different test scenes have been rendered with their respective rendering algorithms. The RMS errors resulting from two importance sampling strategies are compared to the RMS errors resulting from uniform sampling.

5.5 Next Event Estimation

Next event estimation for path tracing has been discussed in Section 4.1.4. The improvements due to next event estimation in path tracing can be arbitrarily large, depending on the size of the light sources. For scenes with very large light sources pure path tracing may perform better, but for most common scenes path tracing with next event estimation should yield a lower variance. The smaller the light sources, the larger the improvement will be. For point light sources next event estimation remains the only viable alternative.

Next event estimation introduces additional shadow rays in the path tracing algorithm. Table 5.3 shows the number of rays per sample for pure path tracing and for path tracing with next event estimation. The first column shows that the average path lengths are 2.4 and 2.6 for test scenes A and D respectively. The total number of rays for any image can then easily be estimated using the expression

$$\begin{aligned} \textit{Total number of rays} &= \textit{average number of rays per sample} \\ &\quad \times \textit{number of samples per pixel} \\ &\quad \times \textit{number of pixels}. \end{aligned}$$

For test scene A, rendered at 100×100 pixels with a 100 samples per pixel, this yields approximately $2.4 \times 100 \times 100 \times 100 = 2.4 \times 10^6$ rays, for example.

The second column of Table 5.3 shows the increase in the numbers of rays for the test scenes. Next event estimation results in twice as many rays, at most. Many rays can be discarded immediately, for instance if the surfaces are facing away from one another. As the tracing of a ray is computationally relatively expensive, it can be considered as a basic unit of work. We have therefore compared the performances of the different algorithms on the basis of the same total number of rays.

Note that in our implementation, path tracing with next event estimation has a smaller computational overhead per ray than pure path tracing. This is most probably true in general, as evaluating BRDFs is less involved than sampling according to them. Path tracing with next event estimation performs relatively more evaluations, while pure path tracing performs relatively more sampling operations. The difference only matters for simple scenes, where the overhead may be important compared against the cost of tracing a ray.

Table 5.4 gives an impression of the reduction of the RMS errors for scenes A and D, compared to pure path tracing with the same number of rays.

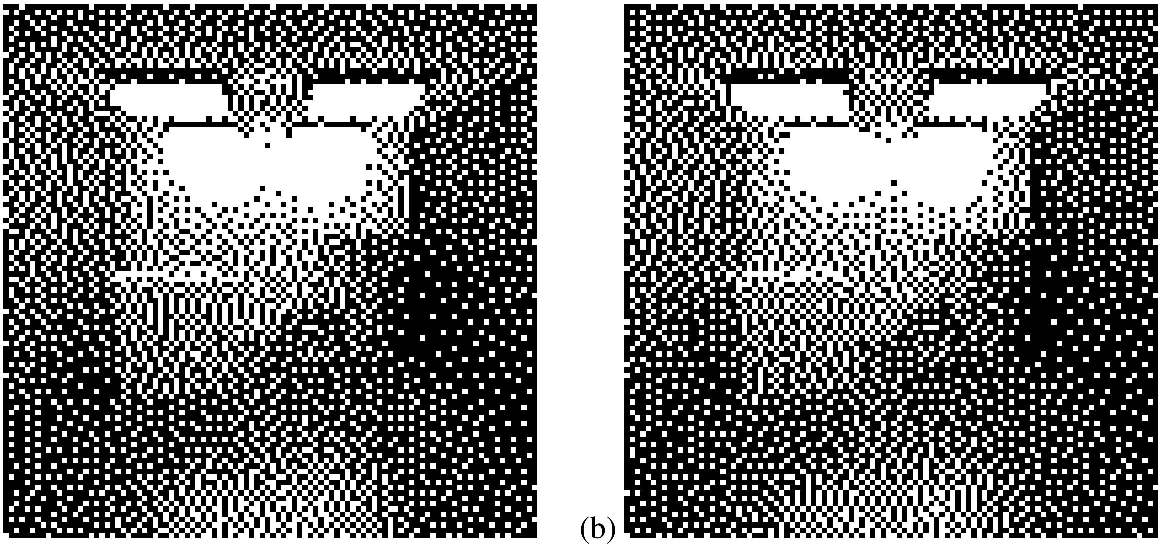
Test scene A has artificially large light boxes, which is relatively unfavourable for next event estimation. Figure 5.6 shows two images rendered with the same amount of work, without and with next event estimation respectively. The image rendered with next event estimation looks better, because it is smoother. There are bright spikes scattered throughout the image however. These result from the division in the estimator, by a squared distance that can become arbitrarily small. Even when ignoring the few worst spikes the RMS errors are up to 100% larger. In test scene D the light boxes are smaller, but the spikes also ruin the improvements of next event estimation.

Test scene	Average number of rays per sample	
	Pure path tracing	Path tracing with next event estimation
A	2.4	3.8
B	–	4.1
C	–	3.4
D	2.6	3.5
E	–	3.6

Table 5.3: Overview of the average numbers of rays per primary sample, for path tracing without and with next event estimation.

Test scene	Reduction of the RMS errors
A	$-100\ \%$
D	$-20\ \%$

Table 5.4: Overview of the reduction of the RMS errors due to next event estimation, for test scenes A and D. For these particular scenes the RMS errors are increased. The increase results from spikes that are introduced, which overwhelm the overall visual improvement of the images.



(a) (b)

Figure 5.6: (a) Test scene A rendered with pure path tracing, at 100×100 pixels with 800 samples per pixel. (b) Test scene A rendered with path tracing with next event estimation, with 500 samples per pixel, such that the total number of rays is approximately the same. The image looks smoother, but contains spurious bright spikes that increase the RMS error.

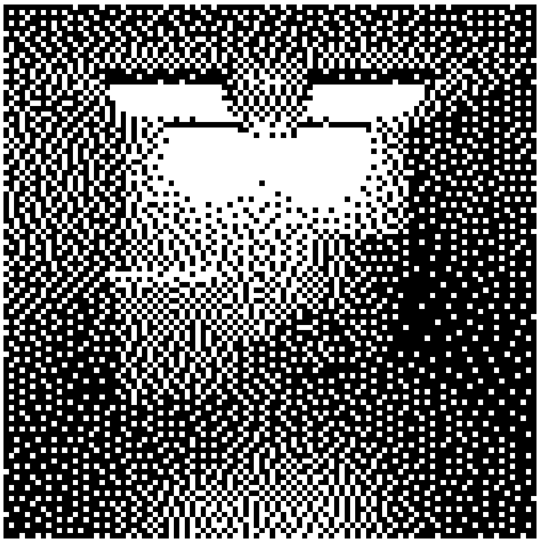


Figure 5.7: Test scene A rendered with path tracing with next event estimation and combined estimators, again with 500 samples per pixel.

5.6 Combining Estimators

As explained in Section 4.1.5 it may be interesting to combine different estimators automatically. In this case pure path tracing and path tracing with next event estimation yield different estimators. These can be combined. For specific scenes, the resulting estimator is not necessarily better than either of the individual estimators. It is more robust however, in that it handles a wider range of scenes and illumination situations more gracefully. It also does not require any additional rays over next event estimation.

Table 5.5 gives an idea of the influence of the combined estimator of the RMS error when rendering test scenes A and D. For the other scenes, which only have point light sources, the combined estimator is equivalent to next event estimation. While next event estimation gave an in increased variance for test scenes A and D, the combination of the two estimators reduces the RMS errors substantially. Figure 5.7 shows test scene A rendered with 500 samples per pixel, which can be compared to Fig. 5.6. The results confirm the original visual comparison presented by Veach and Guibas in [126].

Test scene	Reduction of the RMS errors	
	Next event estimation	Combined estimators
A	-100 %	35 %
D	-20 %	40 %

Table 5.5: Overview of the reduction of the RMS errors due to next event estimation and to combining estimators, both compared with pure path tracing, for test scenes A and D. Unlike next event estimation on its own, the combined estimators do improve the results.

5.7 Control Variates

In Section 4.1.6 we have proposed a further variance reduction for path tracing by means of simple control variates. The control variates are based on a constant estimate for the ambient radiance L_a in the scene. The control variate is then proportional to the BRDF times a cosine factor. For the modified Phong BRDF this factor can be integrated analytically using an algorithm proposed by Arvo [2].

The reduction of the variance and the RMS errors depends on the quality of the estimate. Figure 5.8 shows how the reduction varies as a function of L_a for test scene B, for a fixed number of samples per pixel. The control variate equals the ambient radiance times the BRDF and the cosine factor. A value of 0 for the ambient radiance corresponds to not using control variates at all. Values greater than 0 decrease the RMS errors up to a certain point, after which the RMS errors increase again. The maximum reduction is 10% in this case.

Table 5.6 presents the improvements obtained for test scenes A, B and C. Test scene A has been rendered with pure path tracing. The ambient radiance then has to approximate the total incoming radiance. The overall reduction of the RMS error is small, because the constant ambient radiance is a too simple an approximation of the highly varying incoming radiance.

Test scenes B and C have been rendered with path tracing with next event estimation. The ambient radiance then only has to approximate the more constant incoming radiance due to reflections. The reduction is larger for test scene C because this indirect illumination is relatively more important in this scene.

Images rendered with control variates tend to look better visually, especially for few samples per pixel. Each pixel receives an acceptable estimate that differs from 0. Without control variates pixels may remain black, if none of the random walks happen to receive an illumination contribution. Figure 5.9 illustrates this effect for test scene C, rendered with only 10 samples per pixel.

We have not repeated the experiments with the improved importance sampling and control variates that were presented in [67]. These show that the RMS errors can be reduced by 30% for test scene A, rendered with pure path tracing , and by 20 to 25% for test scene C, rendered with path tracing with next event estimation. The memory overhead of the 5D tree varies between 2 MB and 14 MB, but the improvements depend little on the size of the tree.

