12-2016

Radiative Transfer Using Path Integrals for Multiple Scattering in Participating Media

Paul Michael Kilgo
Clemson University, paulkilgo@gmail.com

Radiative Transfer Using Path Integrals for Multiple Scattering in Participating Media

---

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Computer Science

---

by
Paul M. Kilgo
December 2016

Accepted by:
Dr. Jerry Tessendorf, Committee Chair
Dr. Joshua Levine
Dr. Brian Dean
Dr. Amy Apon

Abstract

The theory of light transport forms the basis by which many computer graphic renderers are implemented. The more general theory of radiative transfer has applications in the wider scientific community, including ocean and atmospheric science, medicine, and even geophysics. Accurately capturing multiple scattering physics of light transport is an issue of great concern. Multiple scattering is responsible for indirect lighting, which is desired for images where high realism is the goal. Additionally, multiple scattering is quite important for scientific applications as it is a routine phenomenon. Computationally, it is a difficult process to model. Many have developed solutions for hard surface scenes where it is assumed that light travels in straight paths, for example, scenes without participating media. However, multiple scattering for participating media is still an open question, especially in developing robust and general techniques for particularly difficult scenes.

Radiative transfer can be expressed mathematically as a Feynman path integral (FPI), and we give background on how the transport kernel of the volume rendering equation can be written in terms of a FPI. To move this model into a numerical setting, we need numerical methods to solve the model. We start by focusing on the spatial and angular integrals of the volume rendering equation, and show a way to generate seed paths without regard as to if they are cast from the emitter or the sensor. Seed paths are converted into a discretized form, and we use an existing numerical method to tackle the FPI. A modified version of this technique shows how to reduce the running time from a quadratic to a linear expression. We then perform experimental analysis of the path integral calculation. The entire numerical method is put to full scale test on a distributed computing platform to

calculate beam spread functions and compare the results to experimental data.

The dissertation is laid out as follows. In Chapter 1, we introduce the basic concepts of light propagation for computer graphics, multiple scattering, and volume rendering. Chapter 2 offers background on the subject of FPIs and some mathematical techniques used in their numerical integration for this work. Chapter 3 is a survey of radiative transfer and multiple scattering as it is studied in computer graphics and elsewhere. Chapter 4 is a full description of the current methodology. In Section 4.1 we describe sensor and emitter geometries used for our experiments. We propose a new algorithm for creating seed paths to use in the numerical integration of the FPI in Section 4.2. Section 4.3 introduces past work in the numerical integration, formalizes it, and improves upon its running time. Section 4.4 presents some analysis of the path weighting. In Chapters 5 and 6 we run experiments using the numerical methods. The first characterizes the calculation of the path integral itself using arbitrary spatial parameters, and shows repeatability and unbiased calculation given enough samples. In the second, we calculate beam spread functions, a basic property of scattering media, and compare the calculations to experimentally acquired data. Chapter 7 presents a summary of contributions, a summary of conclusions, and future directions for the research.

Acknowledgments

No number of words on paper can adequately convey the gratitude I have toward those who have guided and supported me through my graduate education. I have nothing but the utmost respect for my colleagues, friends, and family who have made my graduate education a possibility.

First I would like to thank my advisor, Dr. Jerry Tessendorf, who helped turn my general interest in computer graphics into a clear research direction and career path. Despite his many obligations he always made the time to help me through the latest obstacle, however small it may have been to him. His teaching, relaxed honesty, and confidence in me were crucial to completing this work.

My dissertation committee has helped me along the way in many facets. Dr. Joshua Levine frequently made time to discuss both current and future research directions with me. His penetrating assessment of my work has always encouraged me to be at my best, and his suggestions have often led to new ideas to try. I also acknowledge Dr. Brian Dean for the many helpful suggestions and for organizing the algorithms seminar which I frequented. I thank Dr. Amy Apon for her personal interest in my work and future.

Many others have suffered through early renditions of my talks, papers, and ideas. Matt Dabney was frequently a victim of proximity in this regard, so I would like to thank him for putting up with it. I would like to thank the faculty and students of the Visual Computing lunch seminar for pushing me to present my research ideas frequently and early, and being able to lend a critical eye to all my major talks.

Nothing would have been possible without someone paying my bills, and for that

I am grateful to Dr. Mark Smotherman and the School of Computing for hiring me as a teaching assistant and later a lab assistant for my entire time at Clemson University. I would like to thank Scott Duckworth for taking time out of his day to teach me all the things I did not know about systems administration and software development, an experience which frequently rivalled many of my graduate courses in difficulty. I worked with many other wonderful people along the way–Matt Dabney, Dorothy Puma, and Chuck Cook to name a few.

I would like to thank Dr. Nicholas A. Kraft for encouraging me to pursue a PhD at Clemson, and for his continuing support in my future career endeavors. I truly would have not considered continuing my education otherwise. I would also like to thank Dr. Julie P. Martin whose career guidance was a great personal help.

