Chapter 1

Introduction

The goal of this dissertation is to develop robust, general-purpose algorithms for solving light transport problems. To meet our goal of generality, we concentrate on Monte Carlo methods. Currently, only Monte Carlo approaches can handle the wide range of surface geometries, reflection models, and lighting effects that occur in real environments. By a *robust* algorithm, we mean one that produces an acceptably accurate output for as wide a range of inputs as possible. In this dissertation, we make substantial progress toward these goals, by developing new theoretical models, statistical methods, and rendering algorithms. We also investigate what cannot be achieved — the inherent limitations of certain approaches to light transport.

Despite a great deal of research, current light transport methods are fairly limited in their capabilities. They are optimized for a very restricted class of input models, and typically require a huge increase in resources to handle other types of inputs. For example, they often have problems on scenes with strong indirect lighting, or scenes where most surfaces are non-diffuse. These are not pathological examples by any means, and in fact there is considerable interest in solving these cases well (e.g. in architectural applications).

For light transport algorithms to be widely used, it is important to find techniques that are less fragile. Rendering algorithms must run within acceptable time bounds on real models, yielding images that are physically plausible and visually pleasing. They must support complex geometry, materials, and illumination, since these are all important components of real environments.

In our research, we seek to develop algorithms with reasonable, predictable performance over the widest possible range of real models. Because we have chosen to focus on Monte Carlo approaches, which support complex geometry and materials with relative ease, our main interest is to develop algorithms that can handle complex *illumination* efficiently. This includes features such as glossy surfaces, concentrated indirect lighting, small geometric objects, and caustics, all of which cause problems for a wide variety of current rendering algorithms. Our goal is to find general-purpose algorithms that handle these difficult cases well, without special treatment; in other words, light transport algorithms that are robust.

In the following sections, we start with an overview of the light transport problem and why it is important. We also discuss our assumptions about the transport model (discussed in more detail in Section 1.5). After this brief introduction, we summarize the original contributions of this dissertation, and outline its organization.

In the rest of the chapter, we step back to see how these results fit into a larger context. In Section 1.4 we give a high-level view of the various types of light transport algorithms used in graphics, and explain the advantages of unbiased Monte Carlo algorithms. In Section 1.5, we consider the various phenomena that occur with real light (such as diffraction), and the reasons why these phenomena are easy or difficult to simulate. Finally, in Section 1.6 we look at problems from physics and engineering that are closely related to light transport. The viewpoints in these other fields are often quite different from one another, which has led to a variety of different solution techniques for problems that are actually quite similar.

1.1 The light transport problem

In computer graphics, the simulation of light transport is a tool that helps us to create convincing images of an artificial world. We are given a description of the environment, including the geometry and scattering properties of the surfaces. We are also given a description of the light sources, and the viewpoints from which images should be generated. Light transport algorithms then simulate the physics of this world, in order to generate realistic, accurate images.

1.1.1 Why light transport is important

One of the main goals of light transport algorithms is to increase human efficiency in the modeling of realistic virtual environments. In computer animations, for example, a great deal of effort is currently spent on designing realistic lighting. The main problem is that the algorithms used for production work (such as scan-line rendering and ray tracing) do not have the ability to simulate indirect lighting. Thus indirect illumination does not happen automatically when lights are placed: instead, it must be imitated by carefully placing additional lights. If we could find robust light transport algorithms, then the indirect lighting could be computed automatically, which would make the lighting task far easier.

Another important application of light transport is *predictive modeling*, where we wish to predict the appearance of objects before they are built. This idea has obvious uses in architecture and product design. For these applications, it is important that the results be objectively accurate, as well as visually pleasing.

Finally, better techniques for light transport in graphics may lead to better methods in physics and engineering, because light transport has a structure that is similar to radiation and particle transport problems. Section 1.6 discusses these possibilities in detail.

If robust light transport algorithms can be found, it seems inevitable that they will be widely used. This would continue a trend for computer software in general, whereby algorithms that are simpler or more powerful are eventually favored over those designed for efficiency in special cases. We feel that the benefits of accurate light transport simulations will soon outweigh their moderate computational costs.

1.1.2 Assumptions about the transport model

Light transport algorithms do not simulate the behavior of light in every detail, since this is not necessary for most applications.[1] From a graphics standpoint, physical optics is best thought of as a menu of options. For each application, we decide which optical effects are important, and choose an algorithm that can simulate them.

[1] Strictly speaking, it is not even possible, since the laws of physics are not completely known. However, the theory of light and its interaction with matter is one of the best that physics has to offer, and can predict virtually every observed phenomenon with great accuracy [Feynman 1985]. For the purposes of computer graphics, we can assume that these laws are completely understood.

In our work, we generally assume a geometric optics model. Light is emitted, scattered, and absorbed only at surfaces, and travels along straight lines between these surfaces. Thus, we do not allow participating media such as clouds or smoke, or media with a continuously varying index of refraction (e.g. heated air). We also ignore most properties of light that depend on a wave or quantum model for their explanation (e.g. diffraction or fluorescence). In particular, we ignore the possibility of interference between light beams, i.e. light is assumed to be perfectly incoherent.

In normal environments, the effects we have ignored are not very significant. Geometric optics is adequate to model almost everything we see around us, to a high degree of accuracy. For this reason, virtually all light transport algorithms in graphics are based on assumptions similar to those above. Later in this chapter, we will investigate some of the other choices that could have been made (see Sections 1.5 and 1.6).

1.2 Summary of original contributions

Our contributions fall into three areas: new theoretical models, new statistical methods, and new rendering algorithms. We give an overview of each of these areas, and then discuss our results in detail.

The first part of this dissertation investigates the theory of bidirectional light transport algorithms. We have developed light transport models that are simple, mathematically precise, and reveal the structure of the light transport problem in useful ways. In particular we have studied the relationships between different bidirectional solution techniques (e.g. those based on light and importance) under different assumptions about the physical validity of the scene model. These new light transport formulations have led directly to new insights and rendering techniques.

Statistical methods are another vital component of Monte Carlo algorithms. In the process of investigating light transport algorithms, we have developed new general-purpose methods for variance reduction. We isolated these techniques and presented them in an abstract setting, since we believe that they will be useful in other contexts.