Test scene	Reduction of the RMS errors
A	4 %
B	10 %
C	14 %

Table 5.6: Overview of the reduction of the RMS errors by means of control variates. The control variates are based on a constant approximation of the incoming radiance. Depending on the quality of this approximation the RMS errors are reduced by 4 to 14% for these examples.

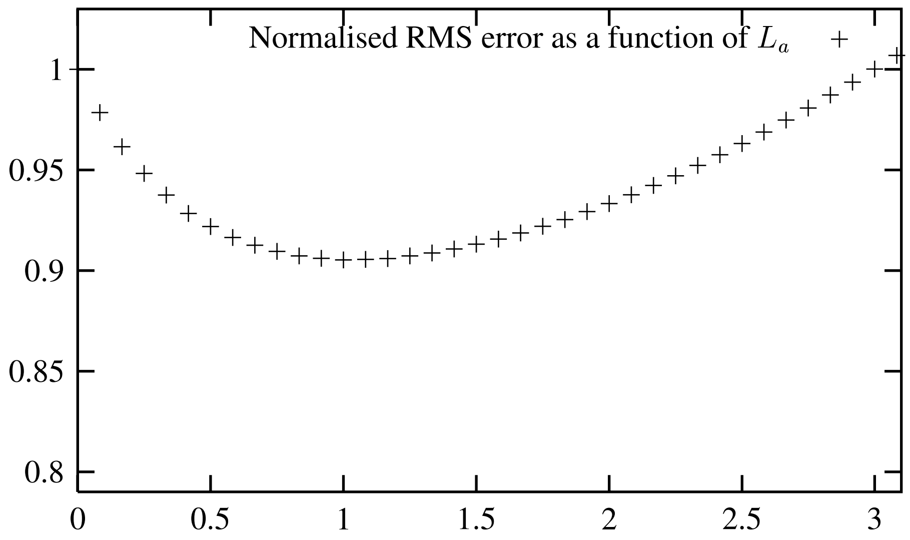


Figure 5.8: RMS errors for test scene B, as a function of the estimated ambient radiance L_a, for a fixed number of samples per pixel. The RMS errors decrease until the ambient radiance reaches its optimum estimate and then increase again.

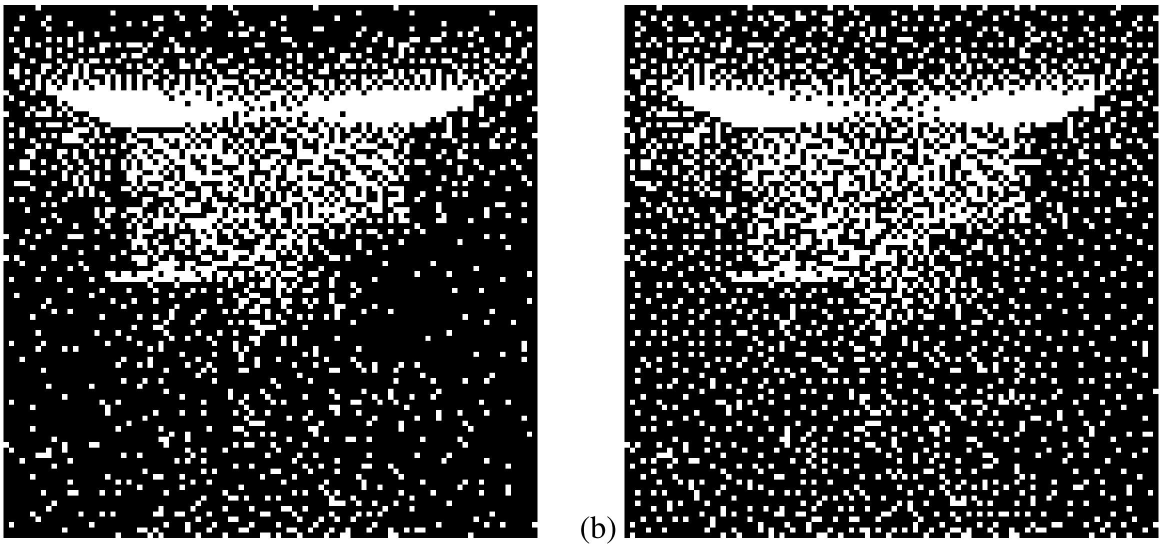
(a)  (b)

Figure 5.9: (a) Test scene C rendered with path tracing with next event estimation, at 100×100 pixels with only 10 samples per pixel. (b) Test scene A rendered in the same way, but with control variates. Each pixel now at least gets an estimated value different from black.

5.8 Bidirectional Path Tracing

We have introduced bidirectional path tracing in Section 4.3. It presents a different approach to physically based rendering than classical path tracing. For each sample it traces not only a path starting from the eye point and the corresponding shadow rays to a light source, but also a path starting from a light source and the additional shadow rays. Table 5.7 shows the increase in numbers of rays per sample for the different scenes. The number of rays is 2 to 3 times larger on average. Note that in our implementation, the computational overhead per ray of bidirectional path tracing is again lower than that of the other algorithms.

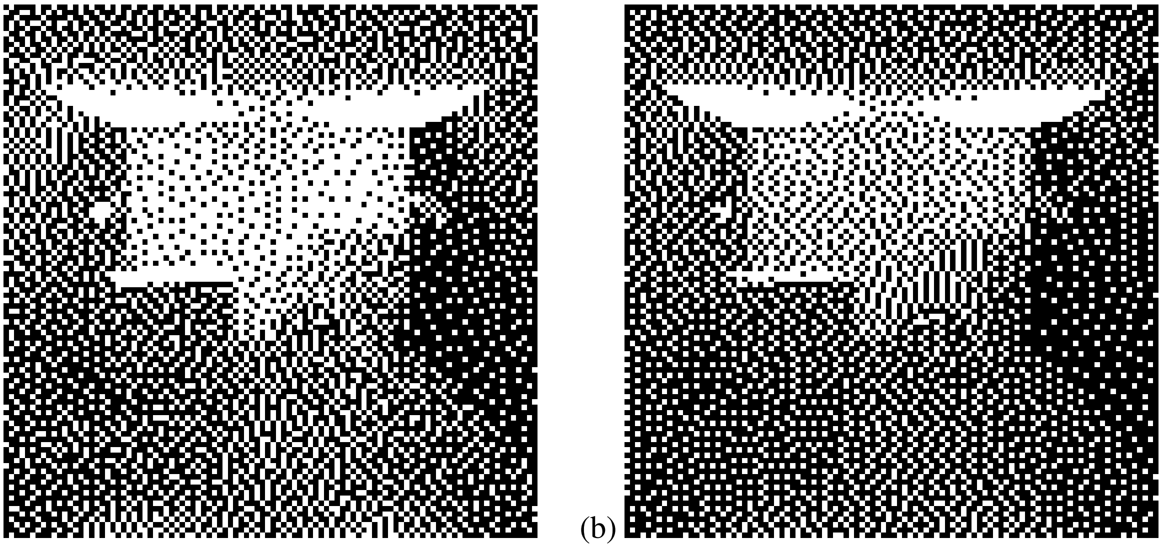
Table 5.8 gives an overview of the results of bidirectional path tracing, compared against path tracing with next event estimation and combined estimators, with the same total number of rays. Bidirectional path tracing only performs worse for test scenes A and D, which are almost completely illuminated directly. The algorithm in these cases still yields lower RMS errors for the same numbers of primary samples. When the results are compared on the basis of the same total number of rays however, the additional rays are only a computational burden that does not pay off. For the other scenes bidirectional path tracing consistently performs better. Even for test scene B, which is mostly illuminated directly, the RMS errors are reduced. Unsurprisingly, the algorithm yields the largest reduction for test scenes C and E, because they are illuminated almost exclusively indirectly. Figures 5.10 and 5.11 illustrate this with images of scenes C and E, rendered with both algorithms. The differences are obvious. Note that obtaining the same RMS errors with the path tracing algorithm as with the bidirectional path tracing algorithm would require at least four times as many samples for these scenes.

Test scene	Average number of rays per sample	
	Path tracing with next event estimation	Bidirectional path tracing
A	3.8	9.7
B	4.1	10.6
C	3.4	8.9
D	3.5	7.9
E	3.6	6.8

Table 5.7: Overview of the average numbers of rays per primary sample, for path tracing with next event estimation and for bidirectional path tracing.

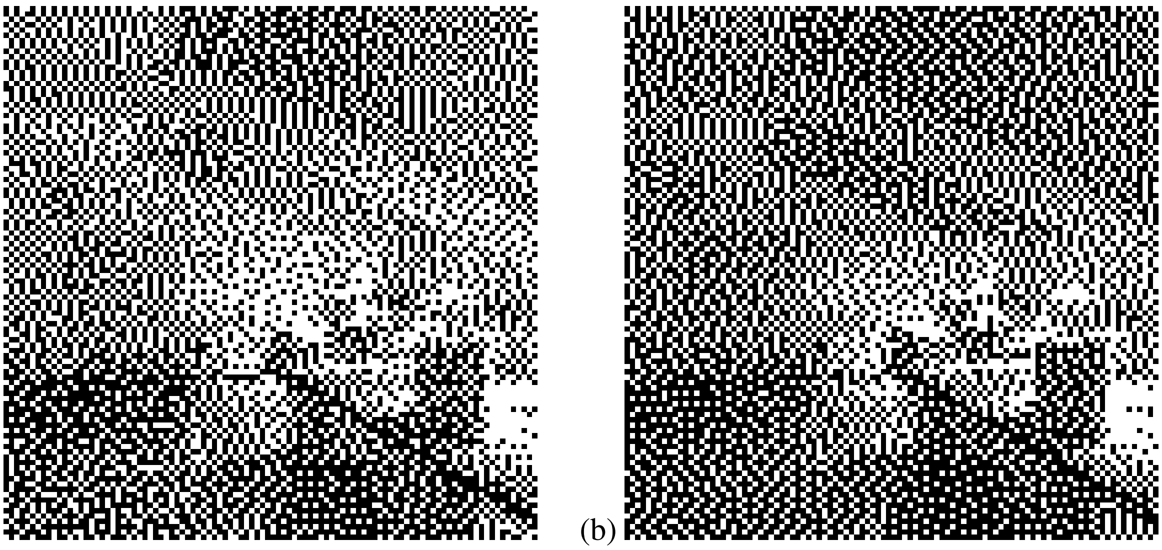
Test scene	Reduction of the RMS errors
A	-30 %
B	40 %
C	70 %
D	-20 %
E	50 %

Table 5.8: Overview of the reduction of the RMS errors by using bidirectional path tracing, for the same total numbers of rays. The reduction is compared against path tracing with next event estimation and combined estimators. It is substantial for scenes containing at least some indirect illumination.



(a) (b)

Figure 5.10: (a) Test scene C rendered with path tracing with next event estimation and combined estimators, with 1000 samples per pixel. (b) Test scene C rendered with bidirectional path tracing, with 400 samples per pixel, such that the total number of rays is approximately the same.



(a) (b)

Figure 5.11: (a) Test scene E rendered with path tracing with next event estimation and combined estimators, with 1000 samples per pixel. (b) Test scene E rendered with bidirectional path tracing, with 500 samples per pixel, such that the total number of rays is approximately the same.

5.9 Summary

In this chapter we have tested the algorithms and optimisations presented in the previous chapters against a set of example scenes. While the examples and the results may not be representative for every conceivable input scene, they do provide an impression of the value of the various techniques.

We have measured the RMS errors of different series of test images, compared against reference images. The latter have been rendered at a high accuracy. The RMS errors give an indication of the variances and standard deviations of the random processes, averaged over the images. Similarly to the standard deviation the RMS errors decrease as $1/\sqrt{N}$, where N is the number of primary samples per pixel.

In our experiments, stratified sampling using multi-dimensional N-rooks sampling provides a more or less consistent reduction of the RMS error by 20%. The reduction hardly varies with the number of samples per pixel in our tests, which go up to a 1000 samples per pixel.

Next event estimation provides a visible reduction of the noise in the images rendered with path tracing, but it can also introduce spurious bright spikes. These are due to divisions by distances that can become arbitrarily small, as explained in the previous chapters. If the spikes are not filtered out, the RMS errors increase significantly. In spite of this, next event estimation remains the only alternative for very small light sources or point light sources. We have not tested more complex sampling strategies of the light sources, which have been mentioned in Chapter 4. However, without any additional cost the estimators of pure path tracing and path tracing with next event estimation can be combined into a more robust estimator. The combined estimator yields a significantly lower RMS errors in our experiments.

We have shown that control variates can also yield a reduction of the RMS error. For constant approximations of the incoming radiance we have observed reductions of about 10%. More accurate approximations on the basis of information about the incoming radiance, stored in a 5D tree, gave reductions of about 25%.

Bidirectional path tracing presents an alternative approach to physically-based rendering. On the basis of the same number of samples per pixel it consistently performs better than the classical path tracing with next event estimation, even with combined estimators. The algorithm does trace more rays to obtain these results; 2 to 3 times as many in our test scenes. Compared on the basis of the same total number of rays, bidirectional path tracing may perform worse if the entire scene is only illuminated directly. It performs markedly better if there is at least some indirect illumination in the scene. For typical interior scenes with indirectly illuminating lamp shades the RMS errors are easily reduced by 50% or more.

Chapter 6

Summary and Conclusions

6.1 Summary

In this dissertation we have investigated the application of Monte Carlo techniques to physically based rendering. We have attempted to show that Monte Carlo rendering can get a long way with just standard numerical techniques and optimisations.

In Chapter 2 we have laid the basis for a systematic approach of the global illumination problem, which is the core problem of physically based rendering. The starting point is a particle model of light. Light is emitted from light sources, reflected at surfaces and eventually absorbed. The fundamental physical quantities that describe these phenomena are radiance and radiant flux. Emittance of the light sources is described by the self-emittant radiance function, while the reflectance characteristics of the surfaces are described by the bidirectional reflectance distribution function (BRDF).

Different mathematical frameworks formally describe the global interreflection in the scene. They are equivalent, but each model presents a different point of view. The rendering equation is the most well-known model, describing the behaviour of radiance. It depends on the self-emitted radiance and on the geometry and reflectance characteristics of the surfaces in the scene. Taking into account the self-emitted potential, the radiant flux can be expressed in terms of radiance.

The potential equation is adjoint to the rendering equation, describing the behaviour of the potential function. The potential function is defined with respect to the self-emitted potential and to the surfaces in the scene. Taking into account self-emitted radiance, radiant flux can now be expressed in terms of potential.

We have then introduced the global reflectance distribution function (GRDF). This function only depends on the surfaces in the scene and describes its global reflectance characteristics. As such, it can be regarded as a generalisation of the bidirectional reflectance distribution function, which describes the local reflectance characteristics. We have shown that the GRDF is defined by two equivalent integral equations that correspond closely to the rendering equation and the potential equation. Taking into account self-emitted radiance and self-emitted potential, radiant flux can finally be expressed in terms of the GRDF.

We have chosen to apply Monte Carlo methods to these models to solve the global illumination problem, in the context of image-based rendering. This approach seems to fit our objectives well, as we aim for versatile algorithms that correctly simulate a broad range of illumination effects. The algorithms should not restrict the geometries and optical properties that can be rendered. The illumination functions that have to be integrated are high-dimensional and generally not well-behaved. Given these observations and considering that physically based rendering only requires a limited accuracy, Monte Carlo methods are a viable alternative.

In Chapter 3 we have given an overview of Monte Carlo techniques that can be relevant to the global illumination problem. While the principle is simple and elegant, convergence is slow. The emphasis therefore lies on variance reduction techniques that improve the results for the same amount of work: stratified sampling, importance sampling, control variates, next event estimation and other variance reduction techniques. The underlying idea of all variance reduction techniques is to transform the original integrand to a function that is more constant and thus easier to integrate. We have stressed the unbiasedness of the techniques, as a minimal reassurance as to the correctness of the results. We have presented potential improvements to recently developed strategies for combining estimators. We have also specifically investigated the effects of combining importance sampling and Russian roulette with control variates. The results indicate that control variates can reduce the variance further in this case, under certain conditions which we have derived. They can therefore also be useful when solving the global illumination problem.

In Chapter 4 we have applied the above Monte Carlo techniques to the mathematical models of the global illumination problem. Each model naturally leads to a different algorithm. The rendering equation gives rise to the well-known path tracing algorithm. Path tracing traces random walks starting from the eye point, through a pixel, adding a contribution to the estimate of the flux of the pixel each time a light source is hit. As such it is a light gathering approach. The variance of the estimates shows up as random noise in the images. Standard Monte Carlo optimisations such as importance sampling and next event estimation considerably reduce the variance. We have also studied how control variates can be applied to good effect.

The potential equation on the other hand leads to light tracing algorithms. These trace random walks starting from the light sources, adding contributions to the pixels as they pass through them. This corresponds to a light shooting approach. In the case of image-based rendering it is mostly of theoretical interest. It is only practical to simulate specific parts of the global illumination solution, as the resulting images are too noisy in practice, even with variance reducing techniques.

Finally, application of Monte Carlo methods to the GRDF and its integral equations leads to a new algorithm, which we have called bidirectional path tracing. It traces light paths from the light sources and eye paths from the eye point. Shadow rays that connect the points on the respective random walks then determine the contributions to the estimate of the flux through the pixel. This combined shooting and gathering approach seamlessly integrates the interesting properties of the previous algorithms. Both a theoretical analysis and practical test results indicate its suitability for rendering scenes with complex indirect illumination. Similar variance reduction techniques as in path tracing and in light tracing are applicable.

In Chapter 5 we have measured the improvements from the different variance reduction techniques and the different algorithms, for different types of scenes. The main improvements for scenes with a lot of indirect illumination result from the bidirectional path tracing algorithm.

6.2 Conclusions

In the introduction in Chapter 1 we have set our objectives for this work. We have researched algorithms that are correct, flexible and efficient, in this order of priority. We believe that we have met the first two conditions and made progress with regard to efficiency:

- **Correctness.** The concise physically based and mathematical approach of this work guarantees the correctness of the rendering algorithms that have been presented. At the same time it allows to assess their limitations.

  The starting point was a physical model of the behaviour of light. The particle model describes most common optical effects, such as emission and reflection. It does not account for more exotic phenomena such as interference and polarisation though, for which a wave model would be required. This alternative point of view would lead to more complex algorithms, while the effects are of marginal importance in practice. Furthermore, we have only included the interaction of light with surfaces and neglected the influence of participating media, such as smoke and fog. Again, this a reasonable approximation for common scenes. The resulting physical model has been formalised in a mathematical notation.

  We have then applied standard Monte Carlo techniques to the mathematical models and analysed the performances of the resulting algorithms. As a minimal reassurance as to the mathematical correctness, we have adhered to the principle of unbiasedness for the Monte Carlo techniques. While there is still an uncertainty on the results, the unbiasedness guarantees that they converge to the exact solutions. The results can be computed to any accuracy desired, given sufficient computation time. Although partly of theoretical interest, this property can also be useful in practice when computation time is less of a concern than accuracy. A typical example in the context of research on image synthesis is the rendering of reference images for other rendering techniques.