Finally, our main contribution has been the development of robust light transport algorithms. The principal advantage of these algorithms is their ability to handle complex illumination efficiently. Because of their Monte Carlo nature, they also support complex scattering models and surface geometries. The combination of these properties allows a wide variety of realistic scenes to be rendered in a reasonable, predictable amount of time, even when there is difficult indirect illumination.

1.2.1 Bidirectional light transport theory in computer graphics

A general linear operator formulation. We present a simple light transport model based on linear operators, extending the work of Arvo [1995]. This new formulation unifies light transport, importance transport, and particle tracing, and concisely summarizes the relationships among them. We do not make any assumptions about the physical validity of the scene model, which gives our framework a richer structure than previous approaches.

New examples of non-symmetric scattering. Certain materials must be treated specially in light transport algorithms, namely those whose *bidirectional scattering distribution function* (BSDF) is not symmetric. We discuss two common examples of this that have not been previously recognized. Specifically, we show that non-symmetric scattering occurs whenever light is refracted, and also whenever shading normals are used. We derive the transformations required to handle these situations correctly in bidirectional algorithms. We also show that if these new transformations are not used, there can be substantial errors and image artifacts.

A reciprocity principle for general materials. It is well known that the reflection of light from physically valid materials is described by a symmetric BSDF. We derive a generalization of this condition that holds for arbitrary materials (i.e. for transmission as well as reflection). We establish this new reciprocity principle using the laws of thermodynamics, in particular Kirchhoff's laws and the principle of detailed balance. We also discuss the historical origins of reciprocity principles, the subtleties involved in their justification, and the conditions under which they are valid.

A self-adjoint operator formulation. Taking advantage of this new reciprocity principle, we propose the first light transport formulation in which the linear operators are *self-adjoint* (symmetric) for all physically valid scenes. We show that this simplifies both the theory and implementation of bidirectional light transport algorithms.

The path integral formulation. Usually the light transport problem is expressed in terms of integral equations or linear operators. Instead, we show how to formulate it as an integration problem over a space of paths. This viewpoint allows new solution techniques to be applied, such as multiple importance sampling, or the Metropolis sampling algorithm.

The inherent limitations of unbiased Monte Carlo methods. We show that certain kinds of transport paths cannot be generated by standard sampling techniques. This implies that the images generated by unbiased Monte Carlo algorithms (such as path tracing) can be missing certain lighting effects. We analyze the conditions under which this occurs, and propose methods for making these path sampling algorithms complete.

1.2.2 General-purpose Monte Carlo techniques

Multiple importance sampling. We describe a new technique for constructing estimators that are robust, i.e. whose variance is low for a broad class of integrands. It is based on the idea of using more than one sampling technique to evaluate an integral, where each technique is designed to sample some feature of the integrand that might otherwise lead to high variance. Our key results are on how to combine the samples: we present combination strategies that are provably close to optimal, compared to any other unbiased method. This leads to low-variance estimators that are useful in a variety of problems in graphics, including distribution ray tracing, multi-pass radiosity algorithms, and bidirectional path tracing.

Efficiency-optimized Russian roulette. Russian roulette is a technique that reduces the average cost of sampling, but increases variance. We propose a new optimization that trades off one property against the other, in order to maximize the efficiency of the resulting estimator. This is particularly useful in the context of visibility tests, where often there are many samples that only make a small contribution.

1.2.3 Robust light transport algorithms

Bidirectional path tracing. We propose a new light transport algorithm based on the idea of using a family of different sampling techniques for paths, and then combining them using multiple importance sampling. Each path is generated by connecting two independently generated subpaths, one starting from the light sources and the other starting from the eye. By varying the lengths of the light and eye subpaths, we obtain a family of different sampling techniques. We show that each technique can efficiently sample different kinds of paths, and that these paths are responsible for different lighting effects in the final image. By combining samples from all of these techniques using multiple importance sampling, a wide range of different lighting effects can be handled efficiently.

We describe the complete set of bidirectional estimators, including the important special cases where the light or eye subpath has at most one vertex. We also discuss extensions for handling ideal specular surfaces, arbitrary path lengths, and efficient visibility testing.

Metropolis light transport. We propose a new Monte Carlo approach to the light transport problem, inspired by the Metropolis sampling method in computational physics. To render an image, we generate a sequence of light transport paths by randomly mutating a single current path (e.g. a mutation might add a new vertex to the path). Each mutation is accepted or rejected with a carefully chosen probability, to ensure that paths are sampled according to the contribution they make to the desired final image. In this way we construct a random walk over the space of transport paths, such that an unbiased image can be formed by simply recording the locations of these paths on the image plane.

This algorithm is unbiased, handles general geometric and scattering models, uses little storage, and can be orders of magnitude more efficient than previous unbiased approaches. It performs especially well on problems that are usually considered difficult, e.g. those involving bright indirect light, small geometric holes, or glossy surfaces. Furthermore, it is competitive with previous unbiased algorithms even on scenes with relatively simple illumination.

The key advantage of the Metropolis approach is that the path space is explored locally, by favoring mutations that make small changes to the current path. This has several consequences. First, the average cost per sample is small (typically only one or two rays). Second, once an important path is found, the nearby paths are explored as well, thus amortizing the expense of finding such paths over many samples. Third, the mutation set is easily extended. By constructing mutations that preserve certain properties of the path (e.g. which light source is used) while changing others, we can exploit various kinds of coherence in the scene. It is often possible to handle difficult lighting problems efficiently by designing a specialized mutation in this way.

To our knowledge, this is the first application of the Metropolis algorithm to transport problems of any kind.

1.3 Thesis organization

The first two chapters consist of introductory and background material. In the rest of Chapter 1, we discuss the advantages and disadvantages of various types of light transport algorithms, we examine the range of optical phenomena that can be simulated by such algorithms, and we compare light transport to similar problems in other fields. In Chapter 2, we give an introduction to Monte Carlo integration, including a survey of the variance reduction techniques that have proven most useful in computer graphics.