- **Versatility.** We have taken care not to limit the geometric and optical characteristics of the input scenes, nor the global illumination effects rendered in the output images. The presented ray tracing-like algorithms are point-sampling techniques. They only probe the surfaces by computing intersections with rays. As this can be done efficiently for most types of surfaces or objects, the algorithms can handle arbitrary geometries. This approach is somewhat more flexible than only allowing a boundary element representation using polygons.

  The optical characteristics can be described by any BRDF model. Although some models may be easier or more efficient to work with, the algorithms can handle any type of BRDF. They can range from perfectly specular BRDFs, over glossy BRDFs, to perfectly diffuse BRDFs. Again this is more flexible than only assuming perfectly diffuse BRDFs or artificially separating BRDFs into diffuse and specular parts.

  Within the given framework the algorithms can render all typical global illumination effects, such as glossy reflection and colour bleeding. Especially the new bidirectional path tracing algorithm excels in rendering a wide range of simple and complex illumination situations. Additionally, it can simulate camera effects such as depth of field and motion blur without much additional effort. Figures 6.1 and 6.2 illustrate these effects.

- **Efficiency.** Given the previous prerequisites we have concentrated on the efficiency of the algorithms. The basic Monte Carlo algorithms are versatile, but generally very slow to converge. First of all we have investigated the application of standard variance reduction techniques to the rendering equation and the corresponding path tracing algorithm. Experiments give an indication of the improvements that can be expected in practice from known techniques such as stratified sampling, importance sampling and next event estimation. Additionally we have shown how control variates can reduce the variance further.

  The main improvements result from starting from a different mathematical model altogether. More specifically the model containing the GRDF and the resulting bidirectional path tracing algorithm augment the efficiency considerably for complex global illumination situations.

  The low memory requirements of the Monte Carlo algorithms are a major advantage. The memory needed to store the input scene aside, the requirements are actually negligible. This allows for very complex and large scenes. With the proper optimisation techniques the complexity of ray intersection tests is reported to be $O(\log N)$, where N is the number of primitives in the scene [24, 75]. The combination of low memory requirements and a ray tracing approach thus yields an excellent scalability to render complex scenes. The view-dependency and light source-dependency additionally ensures that most computational effort is spent on parts of the scene that matter for the final image.

Obviously, there are numerous techniques that are biased from a strict Monte Carlo point of view, but that can improve the appearance of the images considerably, without much additional work. Filtering techniques are an obvious example. Simple Gaussian filters or median filters can be applied to the final image, to reduce the random noise and the disturbing bright spikes, e.g. [51]. Rushmeier and Ward [97] suggest alternative and physically more sound filters. All filters do however introduce a bias in each individual pixel, because the results of neighbouring pixels now affect one another.

Additionally filtering and caching in object space can further reduce the required number of samples. Rushmeier *et al.* [95] and Kok [59, 60] propose grouping the objects in simplified primitives, for instance. In his illumination simulation system Ward [134, 133, 132] caches and interpolates the diffuse component of the illumination in object space, to great success.

Adaptive sampling is another common technique. It concentrates the computational efforts on regions of the image or of the scene that are considered particularly complex. Various strategies have been presented, e.g. in [90, 99, 129, 110]. Adaptive sampling can be effective at reducing the number of rays that have to be traced. Most of these techniques are biased, as shown by Kirk and Arvo [56, 57]. Unbiased adaptive sampling is feasible but expensive, which explains the greater popularity of the biased strategies.

In summary, we have researched versatile and unbiased Monte Carlo algorithms for physically based rendering. The versatility is obtained thanks to the Monte Carlo approach, at the cost of higher computation times, for instance compared to optimised radiosity methods. We have then concentrated on variance reduction techniques that improve the results in an unbiased way. There is a substantial cost associated with maintaining the unbiasedness. Whenever an approximation is available, however accurate, unbiasedness always demands that the result be computed explicitly. The approximation cannot replace the result, it can only help to reduce its variance. This provided an interesting challenge. We have shown that substantial progress can be made, even within these constraints.

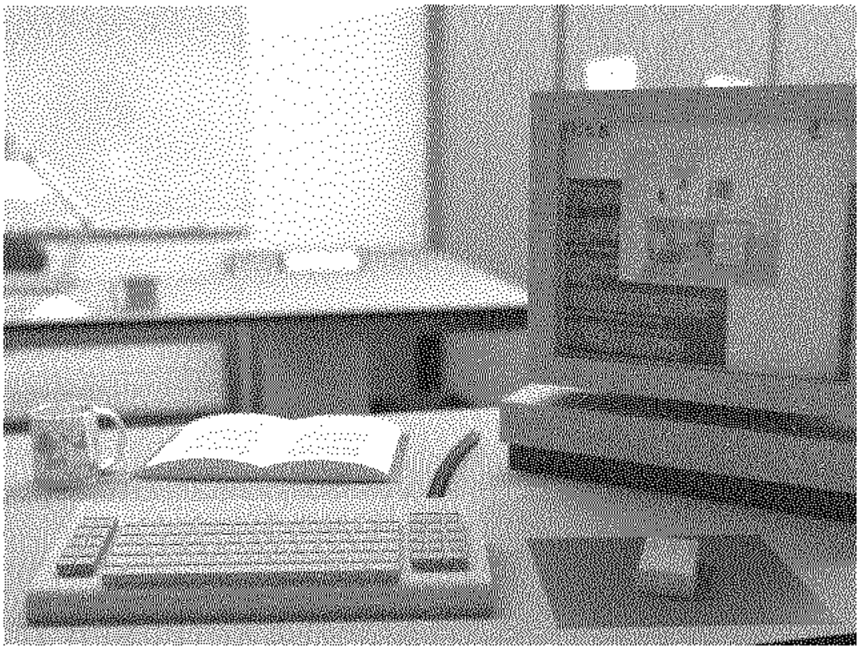


Figure 6.1: An example of the versatility of the algorithms developed. The office scene is rendered with a finite camera aperture. This produces the depth of field: the foreground is in focus, while the background is blurred, as with a real camera.

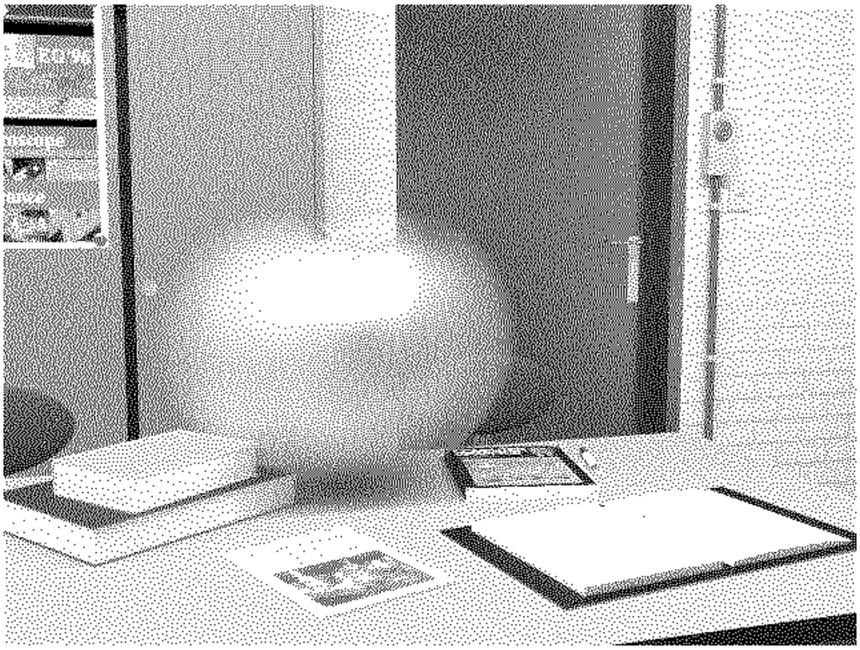


Figure 6.2: The office scene rendered with a finite exposure time, resulting in a motion blurred image of the beach ball that bounces on the table.

6.3 Future Work

Many directions remain to be explored in the field of Monte Carlo rendering. In the context of this dissertation the following subjects seem worthwhile investigating:

- Bidirectional path tracing could also be applied to finite element representations of the scene. As mentioned in Chapter 2 the expression for the radiant flux is valid for any set of points and directions. In the algorithms discussed in Chapter 4 the sets consist of points and directions pointing through the pixels. This starting point yields image-based algorithms. Alternatively the sets could consist of the points and directions corresponding to finite elements. Section 4.2 on light tracing already mentioned that the technique is better suited to compute the fluxes of finite elements, as in Pattanaik's particle tracing algorithm [83]. It is a light shooting method and as such it efficiently takes into account the light sources. On the other hand it may miss receiving elements, especially if they are small. Bidirectional path tracing would be a way to ensure that each element gets sufficient reliable contributions for its estimate, by starting random walks from each element. Alternatively it could be a way to include view-dependence in the computations, by starting random walks from the eye point.

- Participating media have been disregarded in this work. Nothing precludes considering them in the mathematical models however. The bidirectional path tracing algorithm can then be extended fairly easily on the basis of the additional terms in the equations. The approach can be similar to the extension of the path tracing algorithm by Rushmeier [93, 94] and the light tracing algorithm for foggy atmospheres by Rozé *et al.* [92]. By not only scattering rays and casting shadow rays at surfaces, but also inside participating media the resulting global illumination effects could be taken into account efficiently. Again bidirectional path tracing would be most effective for complex global illumination situations, for instance when a scene is illuminated indirectly by light scattered from inside a participating medium.

- The bidirectional path tracing algorithm itself can no doubt be improved further. We have not considered control variates or improved importance sampling as we have for the path tracing algorithm. Although the basic principles would remain the same, the actual application would become more difficult and more elaborate. Improved importance sampling could also be a step in the right direction to handle large numbers of light sources. By adaptively determining the importance of the light sources for the image less light paths would be traced from possibly distant or irrelevant light sources and more from nearby and important light sources. Additionally the directions of the light paths could be sampled more intelligently, as discussed for light tracing in [28, 29].