The remainder of the dissertation is divided into two parts. In the first part, we describe new theoretical models for bidirectional light transport algorithms. Chapter 3 develops the concepts of radiometry and gives an introduction to the standard light transport equations. It also describes a new measure-theoretic basis for defining radiometric quantities. Chapter 4 presents a new light transport model based on linear operators. This formulation does not make any assumptions about the physical validity of the scene model. Chapter 5 investigates the situations where this model is necessary, i.e. materials whose scattering properties are not symmetric. We give both physical and non-physical examples of such materials, and we derive the techniques needed to handle these materials correctly in bidirectional algorithms.

In Chapter 6, we investigate how the scattering of light from materials is constrained by the laws of physics, and we derive a new reciprocity principle for general materials. In Chapter 7, this principle is used to construct the first light transport framework where light, importance, and particles obey the same transport equations for any physically valid scene. Finally, Chapter 8 describes the path integral framework, which forms the basis of our new light transport algorithms.

The second part of the dissertation is more practical in nature. Chapter 9 describes *multiple importance sampling*, a general tool for reducing the variance of Monte Carlo integration. In Chapter 10, we apply this tool to the path integral framework, to obtain the *bidirectional path tracing* algorithm. Finally, Chapter 11 builds upon the path integral framework in a different way, by combining it with a well-known sampling technique from computational physics to obtain the *Metropolis light transport* algorithm.

1.4 Light transport algorithms

Within the field of computer graphics, many different algorithms have been proposed for solving the light transport problem. In this dissertation, we have chosen to focus on unbiased, view-dependent, Monte Carlo algorithms. We first mention the various kinds of algorithms that have been proposed, and then discuss the choices we have made.

1.4.1 A brief history

Light transport algorithms can be roughly divided into two groups: Monte Carlo methods, and finite element methods.

Monte Carlo methods have been used for neutron transport problems since the 1950's [Albert 1956], and have been studied extensively there [Spanier & Gelbard 1969]. In graphics Monte Carlo methods arose independently, starting with Appel [1968] who computed images using random particle tracing. Whitted [1980] introduced ray tracing (the recursive evaluation of surface appearance), and also suggested the idea of randomly perturbing viewing rays. Cook et al. [1984] implemented this idea and extended it to random sampling of light sources, lenses, and time. This led to the first complete, unbiased Monte Carlo transport algorithm as proposed by Kajiya [1986], who recognized that the problem could be written as an integral equation, and could be evaluated by sampling paths. Since then, many refinements to his *path tracing* technique have been adapted from the particle transport literature [Arvo & Kirk 1990].

There has also been a great deal of work on biased Monte Carlo algorithms, which are often more efficient than path tracing. These include the *irradiance caching* algorithm of Ward et al. [1988], the *density estimation* method of [Shirley et al. 1995], and the *photon map* approach of [Jensen 1995].

Finite element methods for light transport were originally adapted from the radiative heat transfer literature. Goral et al. [1984] introduced these methods to the graphics community, where they are typically known as *radiosity algorithms*. Many improvements have been made to the basic radiosity method, including substructuring [Cohen et al. 1986], progressive refinement [Cohen et al. 1988], hierarchical basis functions [Hanrahan et al. 1991], importance-driven refinement [Smits et al. 1992], discontinuity meshing [Lischinski et al. 1992], wavelet methods [Gortler et al. 1993], and clustering [Smits et al. 1994]. Other extensions include the handling of participating media [Rushmeier & Torrance 1987], and finite element methods for non-diffuse surfaces [Immel et al. 1986, Sillion et al. 1991, Aupperle & Hanrahan 1993, Schröder & Hanrahan 1994].

Methods have also been proposed that combine features of Monte Carlo and finite element approaches. Typically, these take the form of *multi-pass methods*, which combine radiosity and ray tracing passes in order to handle more general scene models [Wallace et al. 1987, Sillion & Puech 1989, Chen et al. 1991]. Another approach is *Monte Carlo radiosity*, where the solution is represented as a linear combination of basis functions (as with finite element methods), but where the coefficients are estimated by tracing random light particles [Shirley 1990b, Pattanaik & Mudur 1993, Pattanaik & Mudur 1995].

1.4.2 Monte Carlo vs. deterministic approaches

At the most basic level, a Monte Carlo algorithm uses random numbers, while a deterministic algorithm does not. However, in practice algorithms often use a mixture of techniques, and are not easily classified. The distinction is further blurred by issues that have nothing to do with random numbers *per se*, but that are often associated with one type of algorithm or the other. We discuss some of these differences below.

First, Monte Carlo algorithms are usually more general. This is a very important issue, since the biggest source of error in light transport calculations is often the scene model itself. A key advantage of Monte Carlo approaches is that virtually any environment can be modeled accurately. With deterministic algorithms, on the other hand, there are often severe restrictions on the allowable geometry (e.g. limited to polygons) and materials (e.g. limited to ideal diffuse reflectors).[2] With these restrictions, it is difficult or impossible to model real environments. To use these methods, we must usually resort to solving a different problem, by modifying the scene model. Any claims about the solution "accuracy" under these circumstances are misleading at best.

Monte Carlo and deterministic approaches are also distinguished by how they access the scene model. Deterministic algorithms usually work with explicit representations of the scene and its properties (e.g. lists of polygons). Thus, they are strongly affected by the size and complexity of the scene representation. On the other hand, Monte Carlo algorithms are based on sampling, which means that the scene model is accessed through a small set of queries (e.g. what is the first surface point intersected by a given ray?). This interface hides the scene complexity behind a layer of abstraction, and means that rendering times are only loosely coupled to the scene representation (for example, the scene complexity may affect the time required for ray casting). In effect, Monte Carlo algorithms can sample the scene to determine the information they actually need, while most deterministic algorithms are designed to examine every detail, whether it is relevant or not.

This is an especially important issue for robustness: ideally, the performance of light transport algorithms should depend only on *what* the scene represents, rather than the details of *how* it is represented. For example, consider a scene illuminated by a square area light source. If this light source is replaced with a 10 by 10 grid of point sources, then the visual results will be nearly identical. However, the performance of many light transport algorithms will be much worse in the second case. Similarly, suppose that we replace the same source by a pair of fluorescent bulbs covered by a translucent panel. In this case the

[2] Even when deterministic algorithms support "general" surfaces and reflection models, their form is often quite limited (e.g. polynomial functions of a prespecified maximum degree). This demands an extra approximation step when modeling the scene, and often this approximation is very bad and/or expensive in some cases (e.g. for glossy surfaces).

entire scene is illuminated indirectly, which will cause problems for many algorithms. Ideally, rendering algorithms should not be sensitive to cosmetic changes of this sort. The same comments apply to geometric complexity: whether an object is represented as a thousand polygons or a million Bezier patches, we would like the rendering times to be as similar as possible. Monte Carlo algorithms at least have the potential to deal with these situations effectively, because they are based on sampling.