- Finally, many other developments in Monte Carlo research remain to be investigated in the context of physically based rendering. Pseudo-Monte Carlo methods with deterministic error bounds for instance, as proposed by Zaremba [136], look most attractive. Spanier [120] has presented biased estimators that have a bias that rapidly decreases to 0, but a much smaller variance than their unbiased counterparts. Further research should point out if these techniques are useful for the rendering algorithms discussed in this dissertation.

Appendix A

Camera Models

In our discussions we assume that the necessary data to compute synthetic imagery of a scene are known: the geometry by means of the ray casting function $r(x, \Theta_x)$, the reflective properties by means of BRDFs $f_r(x, \Theta_x, \Theta_y)$, the light sources by means of the self-emitted radiance $L_e(x, \Theta_x)$ and the viewing parameters by means of the self-emitted potential $W_e(x, \Theta_x)$. The geometry, the reflective properties and the self-emitted radiance can in principle be measured and then modelled accordingly. The definition of the self-emitted potential is straightforward for radiosity-like methods: the function is 1 for all points and directions of the patch under consideration and 0 elsewhere. For image-based rendering techniques it depends on the imaging process and is only known indirectly. There are a few alternative approaches to start with, depending on whether one strives for absolute realism or for photo-realism.

A first alternative goal of a synthetic image may be to recreate the same visual experience for the viewer as if he were watching the actual scene [78]. Figure A.1 explains this approach. The visual perception of the viewer of a scene is determined by the field of radiance values reaching the eye point. When replacing the scene by a display a synthetic image may attempt to render the same field of radiance values and thereby recreate the same perception. For digital imagery the display is discretised into pixels that render averaged radiance values. If for a given pixel a

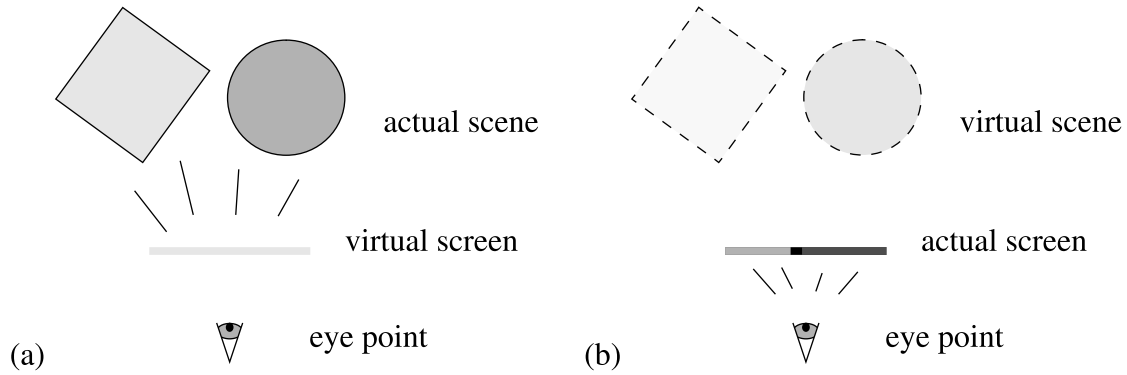


Figure A.1: Synthetic images may attempt to recreate the visual perception of an actual scene. (a) The radiance values reaching the eye of the viewer determine the perception of the actual scene. (b) The viewer has the same visual impression if the image emits the same radiance values.

normalised filter function $f(z)$ is applied, the averaged radiance value can be expressed by

$$\Phi = \int_{A_{screen}} L(z, \Theta_{z \to y}) f(z) d\mu_z,$$

where:

- A_{screen} = the surface area of the virtual screen,
- y = the eye point.

The corresponding self-emitted potential for all surface points and directions consists of a sum of Dirac functions, because each visible point has exactly one direction for which the function differs from 0.

This goal of absolute realism is implicitly pursued by most implementations of ray tracing and radiosity algorithms. Obviously several technological problems hamper this approach. Display devices such as monitors, slide projectors and printers have limited dynamic ranges and limited colour gamuts. The ranges of computed intensities and colours in an image need to be rescaled on the basis of a perceptual model to approximate the actual range. Among others Hall [37] presents solutions for the constraints of colour gamuts and Tumblin *et al.* [124] and Schlick [101] suggest functions that map radiance values on displayed intensities. The images may need to be corrected further for each specific display device. Consider for instance a point on a slide projection screen that has been assigned a certain intensity depending on its computed radiance value. For a given viewing position the same intensity is perceived brighter or darker depending on its position on the screen, as the reflection is in general not perfectly diffuse. A simple image-processing operation based on the characteristics of the device could overcome this effect. In practice however the characteristics are rarely known and the effect is considered small enough to be neglected.

An alternative goal may be photo-realism in the literal sense, where the computed image should resemble a picture taken by an actual camera (Fig. A.2). Effects such as a small dynamic range and a limited colour gamut then become inherent parts of the solution. Further effects

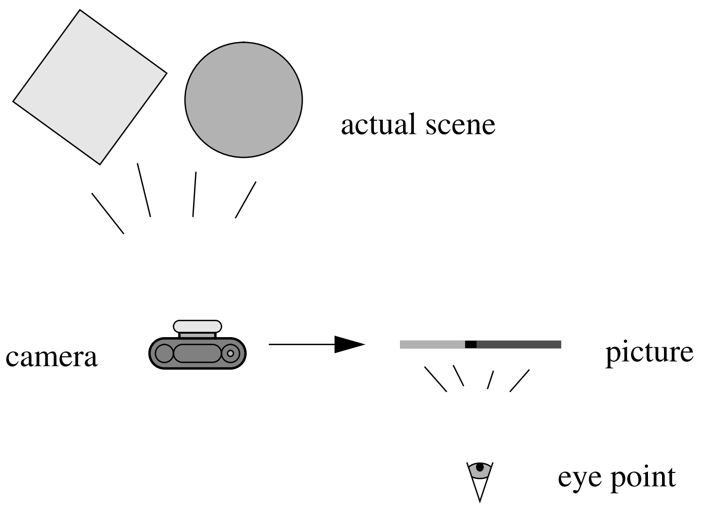


Figure A.2: Alternatively, synthetic images may attempt to recreate a photographic image of the scene, including all its artifacts and limitations.

such as depth of field and motion blur are obtained by modelling a physical camera, including all its deficiencies and constraints. Consider the simple pin-hole camera of Fig. A.3 with a finite aperture. The *irradiance* E $[W/m^2]$, flux per unit surface area, that reaches a point z on the film plane is expressed by

$$\begin{aligned} E(z) &= \int_{\Omega^{-1}_{z,aperture}} L(x,\Theta_x)\,|\Theta_x \cdot V|\,d\omega_x \\ &= \int_{A_{aperture}} L(x,\Theta_x)\,|\Theta_x \cdot V|\,\frac{|\Theta_x \cdot V|}{\|y-z\|^2} d\mu_y \\ &= \int_{A_{aperture}} L(x,\Theta_x) \frac{|\Theta_x \cdot V|^4}{f^2} d\mu_y, \end{aligned}$$

where:

- $\Omega^{-1}_{z,aperture}$ = the set of incoming directions to point z through the camera aperture,
- V = the general camera direction,
- $x = r(z, \Theta_x^{-1})$ and $\Theta_x = \Theta_{y \to z}$.

For a given shutter time the irradiance at each point on the film determines the eventual brightness of that point. In a faithful simulation the irradiance is integrated over time. Any changes in the scene or the viewing parameters during the exposure of the film will then show up as motion-blur. Also note how the cosine factor to the fourth power can be regarded as an artifact of the camera. The picture will be relatively darker further away from the centre of the image, even in an environment with a uniform radiance field in all directions. One factor results from points on the film further from the centre being more turned away from the incoming light. Another factor results from the apparent aperture becoming smaller. The final two factors result from the distance from the camera aperture becoming larger, thereby also reducing its apparent size.

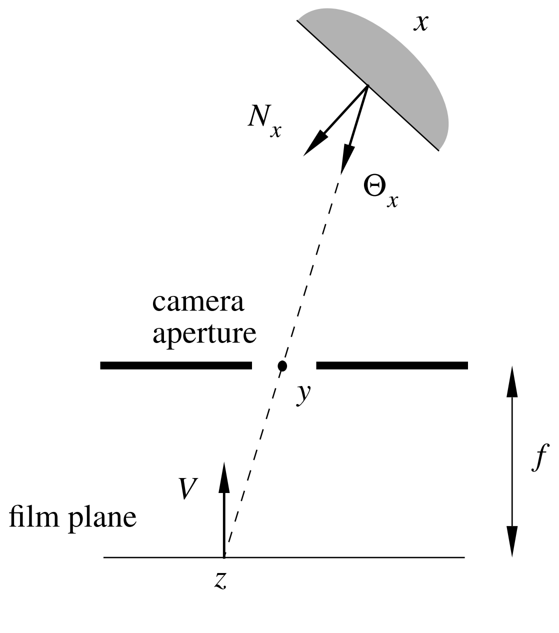


Figure A.3: A model of a pin-hole camera with a finite aperture and a film plane parallel to it.

The finite aperture of the camera accounts for depth of field in the image as different points in the scene contribute to the illumination of a single point on the film plane. Without additional lenses the plane of focus is the film plane or the virtual screen itself. Again algorithms for image synthesis can easily omit depth of field by using a perfect pin-hole point instead of a finite aperture. Kolb *et al.* [61] discuss the modelling and application of more complex lens systems.