The distinction between Monte Carlo and deterministic methods is somewhat blurred by the fact that Monte Carlo algorithms place very weak restrictions on the "randomness" of the numbers they use (e.g. often the only requirement is that these numbers are uniformly distributed). It is usually possible to design fixed sampling patterns which satisfy the same restrictions, and this often leads to better performance (these are called *quasi-Monte Carlo methods* [Niederreiter 1992]). The principle of Monte Carlo methods is not that the samples are truly random, but that random samples *could* be used in their place.

1.4.3 View-dependent vs. view-independent algorithms

The purpose of all light transport algorithms in computer graphics is to produce images, i.e. rectangular arrays of color values, suitable for display on a monitor or printing device. A *view-independent* algorithm is one that computes an intermediate representation of the solution, from which arbitrary views can be generated very quickly. Any other algorithm is *view-dependent*, which can mean one of several things. *Importance-driven* methods compute a solution that is defined globally, but is optimized for a particular view. That is, the solution is detailed in the visible portions of the scene, but it may be very coarse elsewhere. *Multi-pass* methods compute a global solution that is valid for all views, but where the final rendering step to obtain an image is relatively slow (e.g. it requires ray tracing). Finally, *image space* methods compute an image directly from the scene model, without trying to represent the solution everywhere. This category includes Monte Carlo algorithms such as path tracing.

The distinction between view-dependent and view-independent methods raises a number of interesting issues. First, these two types of algorithms generally have different purposes. View-dependent methods are useful for animations, where the scene model can change substantially from one frame to the next. They are also the natural choice for rendering still images. On the other hand, view-independent solutions are useful for interactive applications, such as architectural walkthroughs or computer games.

One problem with "view-independent" algorithms is that they do not make any guarantees about the error for any particular view. Ideally, these algorithms would ensure that every view has a small error. Instead, error is usually measured globally (averaged over the entire scene), which implies that local errors can still be large. This means that if we render an image of a region where the view-independent solution is particularly bad, the results can be completely wrong.

Another problem with view-independent solutions is that they are often more expensive than view-dependent ones, because they compute a representation of the full solution (essentially solving for all views simultaneously). When non-diffuse materials are allowed, this can be a great deal of extra work compared to computing a single view, since the appearance of glossy surfaces changes rapidly with the viewpoint.

Even if only diffuse surfaces are allowed, view-dependent algorithms are often more efficient, since they only need to compute the portion of the solution that we are interested in. For example, if the scene model is complex, and only a small part of it is visible, then it can be much more efficient to compute an image directly. Image space algorithms have the greatest potential here, since importance-driven methods do not scale as well to complex scenes (where it can be very expensive to compute even a coarse solution over the whole domain).

The difference between view-dependent and view-independent algorithms is actually not as large as it might appear at first, since it is often possible to convert a view-dependent algorithm into view-independent one. The similarity is that both types of algorithms compute a finite set of linear measurements of the global solution. For a view-dependent algorithm, these measurements are pixel values: each pixel is defined by integrating the light falling on a small region of the image plane. This is closely related to the view-independent approach, where the solution is usually represented as a linear combination of basis functions. View-dependent algorithms can often be adapted to estimate the coefficients of these basis functions, rather than the pixel values of an image, since they are both defined as linear measurements.

1.4.4 Unbiased vs. consistent Monte Carlo algorithms

A Monte Carlo estimator computes a value $F_N(X_1, \ldots, X_N)$ that is supposed to approximate some unknown quantity Q. Typically, Q is a parameter of a known density function p, and the X_i are random samples from p. The quantity $F_N - Q$ is called the *error*, and its expected value $\beta[F_N] = E[F_N - Q]$ is called the *bias*. The estimator is *unbiased* if $\beta[F_N] = 0$ for all sample sizes N, while it is *consistent* if the error $F_N - Q$ goes to zero with probability one as N approaches infinity [Kalos & Whitlock 1986].

Intuitively, an unbiased estimator computes the correct answer, on average. A biased estimator computes the wrong answer, on average. However, if a biased estimator is also consistent, then the average error can be made arbitrarily small by increasing the sample size.

We argue that unbiased estimators are essential in order for light transport calculations to be robust. This is an important point, since many algorithms used in graphics are merely consistent.

The basic reason for preferring unbiased algorithms is that they make it far easier to estimate the error in a solution. To have any confidence in the computed results, we must have some estimate of this error. For unbiased algorithms, this simply involves computing the sample variance, since any error is guaranteed to show up as random variation among the samples. For algorithms which are merely consistent, however, we must also bound the bias. In general this is very difficult to do; we cannot estimate bias by simply drawing a few more samples. Bias leads to results that are not noisy, but are nevertheless incorrect. In graphics algorithms, this error is often noticeable visually, in the form of discontinuities, excessive blurring, or objectionable surface shading.

Unbiased algorithms are often used to generate reference images, against which other rendering algorithms can be compared. Because unbiased methods make strong guarantees about the kinds of errors that can occur, they are useful for detecting and measuring the artifacts introduced by approximations.[3] For scenes of realistic complexity, unbiased

[3] Improvements in unbiased algorithms may also lead to better approximation techniques. (Similarly, [Arvo 1995] has pointed out that better analytic methods can lead to better Monte Carlo methods.) Our viewpoint is that one should start with an unbiased algorithm, and adopt approximations only where they are clearly necessary (and their effects are well-understood).

algorithms are the only practical way to generate images that we can confidently say are correct.

Other things being equal, it is clear that we should prefer an unbiased algorithm. The conventional wisdom in graphics is that unbiased methods are "too expensive", and that an acceptable image can be achieved in less time by making approximations. However, there is little research to support this claim. While there has been a great deal of work on light transport algorithms in graphics, very little of this has been directed toward unbiased algorithms. In our view, considerably more research is necessary before we can judge their capabilities. One of the goals of this dissertation is to explore what can and cannot be achieved by unbiased methods, to help resolve these questions.

1.5 Models of light and their implications for graphics

Light transport can be studied at many levels of abstraction, ranging from two-dimensional "flatland radiosity" to quantum simulations. It is useful to have a variety of these mathematical models at hand, so that we can select the simplest model that is adequate for each task. As we will see, some optical phenomena have profound implications for algorithm design, while others can be added quite easily. It is this choice about which effects to simulate that distinguishes different classes of rendering algorithms, and that separates light transport in graphics from similar problems in other fields.

In the following sections, we summarize the important optical effects that occur in the real world, and discuss their implications for light transport algorithms. Optical phenomena are grouped according to the least-complicated optical theory that can explain them (geometric, wave, or quantum optics). Each of these theories explains different aspects of the observed behavior of light.

1.5.1 Geometric optics

Geometric optics is essentially the particle theory of light. This model can describe a wide range of optical phenomena, including emission, diffuse and specular reflection, refraction, and absorption. This covers most of what we see in everyday environments, which is why so many rendering algorithms are based on geometric optics.

However, full geometric optics is too complex for most rendering applications. In computer graphics we usually make more restrictive assumptions, to obtain simpler and faster light transport algorithms.

For example, *participating media* are often ignored. In general, light can be emitted, scattered, or absorbed in a three-dimensional medium, such as fog or gelatin. By ignoring these possibilities, all scattering is assumed to happen at surfaces (which are infinitely thin). This also implies that no energy is lost as light travels between surfaces.

In principle, it is easy to include participating media in Monte Carlo algorithms, by simply extending the ray casting procedure to sample the volume scattering and absorption along the ray [Rushmeier 1988]. The main effort required is the implementation of additional geometric primitives. Considerably more work is necessary to implement participating media with finite element approaches, since three-dimensional volumes must be meshed and subdivided, and the interaction with two-dimensional elements must be properly accounted for [Rushmeier & Torrance 1987]. With either approach, it is easier to handle media that only absorb light (no emission or scattering), since this can be handled in the same way as surface occlusion (these media block a fraction of the light traveling on a given ray, rather than all or none).

Geometric optics also allows media that have a *continuously varying refractive index*. This situation occurs when air is heated, for example, leading to shimmering "mirage" effects. In theory, this effect makes the light transport problem much more complicated, since beams of light no longer travel in straight lines between surfaces. Instead, they follow curved trajectories described by the *eiconal equation* [Born & Wolf 1986], which must be integrated in small steps to determine the path of a beam. To check for "visibility" between two points (i.e. the existence of an optical path that connects them), we must solve a difficult optimization problem. Some of these problems can be alleviated by making approximations [Stam & Languenou 1996]. However, since this effect is not important for most graphics models, it is usually just ignored.

Another common assumption is that light is *monochromatic* (i.e. that it has a single frequency). This is usually just a convenience, to simplify the description of algorithms. It is usually straightforward to deal with polychromatic light, by calculating with full spectra rather than monochromatic intensities. Operations on spectra are usually handled through a generic interface, so that different spectral representations can be substituted easily. A large number of representations have been proposed, with various tradeoffs between accuracy and expense [Hall 1989, Peercy 1993]. Sometimes, it is argued that polychromatic light can be handled by simply repeating a monochromatic algorithm at different wavelengths. However, this is rarely a good idea. Many calculations must be repeated separately at each wavelength, and any variations between the results at different wavelengths (for example, the mesh resolution or the location of random samples) can lead to objectionable color artifacts.

Similarly, *transmission* through surfaces is often disallowed. Again, this is usually just a convenience in describing algorithms. Transmission can be handled just like reflection, except that light is scattered to the opposite side of the surface. However, some care must be taken when the refractive index changes from one side to the other, since the radiance of a light beam changes according to the square of refractive index (see Chapter 5). Also, the index of refraction may depend on the frequency of the incident light, leading to the familiar rainbow effect known as *dispersion*.

For some algorithms, *ideal specular scattering* is not supported. This includes reflection by mirrors, and refraction between water and air. This is mainly a problem for algorithms that require an explicit representation of the scattering properties of a surface (e.g. as a polynomial function). In these representations, mirror-like surfaces correspond to Dirac delta distributions, which are not easily handled. If specular surfaces are supported by these algorithms at all, it is often only large, flat mirrors, which can be handled by reflecting the environment around the plane of the mirror, and treating the mirror as a window [Rushmeier 1986, Wallace et al. 1987, Rushmeier & Torrance 1990]. It is relatively easy to support specular surfaces in Monte Carlo algorithms, although this may add considerable variance to the calculations (see Chapter 8).

Finally, some algorithms support only *ideal diffuse* reflection (or transmission). A diffuse surface appears equally bright from all viewing directions; the direction in which a photon is scattered does not depend on how it arrived. This is a serious limitation, since real scenes contain a wide variety of materials, and it is often the variation in their scattering properties that makes an image look interesting or real.

The main advantage of diffuse surfaces is that their appearance depends only on position, rather than position and direction. This reduces a four-dimensional problem to a two-dimensional one, which can obviously lead to simpler algorithms. However, it is usually quite difficult to convert an algorithm the other way, from diffuse surfaces to general materials. Some algorithms handle surfaces that are a linear combination of ideal diffuse and ideal specular, but this is not at all the same as supporting general scattering functions. There are also some algorithms which appear to be general, but where in fact only diffuse surfaces are handled efficiently (e.g. other materials are handled via distribution ray tracing). Claims of generality for these algorithms are misleading, since they do not perform well unless most surfaces are diffuse. For testing generality, it is perfectly reasonable to use a scene with *no* ideal diffuse materials, since these materials do not exist in the real world.

1.5.2 Wave optics

Light can also be regarded as an electromagnetic wave [Born & Wolf 1986]. This model explains all of the phenomena handled by geometric optics, plus a few more. It is not always necessary to simulate the wave model of light to obtain wave effects. For example, polarization can be added quite easily to rendering systems based on geometric optics. In fact, the models of light transport in graphics often combine features from all three optical theories.