In image synthesis, the pin-hole and the film plane behind it correspond to the eye point and the virtual screen in front of it. Filtering the irradiance over a pixel on the film plane with a function $f(z)$ yields an averaged irradiance value that determines the brightness of the pixel:

$$\Phi = \int_{A_{film}} \int_{\Omega^{-1}_{z,aperture}} L(x, \Theta_x) f(z) \, |\Theta_x \cdot V| \, d\omega_x d\mu_z.$$

In the simplest case the filter is a box filter which is constant within the pixel and 0 otherwise. The expression can be unified with the general expression of the radiant flux in terms of radiance (2.6) after a transformation. For each pixel the self-emitted potential for each point x and direction Θ_x is thus defined by the corresponding value of the filtering function on the film plane:

$$W_e(x, \Theta_x) = \begin{cases} f(z) & \text{if } x = r(z, \Theta_x^{-1}), \\ 0 & \text{otherwise.} \end{cases}.$$

List of Symbols

The following tables give an overview of the most important symbols used throughout the text. Numbers between brackets refer to the definition or to the first equation the symbol occurs in.

x	A point in the three-dimensional space or on a surface.
Θ_x	An outgoing direction in the hemisphere at point x.
Θ_x^{-1}	The opposite direction of Θ_x.
A	All surfaces in the scene.
Ω_x	The hemisphere of outgoing directions at point x.
Ω_x^{-1}	The hemisphere of incoming directions at point x.
$dA_{x\perp}$	A differential surface area around point x, at a right angle to direction Θ_x.
$d\mu_x$	A differential surface area around point x, on the surface.
$d\omega_x$	A differential solid angle around direction Θ_x.
$\|\Theta_x \cdot N_x\|$	The absolute value of the cosine between direction Θ_x and the surface normal at point x.
$r(x, \Theta_x)$	The ray casting function, yielding the nearest point on a surface from point x along direction Θ_x (2.4).
$G(x, y)$	The geometry function (2.13).
S	The set of pairs of points and directions that contribute directly to the flux and for which the self-emitted potential is 1 (2.5).
Φ	The radiant flux (2.6, 2.16, 2.22).
$L(x, \Theta_x)$	The radiance value for point x and direction Θ_x (2.1).
$L_e(x, \Theta_x)$	The self-emitted radiance at point x and in direction Θ_x.
$L_r(x, \Theta_x)$	The radiance reflected at point x and in direction Θ_x.
$L_d(x, \Theta_x)$	The radiance resulting from direct illumination reflected at point x and in direction Θ_x.
$L_i(x, \Theta_x)$	The radiance resulting from indirect illumination reflected at point x and in direction Θ_x. Also the incoming radiance or field radiance for point x and direction Θ_x (2.2).
$L_o(x, \Theta_x)$	The outgoing radiance or surface radiance for point x and direction Θ_x (2.3).
$\overline{L}$	A weighted average of the radiance function.
L	The total self-emitted power in the scene (4.5).

$f_r(x, \Theta_i, \Theta_o)$	The bidirectional reflectance distribution function (BRDF) at point x and for incoming direction Θ_i and outgoing direction Θ_o (2.8).
$\rho(x, \Theta_i)$	The directional-hemispherical reflectance for point x and direction Θ_x (2.9).
$W(x, \Theta_x)$	The potential value for point x and direction Θ_x (2.14).
$W_e(x, \Theta_x)$	The self-emitted potential at point x and in direction Θ_x.
W	The total self-emitted potential in the scene (4.2).
$\langle f, g \rangle$	A dot product of two functions, yielding a scalar number (2.7).
T	The integral operator of the rendering equation (2.12).
T^*	The integral operator of the potential equation (2.19).
$F_r(x, \Theta_x, y, \Theta_y)$	The global reflectance distribution function (GRDF) for point x and direction Θ_x and point y and direction Θ_y (2.21).

$\langle I \rangle$	An estimator for the value of I (3.2).
σ	The standard deviation of an estimator (3.3).
σ^2	The variance of an estimator (3.3).
N	The number of samples in a secondary estimator (3.4).
$p(x)$	A probability density function (PDF) (3.9).
$P(x)$	A probability distribution function (3.11).
$w(x)$	A weight function (3.13).
P	The probability of a Russian roulette process (3.29).

List of Figures

Bibliography

[1] J. Arvo, "Backward ray tracing," Aug. 1986.

[2] J. Arvo, "Applications of irradiance tensors to the simulation of non-lambertian phenomena," *Computer Graphics*, vol. 29, pp. 335–342, Aug. 1995.

[3] J. Arvo, "The role of functional analysis in global illumination," in *Proceedings of the Sixth Eurographics Workshop on Rendering*, (Dublin, Ireland), pp. 371–382, June 1995.

[4] J. Arvo, "Stratified sampling of spherical triangles," *Computer Graphics*, vol. 29, pp. 437–438, Aug. 1995.

[5] J. Arvo and D. Kirk, "Fast ray tracing by ray classification," *Computer Graphics*, vol. 21, pp. 55–64, July 1987.

[6] J. Arvo and D. Kirk, "Particle transport and image synthesis," *Computer Graphics*, vol. 24, pp. 63–66, Aug. 1990.

[7] L. Aupperle and P. Hanrahan, "Importance and discrete three point transport," in *Proceedings of the Fourth Eurographics Workshop on Rendering*, (Paris, France), pp. 85–94, June 1993.

[8] D. Baum, H. Rushmeier, and J. Winget, "Improving radiosity solutions through the use of analytically determined form factors," *Computer Graphics*, vol. 23, pp. 325–334, July 1989.

[9] Ph. Bekaert and Y. Willems, "A progressive importance-driven rendering algorithm," in *Proceedings of the 10th Spring School on Computer Graphics and its Applications*, (Bratislava, Slovakia), June 1994.

[10] Ph. Bekaert and Y. Willems, "Importance-driven progressive refinement radiosity," in *Proceedings of the Sixth Eurographics Workshop on Rendering*, (Dublin, Ireland), pp. 349–358, June 1995.

[11] S. Chen, H. Rushmeier, G. Miller, and D. Turner, "A progressive multi-pass method for global illumination," *Computer Graphics*, vol. 25, pp. 165–174, July 1991.

[12] K. Chiu, P. Shirley, and C. Wang, "Multi-jittered sampling," in *Graphics Gems IV* (P. Heckbert, ed.), pp. 54–59, Academic Press, 1994.

[13] P. Christensen, D. Salesin, and T. DeRose, "A continuous adjoint formulation for radiance transport," in *Proceedings of the Fourth Eurographics Workshop on Rendering*, (Paris, France), pp. 95–104, June 1993.

[14] M. Cohen, "Is image synthesis a solved problem?," in *Proceedings of the Third Eurographics Workshop on Rendering*, (Bristol, UK), pp. 161–167, May 1992.

[15] M. Cohen, S. Chen, J. Wallace, and D. Greenberg, "A progressive refinement approach to fast radiosity image generation," *Computer Graphics*, vol. 22, pp. 75–84, July 1988.

[16] M. Cohen and D. Greenberg, "The hemicube: a radiosity solution for complex environments," *Computer Graphics*, vol. 19, pp. 31–40, July 1985.

[17] M. Cohen and J. Wallace, *Radiosity and Realistic Image Synthesis*. Cambridge, MA: Academic Press, 1993.

[18] S. Collins, "Adaptive splatting for specular to diffuse light transport," in *Proceedings of the Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 119–135, June 1994.

[19] R. Cook, "Stochastic sampling in computer graphics," *ACM Transactions on Graphics*, vol. 5, no. 1, pp. 51–72, 1986.

[20] R. Cook, T. Porter, and L. Carpenter, "Distributed ray tracing," *Computer Graphics*, vol. 18, pp. 137–145, July 1984.

[21] R. Cook and K. Torrance, "A reflectance model for computer graphics," *Computer Graphics*, vol. 15, pp. 187–196, July 1981.

[22] R. Cook and K. Torrance, "A reflectance model for computer graphics," *ACM Transactions on Graphics*, vol. 1, pp. 7–24, Jan. 1982.

[23] Ph. Davis and Ph. Rabinowitz, eds., *Methods of Numerical Integration*. Orlando: Academic press, 1984.

[24] M. de Berg, *Efficient Algorithms for Ray Shooting and Hidden Surface Removal*. PhD thesis, Rijksuniversiteit Utrecht, The Netherlands, 1992.

[25] M. Dippé and E. Wold, "Antialiasing through stochastic sampling," *Computer Graphics*, vol. 19, pp. 69–78, July 1985.

[26] Ph. Dutré, *Mathematical Frameworks and Monte Carlo Algorithms for Global Illumination in Computer Graphics*. PhD thesis, Katholieke Universiteit Leuven, Belgium, in preparation.

[27] Ph. Dutré, E. Lafortune, and Y. Willems, "Monte Carlo light tracing with direct computation of pixel intensities," in *Proceedings of CompuGraphics*, (Alvor, Portugal), pp. 128–137, Dec. 1993.

[28] Ph. Dutré and Y. Willems, "Importance-driven light tracing," in *Proceedings of the Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 185–194, June 1994.

[29] Ph. Dutré and Y. Willems, "Potential-driven Monte Carlo particle tracing for diffuse environments with adaptive probability density functions," in *Proceedings of the Sixth Eurographics Workshop on Rendering*, (Dublin, Ireland), pp. 339–348, June 1995.

[30] S. Ermakow, *Die Monte-Carlo-Methode und verwandte Fragen*. Berlin: VEB Deutscher Verlag der Wissenschaften, 1975.

[31] A. Fournier, "Separating reflection functions for linear radiosity," in *Proceedings of the Sixth Eurographics Workshop on Rendering*, (Dublin, Ireland), pp. 383–392, June 1995.