One effect exhibited by waves is *diffraction*, which causes light to "bend" slightly around obstacles. While diffraction is rarely noticeable at human scales, it cannot be neglected for small objects (e.g. those which are less than ten wavelengths across). This is an important issue in predicting reflection from rough surfaces, for example by simulating light transport at the microgeometry level [He et al. 1991]. However, it is difficult to incorporate diffraction into most light transport algorithms, since it violates the assumption that light travels in straight lines.

Another important wave effect is *coherence*. Coherence is a relationship between two beams of light, which measures the average correlation between their phases [Born & Wolf 1986]. So far, we have been assuming that light waves are perfectly incoherent, meaning that any two such waves have no phase correlation. The most important property of incoherent beams is that when they are superposed, their intensities add linearly (where intensity means the mean squared amplitude). This agrees with our usual intuition, e.g. two 100 watt bulbs are twice as bright as one 100 watt bulb.

When two beams are partially or fully coherent, their superposition results in *interference*. If there is positive correlation between their phases, it is called constructive interference, otherwise it is destructive interference. When two coherent beams of equal intensity are combined, the resulting intensity can be anywhere from zero to four times as great.[4] This effect is responsible for the light and dark bands in the classic "two-slit experiment" [Born & Wolf 1986].

Interference is important when modeling very small features, such as thin coatings or soap bubbles [Gondek et al. 1994]. Light is reflected back and forth inside the coating, so that the incident light wave is superposed on itself. This leads to interference, since any light beam is perfectly coherent with itself, and there is still partial coherence between two points on the beam that are several wavelengths apart. This applies even to beams from "incoherent" sources, such as incandescent light bulbs.[5]

Interference can be included in light transport algorithms by keeping track of the phases of all light beams [Gondek et al. 1994]. This requires keeping track of the optical length of the path traveled by each beam from the same source, including any coherent reflections or refractions. However, for most applications this additional expense is not justified.

Coherence is also related to *polarization*. Light is a transverse electromagnetic wave, which can be represented as point moving in a two-dimensional plane (this point is the tip of the electric vector, which is always contained in the plane perpendicular to the direction of propagation). Equivalently, we can regard light as the superposition of two independent waves, vibrating at right angles to each other. (Project the function onto two perpendicular vectors, such as the x- and y-axes.) Just as with any waves, these two waves can be partially or fully coherent, or have different amplitudes. If any of these things are true, we say the light is polarized.

[4] This is not an example of non-linear optics (discussed in the next section), since the waves themselves add linearly. However, a wave with double the amplitude corresponds to a fourfold increase in intensity.

[5] Note that the assumption of perfect incoherence in Section 1.1.2 is simply a mathematical abstraction; perfectly incoherent light does not exist in the real world.

Polarization is important for modeling materials such as water or glass, where the scattering properties depend strongly on how the incident light is polarized. Another effect that depends on polarization is *birefringence* (also known as *double refraction*) [Drude 1900]. It occurs in certain kinds of crystals, where the refractive index is different for light polarized parallel or perpendicular to the crystal surface. This has the effect of splitting an incident beam of light into two beams with opposite polarizations, which are refracted in different directions.

Polarization is quite easy to include in most light transport algorithms; the effort is similar to that of adopting a different spectral representation. There are two common representations of polarization: the Jones matrix (appropriate for monochromatic light, which is always completely polarized), and the Stokes matrix (which applies to partially polarized, perfectly incoherent light beams). The general problem of superimposing two partially coherent, partially polarized beams is more difficult; there are no simple representations in general, other than working with an explicit description of the waveforms [Perina 1985].

1.5.3 Quantum optics

Quantum physics offers the most detailed, accurate model of the behavior of light.[6] Some of these effects are not explained by the geometric or wave theories, but are still relevant to computer graphics.

One of these effects is *fluorescence*. This occurs when photons are absorbed by a molecule, and then a new photon is emitted at a different wavelength. This effect is actually quite common in the real world. For example, fluorescent dyes are used commercially to obtain brighter colors; this is why clothing often "glows in the dark" under ultraviolet lights.

Fluorescence is quite easy to add to rendering systems [Glassner 1994], by allowing energy at different wavelengths to interact (in a linear way). If light spectra are represented as vectors (with one coefficient per wavelength), then scattering from a surface can be represented as a matrix. When there is no fluorescence, this matrix is diagonal; otherwise, some

[6]Feynman [1985] has written a very readable account of the basics of this theory, and makes fascinating connections between the macroscopic and quantum behaviors of light.

of the off-diagonal entries will be nonzero.[7]

Another interesting effect is *phosphorescence* [Glassner 1994]. Here photons are absorbed, and re-emitted at a later time (usually at a different wavelength). This effect is not important for most computer graphics applications; however, there are similar problems in other fields where this kind of time-delay reaction is crucial (e.g. the decay of radioactive elements). The implementation of phosphorescence requires that the rendering algorithm integrate the incident light over time, since the current emission of a phosphorescent surface depends on its exposure in the past.

All of the effects we have described so far belong to *linear optics*. Consider an arbitrary optical system, which takes one light beam as input, and produces another light beam as output. The optical system is *linear* if the output wave is a linear function of the input wave; e.g. if we superpose two input waves, the output must be the sum of the outputs we would get if each wave were used alone. This property holds for practically every optical system.

However, with the introduction of lasers, non-linear effects have been discovered. For example, when high-intensity laser light passes through certain crystals, the light that exits the crystal is twice the frequency of the light which enters it. This is known as *frequency doubling* [Bloembergen 1996]. It does not happen with low-intensity light, so this is an example of non-linearity.[8]

There are many other effects whose explanation rests on quantum physics. For example, the photoelectric effect, or the observed spectral distribution of blackbody radiation. Lasers also depend on quantum physics for their explanation. However, these effects are irrelevant for computer graphics. We do not need to simulate blackbody radiation from first principles to include it in our scene models. Similarly, special and general relativity can be ignored for all practical purposes (e.g. the bending of light in a gravitational field).

[7] Glassner [1994] points out that for real materials, the matrix is often triangular. Photons often migrate from higher to lower energies during scattering, but rarely move in the other direction. This is why clothes do not "glow in the dark" when exposed to heat lamps.

[8] Consider a beam of light that is so intense that it heats the receiving surface, until it begins to glow. This effect is non-linear (since with a dim beam of light, the surface will not glow at all). However, this is not what is meant by non-linear optics. The surface temperature depends on the *integral* of the incident light over time (unlike the frequency doubling example). At each instant in time, the system is still linear, since the surface emission does not depend on the current intensity of the incident light.

1.6 Related problems from other fields

Light transport is similar to a variety of problems in physics and engineering. It is important to have a clear understanding of the connections between these problems, since many of the techniques used in graphics were first discovered in other areas. There is still much to learn from other scientific fields, and conversely these fields also have something to learn from computer graphics.

However, the underlying assumptions in other fields are often very different from those in graphics. This can make it difficult to transfer results from one field to another. In fact, some aspects of the light transport problem seem to be unique to computer graphics.

One important difference is the representation of the final output. In computer graphics, the final output always consists of images, and any other representations of the solution are just intermediate steps toward this goal. In physics and engineering, images are not important (except possibly as a visualization aid). Instead, the objective is to compute a set of numerical measurements, or even better, a functional representation of the solution over its entire domain. A full representation of the solution makes it easier to locate design problems (e.g. a leak through the shielding of a nuclear reactor).

Another difference is the way in which the quality of a solution is measured. In other fields, the goal is to compute results that are objectively accurate, according to standard numerical error metrics (e.g. the L_2 norm). In computer graphics, on the other hand, the ultimate error metrics are perceptual (and are thus not easy to define explicitly). Visual artifacts such as discontinuities or Mach bands are very objectionable in graphics, yet they are perfectly acceptable in heat transfer or nuclear engineering problems (as long as the numerical error is satisfactory). Because of this, popular methods in other fields are not always well-suited for graphics applications. In fact, perceptual error has been one of the main forces driving further research on light transport algorithms.

In the remainder of this section, we discuss the light transport problem as it relates to nuclear engineering, radiative heat transfer calculations, radar and acoustic wave scattering, and many-body problems.

1.6.1 Neutron transport

One of the first applications of Monte Carlo methods was the design of nuclear devices. Early Monte Carlo pioneers, such as von Neumann and Ulam, discovered techniques in this context that have now found much wider applicability [Ulam 1987]. Neutron transport problems are natural candidates for Monte Carlo methods, because of the relatively large number of dimensions involved (position, direction, energy, time), and the complexity of the interactions with atomic nuclei.

Light transport has much in common with neutron transport. They are governed by the same underlying equation (the *Boltzmann equation*), which describes the transport of virtually any kind of particles that do not interact with each other.[9] This equation is one of the central aspects of *transport theory*, which studies the transport of generic particles without regard for their physical meaning [Duderstadt & Martin 1979].

However, neutron and light transport differ substantially in emphasis. For example, the simulation of participating media is not important for most applications in computer graphics, whereas it is absolutely essential for neutron transport. Neutrons penetrate much farther into solid objects than photons, so that volume scattering (and volume emission) are the dominant effects. In fact, surface scattering and emission are often completely ignored in these simulations [Spanier & Gelbard 1969].

Another important difference is the interaction between particles at different energy levels. In graphics, fluorescence and phosphorescence are relatively insignificant effects. This means that to a good approximation, photon scattering is *elastic* (its wavelength does not change) and *instantaneous* (there is no significant delay between the arrival and departure of the photon). On the other hand, the scattering of neutrons is inelastic: they generally gain or lose some energy upon collision with a nucleus (an effect similar to fluorescence). Likewise, there is a small delay between the arrival of a neutron, and the scattering or emission of other neutrons (similar to phosphorescence). These delays can substantially affect the outcome of the calculation, and cannot be ignored.

[9] The Boltzmann equation does not model particle transport perfectly, since it is based on assumptions similar to those of geometric optics. For example, it ignores wave effects such as diffraction.

A third major difference is the existence of conservation principles. In graphics, we often rely on conservation of energy: the light scattered from a surface is no greater than the light incident upon it. With neutrons, on the other hand, it is often the objective to avoid this type of conservation. It is possible for a nuclear reaction to be critical or supercritical, in which case the number of neutrons in the environment increases quickly with time. In terms of individual collision events, a single incident neutron may cause several new neutrons to be emitted (by splitting the atomic nucleus). Other kinds of particles may be emitted as well, such as high-energy photons (gamma rays), and it is often necessary to track these particles as well.

Despite these differences, many techniques from the neutron transport literature can be adapted to computer graphics. This is usually quite easy, since light transport is a simpler problem.

There is also some interest in transport algorithms for charged particles, such as electrons. However, an important property of charged particles is that they interact with each other at a distance, by means of the electromagnetic field. Similarly, the path of a charged particle is influenced by fixed electric and magnetic fields, so that these particles follow curved trajectories (similar to photons passing through a medium with a continuously varying refractive index). These features give the transport of charged particles a considerably different flavor, and most light transport algorithms cannot easily be adapted to this purpose.

1.6.2 Heat transfer

Radiative heat transfer is also very similar to light transport. In fact, the only difference is that the photons in heat transfer have longer wavelengths (in the infrared portion of the spectrum). As with neutron transport, however, different aspects of the problem are emphasized.

First, we review the three mechanisms of heat transfer: conduction, convection, and radiation. With *conduction*, energy is exchanged between adjacent vibrating atoms, as they bump into each other. This causes a slow migration of heat away from "hot spots" (e.g. this is what causes the handle of a frying pan to become hot). With *convection*, heat is transferred by the large-scale movements of atoms (e.g. a draft of hot air). Finally, heat can be transferred by the *radiation* of photons, which carry energy almost instantaneously across large distances (e.g. the heat that is felt when standing near a campfire). It is this last mechanism that is similar to light transport.

This brings us to the first important difference from light transport, namely that radiation is only one aspect of heat transfer problem. For many applications, conduction and convection are at least as important. (One indication of this is that the *heat equation* in the applied mathematics and engineering literature often refers only to conduction [Gustafson 1987, Hughes 1987].) In theory, conduction and convection can also affect light transport calculations, if portions of the surrounding environment are so hot that they begin to glow (i.e. emit photons in the visible wavelengths). However, this definitely falls outside the traditional realm of computer graphics.