[32] R. Gershbein, P. Shröder, and P. Hanrahan, "Textures and radiosity: Controlling emission and reflection with texture maps," *Computer Graphics*, vol. 28, pp. 51–58, July 1994.

[33] A. Glassner, *Principles of Digital Image Synthesis*. San Francisco: Morgan Kaufmann, 1995.

[34] C. Goral, K. Torrance, D. Greenberg, and B. Battaile, "Modelling the interaction of light between diffuse surfaces," *Computer Graphics*, vol. 18, pp. 213–222, July 1984.

[35] S. Gortler, P. Schröder, and M. Cohen, "Wavelet radiosity," *Computer Graphics*, vol. 27, pp. 221–230, Aug. 1993.

[36] A. Hall, "On an experiment determination of π," *Messeng. Math.*, vol. 2, pp. 113–114, 1873.

[37] R. Hall, *Illumination and Color in Computer Generated Imagery*. New York: Springer Verlag, 1989.

[38] J. Halton, "A retrospective and prospective survey of the Monte Carlo method," *SIAM Review*, vol. 12, pp. 1–63, Jan. 1970.

[39] J. Halton, "On the relative merits of correlated and importance sampling for Monte Carlo integration," *Proceedings of the Cambridge Philosophical Society*, vol. 61, pp. 497–498, 1965.

[40] J. Hammersly and D. Handscomb, *Monte Carlo Methods*. London: Chapman and Hall, 1964.

[41] P. Hanrahan, D. Salzman, and L. Aupperle, "A rapid hierarchical radiosity algorithm," *Computer Graphics*, vol. 25, pp. 197–206, July 1991.

[42] X. He, K. Torrance, F. Sillion, and D. Greenberg, "A comprehensive physical model for light reflection," *Computer Graphics*, vol. 25, pp. 175–186, July 1991.

[43] P. Heckbert, "Adaptive radiosity textures for bidirectional ray tracing," *Computer Graphics*, vol. 24, pp. 145–154, Aug. 1990.

[44] P. Heckbert, "Discontinuity meshing for radiosity," in *Proceedings of the Third Eurographics Workshop on Rendering*, (Bristol, UK), pp. 203–216, May 1992.

[45] P. Heckbert, "Radiosity in flatland," in *Proceedings of Eurographics '92*, (Cambridge, UK), pp. 181–192, Sept. 1992.

[46] S. Heinrich and A. Keller, "Quasi-Monte Carlo methods in computer graphics, part I: The qmc-buffer," Technical Report 242/94, Department of Computer Science, University of Kaiserslautern, Kaiserslautern, Germany, Apr. 1994.

[47] S. Heinrich and A. Keller, "Quasi-Monte Carlo methods in computer graphics, part II: The radiance equation," Technical Report 243/94, Department of Computer Science, University of Kaiserslautern, Kaiserslautern, Germany, Apr. 1994.

[48] J. Hoogenboom, *Adjoint Monte Carlo Methods in Neutron Transport Calculations*. PhD thesis, Technical University Delft, The Netherlands, 1977.

[49] D. Immel, M. Cohen, and D. Greenberg, "A radiosity method for non-diffuse environments," *Computer Graphics*, vol. 20, pp. 133–142, Aug. 1986.

[50] H. Jensen, "Importance driven path tracing using the photon map," in *Proceedings of the Sixth Eurographics Workshop on Rendering*, (Dublin, Ireland), pp. 359–369, June 1995.

[51] H. Jensen and N. Christensen, "Optimising path tracing using noise reduction filters," in *Proceedings of WSCG 95*, (Pilsen, Czech Republic), pp. 134–142, Feb. 1995.

[52] H. Jensen and N. Christensen, "Photon maps in bidirectional monte carlo ray tracing of complex objects," *Computers & Graphics*, vol. 19, no. 2, pp. 215–224, 1995.

[53] J. Kajiya, "The rendering equation," *Computer Graphics*, vol. 20, pp. 143–150, Aug. 1986.

[54] M. Kalos and P. Whitlock, *Monte Carlo Methods.* New York: Wiley & Sons, 1986.

[55] A. Keller, "A quasi-Monte Carlo algorithm for the global illumination problem in the radiosity setting," Technical Report 260/94, Department of Computer Science, University of Kaiserslautern, Kaiserslautern, Germany, Apr. 1994.

[56] D. Kirk and J. Arvo, "Unbiased sampling techniques for image synthesis," *Computer Graphics*, vol. 25, pp. 153–156, July 1991.

[57] D. Kirk and J. Arvo, "Unbiased variance reduction for global illumination," in *Proceedings of the Second Eurographics Workshop on Rendering*, (Barcelona, Spain), May 1991.

[58] A. Kok and F. Jansen, "Source selection for the direct lighting computation in global illumination," in *Proceedings of the Second Eurographics Workshop on Rendering*, (Barcelona, Spain), May 1991.

[59] A. Kok, "Grouping of patches in progressive radiosity," in *Proceedings of the Fourth Eurographics Workshop on Rendering*, (Paris, France), pp. 221–232, June 1993.

[60] A. Kok, *Ray Tracing and Radiosity Algorithms for Photorealistic Image Synthesis.* PhD thesis, Technical University Delft, The Netherlands, May 1994.

[61] C. Kolb, D. Mitchell, and P. Hanrahan, "A realistic camera model for computer graphics," *Computer Graphics*, vol. 29, pp. 317–324, Aug. 1995.

[62] E. Lafortune and Y. Willems, "Bi-directional path tracing," in *Proceedings of CompuGraphics*, (Alvor, Portugal), pp. 145–153, Dec. 1993.

[63] E. Lafortune and Y. Willems, "Hierarchical and adaptive meshing with linear interpolation of vertex radiosities," in *Proceedings of ICCG*, (Bombay, India), pp. 71–78, Feb. 1993.

[64] E. Lafortune and Y. Willems, "The ambient term as a variance reducing technique for Monte Carlo ray tracing," in *Proceedings of the Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 163–171, June 1994.

[65] E. Lafortune and Y. Willems, "A theoretical framework for physically based rendering," *Computer Graphics Forum*, vol. 13, pp. 97–107, June 1994.

[66] E. Lafortune and Y. Willems, "Using the modified phong BRDF for physically based rendering," Technical Report CW197, Department of Computer Science, Katholieke Universiteit Leuven, Leuven, Belgium, Nov. 1994.

[67] E. Lafortune and Y. Willems, "A 5D tree to reduce the variance of Monte Carlo ray tracing," in *Proceedings of the Sixth Eurographics Workshop on Rendering*, (Dublin, Ireland), pp. 11–20, June 1995.

[68] E. Lafortune and Y. Willems, "Reducing the number of shadow rays in bidirectional path tracing," in *Proceedings of WSCG 95*, (Pilsen, Czech Republic), pp. 384–392, Feb. 1995.

[69] B. Lange, "The simulation of radiant light transfer with stochastic ray-tracing," in *Proceedings of the Second Eurographics Workshop on Rendering*, (Barcelona, Spain), May 1991.

[70] E. Languénou, K. Bouatouch, and M. Chelle, "Global illumination in presence of participating media with general properties," in *Proceedings of the Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 69–85, June 1994.

[71] M. Lee, R. Redner, and S. Uselton, "Statistically optimized sampling for distributed ray tracing," *Computer Graphics*, vol. 19, pp. 61–67, July 1985.

[72] R. Lewis, "Making shaders more physically plausible," in *Proceedings of the Fourth Eurographics Workshop on Rendering*, (Paris, France), pp. 47–62, June 1993.

[73] R. Lewis, "Making shaders more physically plausible," *Computer Graphics Forum*, vol. 13, pp. 109–120, June 1994.

[74] D. Lischinski, F. Tampieri, and D. Greenberg, "Discontinuity meshing for accurate radiosity," *IEEE Computer Graphics and Applications*, vol. 12, pp. 25–39, Nov. 1992.

[75] G. Márton and L. Szirmay-Kalos, "On average-case complexity of ray tracing algorithms," in *Proceedings of WSCG 95*, (Pilsen, Czech Republic), pp. 187–196, Feb. 1995.

[76] N. Max, "Efficient light propagation for multiple anisotropic volume scattering," in *Proceedings of the Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 87–104, June 1994.

[77] N. Metropolis and S. Ulam, "The Monte Carlo method," *J. Amer. Stat. Assoc.*, vol. 44, no. 247, pp. 335–341, 1949.

[78] G. Meyer, H. Rushmeier, M. Cohen, D. Greenberg, and K. Torrance, "An experimental evaluation of computer graphics imagery," *ACM Transactions on Graphics*, vol. 5, no. 1, pp. 30–50, 1986.

[79] D. Mitchell, "Spectrally optimal sampling for distribution ray tracing," *Computer Graphics*, vol. 25, pp. 157–164, July 1991.

[80] D. Mitchell, "Ray tracing and irregularities of distribution," in *Proceedings of the Third Eurographics Workshop on Rendering*, (Bristol, UK), pp. 61–69, May 1992.

[81] L. Neumann, M. Feda, M. Kopp, and W. Purgathofer, "A new stochastic radiosity method for highly complex scenes," in *Proceedings of the Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 195–206, June 1994.

[82] C. Patmore, "Illumination of dense foliage models," in *Proceedings of the Fourth Eurographics Workshop on Rendering*, (Paris, France), pp. 63–71, June 1993.

[83] S. Pattanaik, *Computational Methods for Global Illumination and Visualisation of Complex 3D Environments*. PhD thesis, NCST Birla Institute of Technology & Science, Pilani, India, Feb. 1993.

[84] S. Pattanaik, "The mathematical framework of adjoint equations for illumination computation," in *Proceedings of ICCG*, (Bombay, India), pp. 265–288, Feb. 1993.

[85] S. Pattanaik and K. Bouatouch, "Adjoint equations and particle tracing for global illumination," Publication Interne 903, Programme 4, IRISA, Rennes Cedex, France, Jan. 1995.

[86] S. Pattanaik and S. Mudur, "Computation of global illumination by Monte Carlo simulation of the particle model of light," in *Proceedings of the Third Eurographics Workshop on Rendering*, (Bristol, UK), pp. 71–83, May 1992.

[87] S. Pattanaik and S. Mudur, "Efficient potential equation solutions for global illumination computation," *Computers and Graphics*, vol. 17, no. 4, pp. 387–396, 1993.

[88] S. Pattanaik and S. Mudur, "The potential equation and importance in illumination computations," *Computer Graphics Forum*, vol. 12, pp. 131–136, June 1993.

[89] S. Pattanaik and S. Mudur, "Adjoint equations and random walks for illumination computation," *ACM Transactions on Graphics*, vol. 14, no. 1, pp. 77–102, 1995.

[90] W. Purgathofer, "A statistical model for adaptive stochastic sampling," in *Proceedings of Eurographics '86*, (Lisbon, Portugal), pp. 145–152, Aug. 1986.

[91] E. Reinhard, L. Thijssen, and F. Jansen, "Environment mapping for efficient sampling of the diffuse interreflection," in *Proceedings of the Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 401–414, June 1994.

[92] C. Rozé, B. Maheu, and G. Gréhan, "Evaluations of the sighting distance in a foggy atmosphere by Monte Carlo simulation," *Atmospheric Environment*, vol. 28, no. 5, pp. 769–775, 1994.

[93] H. Rushmeier, *Realistic Image Synthesis for Scenes with Radiatively Participating Media*. PhD thesis, The Sibley School of Mechanical and Aerospace Engineering, Cornell University, 1988.

[94] H. Rushmeier, "Rendering participating media: Problems and solutions from application areas," in *Proceedings of the Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 35–55, June 1994.

[95] H. Rushmeier, Ch. Patterson, and A. Veerasamy, "Geometric simplification for indirect illumination calculations," in *Proceedings of Graphics Interface '93*, (Toronto, Ontario), pp. 227–236, May 1993.

[96] H. Rushmeier, G. Ward, C. Piatko, P. Sanders, and B. Rust, "Comparing real and synthetic images: Some ideas about metrics," in *Proceedings of the Sixth Eurographics Workshop on Rendering*, (Dublin, Ireland), pp. 213–222, June 1995.

[97] H. Rushmeier and G. Ward, "Energy preserving non–linear filters," *Computer Graphics*, vol. 28, pp. 131–138, July 1994.

[98] D. Salesin, D. Lischinski, and T. DeRose, "Reconstructing illumination functions with selected discontinuities," in *Proceedings of the Third Eurographics Workshop on Rendering*, (Bristol, UK), pp. 99–112, May 1992.

[99] Ch. Schlick, "An adaptive sampling technique for multidimensional integration by ray-tracing," in *Proceedings of the Second Eurographics Workshop on Rendering*, (Barcelona, Spain), May 1991.

[100] Ch. Schlick, "A customizable reflectance model for everyday rendering," in *Proceedings of the Fourth Eurographics Workshop on Rendering*, (Paris, France), pp. 73–83, June 1993.

[101] Ch. Schlick, "Quantization techniques for visualization of high dynamic range pictures," in *Proceedings of the Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 7–18, June 1994.

[102] Ch. Schlick, "A survey of shading and reflectance models," *Computer Graphics Forum*, vol. 13, pp. 121–131, June 1994.

[103] P. Schröder, "Numerical integration for radiosity in the presence of singularities," in *Proceedings of the Fourth Eurographics Workshop on Rendering*, (Paris, France), pp. 177–184, June 1993.

[104] P. Schröder, S. Gortler, M. Cohen, and P. Hanrahan, "Wavelet projections for radiosity," in *Proceedings of the Fourth Eurographics Workshop on Rendering*, (Paris, France), pp. 105–114, June 1993.

[105] P. Schröder and P. Hanrahan, "On the form factor between two polygons," *Computer Graphics*, vol. 27, pp. 163–164, Aug. 1993.

[106] P. Shirley, *Physically Based Lighting Calculations for Computer Graphics*. PhD thesis, University of Illinois, Nov. 1990.

[107] P. Shirley, "A ray tracing method for illumination calculation in diffuse-specular scenes," in *Proceedings of Graphics Interface '90*, (Halifax, Nova Scotia), May 1990.

[108] P. Shirley, "Discrepancy as a quality measure for sample distributions," in *Proceedings of Eurographics '91*, (Vienna, Austria), pp. 183–194, Sept. 1991.

[109] P. Shirley, "Nonuniform random point sets via warping," in *Graphics Gems III* (D. Kirk, ed.), pp. 80–83, San Diego: Academic Press, 1992.

[110] P. Shirley and K. Chiu, "Notes on adaptive quadrature on the hemisphere," Technical Report 411, Department of Computer Science, Indiana University, Bloomington, Indiana, 1994.

[111] P. Shirley, B. Wade, Ph. Hubbard, D. Zareski, D. Walter, and D. Greenberg, "Global illumination via density-estimation," in *Proceedings of the Sixth Eurographics Workshop on Rendering*, (Dublin, Ireland), pp. 187–199, June 1995.

[112] P. Shirley and C. Wang, "Direct lighting by Monte Carlo integration," in *Proceedings of the Second Eurographics Workshop on Rendering*, (Barcelona, Spain), May 1991.

[113] P. Shirley, C. Wang, and K. Zimmerman, "Monte Carlo techniques for direct lighting calculations," *ACM Transactions on Graphics*, vol. 15, no. 1, 1996.

[114] Y. Shreider, ed., *The Monte Carlo Method*. Oxford: Pergamon Press, 1966.

[115] F. Sillion, "Clustering and volume scattering for hierarchical radiosity calculations," in *Proceedings of the Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 105–117, June 1994.

[116] F. Sillion, J. Arvo, S. Westin, and D. Greenberg, "A global illumination solution for general reflectance distributions," *Computer Graphics*, vol. 25, pp. 187–196, July 1991.

[117] F. Sillion and C. Puech, "A general two-pass method integrating specular and diffuse reflection," *Computer Graphics*, vol. 23, pp. 335–344, July 1989.

[118] F. Sillion and C. Puech, *Radiosity and Global Illumination*. San Francisco: Morgan Kaufmann, 1994.

[119] B. Smits, J. Arvo, and D. Salesin, "An importance-driven radiosity algorithm," *Computer Graphics*, vol. 26, pp. 273–282, Aug. 1992.

[120] J. Spanier, "A new family of estimators for random walk problems," *J. Inst. Maths Applics*, vol. 23, pp. 1–31, Jan. 1979.

[121] J. Spanier and E. Maize, "Quasi-random methods for estimating integrals using relatively small samples," *SIAM Review*, vol. 36, pp. 18–44, Mar. 1994.

[122] F. Tampieri and D. Lischinski, "The constant radiosity assumption syndrome," in *Proceedings of the Second Eurographics Workshop on Rendering*, (Barcelona, Spain), May 1991.

[123] H. Trotter and J. Tukey, "Conditional Monte Carlo for normal samples," in *Symposium on Monte Carlo Methods*, (University of Florida), pp. 64–79, 1954.

[124] J. Tumblin and H. Rushmeier, “Tone reproduction for realistic images,” *IEEE Computer Graphics and Applications*, vol. 13, pp. 42–48, Nov. 1993.

[125] E. Veach and L. Guibas, “Bidirectional estimators for light transport,” in *Proceedings of the Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 147–162, June 1994.

[126] E. Veach and L. Guibas, “Optimally combining sampling techniques for Monte Carlo rendering,” *Computer Graphics*, vol. 29, pp. 419–428, Aug. 1995.

[127] J. Wallace, M. Cohen, and D. Greenberg, “A two-pass solution to the rendering equation: a synthesis of ray tracing and radiosity methods,” *Computer Graphics*, vol. 21, pp. 311–320, July 1987.

[128] J. Wallace, K. Elmquist, and E. Haines, “A ray tracing algorithm for progressive radiosity,” *Computer Graphics*, vol. 23, pp. 315–324, July 1989.

[129] G. Ward, “Adaptive shadow testing for ray tracing,” in *Proceedings of the Second Eurographics Workshop on Rendering*, (Barcelona, Spain), May 1991.

[130] G. Ward, “Real pixels,” in *Graphics Gems II* (J. Arvo, ed.), pp. 80–83, San Diego: Academic Press, 1991.

[131] G. Ward, “Measuring and modeling anisotropic reflection,” *Computer Graphics*, vol. 26, pp. 265–272, July 1992.

[132] G. Ward, “The RADIANCE lighting simulation and rendering system,” *Computer Graphics*, vol. 28, pp. 459–472, July 1994.

[133] G. Ward and P. Heckbert, “Irradiance gradients,” in *Proceedings of the Third Eurographics Workshop on Rendering*, (Bristol, UK), pp. 85–98, May 1992.

[134] G. Ward, F. Rubinstein, and R. Clear, “A ray tracing solution for diffuse interreflection,” *Computer Graphics*, vol. 22, pp. 85–92, July 1988.

[135] T. Whitted, “An improved illumination model for shaded display,” *Communications of the ACM*, vol. 23, no. 6, pp. 343–349, 1980.

[136] S. Zaremba, “The mathematical basis of Monte Carlo and quasi-Monte Carlo methods,” *SIAM Review*, vol. 10, pp. 303–314, July 1968.

[137] K. Zimmerman and P. Shirley, “A two-pass solution to the rendering equation with a source visibility preprocess,” in *Proceedings of the Sixth Eurographics Workshop on Rendering*, (Dublin, Ireland), pp. 303–314, June 1995.