A second difference is that heat transfer problems are often non-linear. For example, the spectrum of radiation emitted by a hot surface depends on the fourth power of its temperature, and convection is also affected by temperature in complex ways. However, these non-linearities are irrelevant for our purposes, because the *radiative* aspect of heat transfer is always a linear problem. Temperature changes due to conduction, convection, and even radiation are extremely slow compared to the speed of light, so that the system is effectively in radiative equilibrium at all times.

Unlike neutron transport, most heat transfer algorithms are based on the finite element method.[10] There are several reasons for this. First, finite element methods compute a representation of the entire solution (rather than isolated measurements), which makes it easier to locate design problems. Second, a full solution also makes it easier to include the effects of conduction and convection, and to follow the evolution of the system over time. Finally, finite element methods are a standard tool in civil and mechanical engineering, so that it was natural to extend these methods to heat transfer problems.

The heat transfer literature has thus inspired finite element approaches to light transport, just as neutron transport algorithms have inspired Monte Carlo work.

[10]Technically, these are often *boundary element methods* [Siegel & Howell 1992], where the solution is represented only on the boundary of the domain rather than its interior. This is the preferred representation in the absence of convection.

1.6.3 Radar and acoustics problems

The scattering of radio waves is another problem that is similar to light transport. Radio waves are simply another part of the electromagnetic spectrum, but with much longer wavelengths than visible light or radiant heat. Consequently, the wave nature of electromagnetic radiation becomes important: effects such as diffraction and interference cannot be ignored. For this reason, the mathematical models and algorithms for these problems are based on the wave model of light, rather than geometric optics. This yields a totally different set of algorithms and insights.

Radio scattering problems arise in the design of objects that are difficult for radar systems to detect (e.g. military aircraft). Similar problems arise in the design of auditoriums and concert halls, where it is important to predict the scattering of sound waves. The wavelengths of audible sounds are comparable to the dimensions of ordinary objects (ranging approximately from one centimeter to ten meters), so that wave effects cannot be neglected.

At their most basic level, these problems involve solving the *wave equation*, a partial differential equation that describes how waves propagate with time [Strang 1986, Gustafson 1987, Zauderer 1989]. This formulation is extremely general, but for realistic problems it is also difficult and expensive to solve. The problem can be greatly simplified by assuming that all radio sources have a single frequency, and that their intensity does not change with time. This is called the *time-harmonic* version of the problem. Such a system will rapidly converge to an equilibrium state, where the intensity of the electromagnetic field at each point is a sinusoidal function of time. The amplitude and phase of the electromagnetic vibration at each point can be represented by a complex number.

Mathematically, the reduced problem is described by the *Helmholtz equation*, also known as the *reduced wave equation* [Zauderer 1989]. This is a partial differential equation, like the wave equation, except that there is no time dependence (since we are solving for an equilibrium state). Formally, this means that the Helmholtz equation is an elliptic problem, rather than a hyperbolic problem like the wave equation. Elliptic problems require an entirely different set of solution techniques than hyperbolic ones, and are generally easier to solve.

Methods for the scattering of radio and sound waves can be applied directly to the light transport problem; the restriction to a single frequency means that only monochromatic light can be handled (or that each frequency must be simulated independently). This formulation correctly handles diffraction and interference, as well as all of the phenomena handled by geometric optics. This could lead to interesting solution techniques for graphics problems where the wave nature of light is important.

1.6.4 Many-body problems

Efficient algorithms for many-body problems are an important recent influence on computer graphics. The simplest version of this problem involves a set of N particles, each with a different mass. The problem is to determine the gravitational force exerted on each particle by the others. This can be used to simulate the motion of the particles, by integrating their velocity and position over time. The problem can be extended to charged particles, and also to bodies with more complex shapes.

The obvious algorithm for this problem is to compute the $O(N^2)$ pairs of forces, and add them together to find the net force acting on each particle. However, recently several algorithms have been proposed that are far more efficient. These algorithms have complexities of $O(N \log N)$ [Barnes & Hut 1986] or even $O(N)$ [Greengard & Rokhlin 1987, Greengard 1988]. The basic idea is that distant particles can be grouped together, replacing the calculations for many individual particles with a single computation for the group. Because of the $O(1/r^2)$ falloff of gravitational and electric force, these approximations are possible without significant loss of accuracy. Particles are organized into a hierarchical data structure, so that nearby particles can be processed in small groups, while distant particles are handled in large groups.

These techniques were the inspiration for hierarchical light transport algorithms [Hanrahan et al. 1991]. It is easy to see that there is some connection; for example, the intensity of a point light source obeys the same kind of $O(1/r^2)$ falloff law as gravity does. In fact, if we simply replace point masses by point light sources, many-body algorithms can be used to efficiently compute the *fluence rate* due to these light sources at many points simultaneously. (The fluence rate at a point in space is the integral of the incident radiance over all directions, i.e. the total power per unit cross-sectional area that would be received by a tiny spherical light sensor [American National Standards Institute 1986].)

However, computing the fluence at isolated points is not particularly useful for making images. There are substantial differences between light transport and the many-body problem, such as occlusion. Gravity passes through walls, while light does not. Furthermore, the gravitational force is a function only of position, while light intensity (radiance) is a function of position and direction. (This is because a point mass creates the same gravitational force in all directions, while a point light source can radiate different amounts of light in different directions.)

These differences make light transport considerably more complex than the many-body problem, and help to explain why hierarchical algorithms in graphics have not been able to make the same accuracy and performance guarantees that are available for many-body algorithms. The results for many-body algorithms are quite impressive: solutions can be computed with any accuracies comparable to the machine's floating-point resolution, with a time complexity of $O(N)$ [Greengard 1988].[11] It is doubtful that similar results will ever be obtained for realistic light transport problems.

[11]Note that although the force-calculation component of the Greengard algorithm is $O(N)$, there is also a tree building component that can take $O(N \log N)$ time. Similarly, the complexity of the Barnes & Hut [1986] algorithm can be significantly worse than $O(N \log N)$ when the particle distribution is non-uniform [Anderson 1996].