The Open Science Grid (OSG), which is supported by the National Science Foundation and the U.S. Department of Energy's Office of Science, provided many centuries of compute time for this research project alone. I thank the Center for High Throughput Computing and the University of Wisconsin-Madison for hosting and financing my trip to the OSG User School 2015. Also, I would like to thank the Cyberinfrastructure Technology Integration group at Clemson University for use of the Palmetto supercomputer. The resources and learning they provided were instrumental in achieving the results of this dissertation. I would also like to thank the Clemson University Graduate Student Government and the M&C 2015 planning committee for providing travel grants to present this research. Parts of this work have appeared in other publications (see [34, 35]).

My family has always been supportive in my academic career–financially, emotionally, and intellectually. I would like to thank my father and mother for putting me through college and graduate school and planning early on to do so. I would like to thank my father for reading a lot of my early drafts of papers, providing me with books, and asking questions I sometimes did not know how to answer. Finally, my deepest thanks goes to my fiancée (now wife!) Kayla who has put up with me through the most manic of times. I dedicate this dissertation to her.

To Kayla

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The propagation of visible light and other near-visible electromagnetic radiation has been studied for several centuries. For computer graphics purposes, the predominant theory for describing light propagation is known as radiative transfer. It dictates that radiant energy can be redistributed by three fundamental mechanisms: absorption, where the electromagnetic energy is converted into other forms of energy; scattering, where the energy is redistributed into many different directions; and emission, where energy is gained from the optical properties of the material. We experience absorption and emission every day. A car in the sun absorbs its radiant energy. A candle emits light due to a chemical reaction. Scattering is an everyday phenomenon as well, though its effects are often subtle: the blueness of the sky, the appearance of clouds.

Another characteristic of scattering is that it is complicated to compute. When a scattering event occurs, it is not often as simple as a single beam being redirected in another direction. A beam may scatter in several different directions, its energy being redistributed in each child beam. If we allow arbitrary numbers of scattering events, we can readily see an exponential growth in the hierarchy of paths we must trace. Moreover, the distribution of the beams that scatter may not be simple. In computer graphics, light can interact in two primary ways with a scene: either at a surface or through a region of *participating media*. On a surface which does not allow transmitted rays, light can only scatter in a hemisphere

defined by the surface normal. In this case, the distribution of the scattering is known as the bidirectional reflectivity distribution function (BRDF).

With participating media, light propagates through a density of material with spatially varying optical properties. In this case, the material alters the course light takes–it participates in its transport. The macroscopic behavior of the light is affected by a series of scattering events causing it to not travel in straight paths. Each of these scattering events behaves similarly to surface scattering, except the scattering distribution function is defined over a sphere rather than a hemisphere. In this case, the distribution function is known as the *phase function*. Phase functions often are discussed by the kinds of scattering they support. Forward-peaked phase functions support more scattering in the forward direction and likewise backward-peaked phase function support more scattering in the backward direction, also known as backscattering.

The phase function and the *scattering coefficient* define the scattering behavior of participating media. The scattering coefficient has units of inverse length and measures the likelihood the media will scatter the light per unit length. A higher scattering coefficient implies a higher likelihood of scattering. We write the scattering coefficient as b. An important distance measure is the *scattering length* which is the inverse of the scattering coefficient, $1/b$. For a distance L it is often useful for establishing relative behavior to express that distance in number of scattering lengths, written Lb. A similar quantity to the scattering coefficient exists called the *absorption coefficient*, written a, which describes the likelihood of the media to absorb the light per unit length. The two form the total extinction term, $c = a + b$.

Because of the complexity scattering offers, designers often limit the number of scattering events in their rendering algorithm – either as a controllable parameter, assuming a fixed number, or by a process which is Monte Carlo in nature. We call the special case where only one scattering event is allowed *single scattering*. Allowing more scattering events is called *multiple scattering*. Using a single scattering approach is often a useful approximation because generally single-scattered light beams contribute the most intensity to the

final image. However, multiply-scattered light is more physically correct, but either computationally costly or difficult to implement. Computing multiple scattering is a problem of great interest to the computer graphics community as it is a piece of the solution to global illumination.

Generally to render participating media, we use one of a class of techniques known as *volume rendering*. One of the most common volume rendering approaches is ray casting. Similar to ray tracing, rays are traced originating from the camera position. Along the ray, samples of the density and color functions are taken. At each of these locations, a scattering event redirects light from the source based on the phase function and the optical parameters. The color and transmission are accumulated along the ray to obtain the final pixel color. This is a single scatter approximation.

Within computer graphics, the multiple scattering problem is often addressed trying to solve global illumination. One of the earliest solutions was path tracing [27] and the later extension bidirectional path tracing [36, 37]. There was interest in improving the running time, giving rise to methods like photon mapping [25] and instant radiosity [32]. Often the physical accuracy of the model is abandoned in the interest in finding a solution faster. Physical accuracy is the main motivation of the path integral formulation of radiative transfer, and we hypothesize that it can be implemented numerically to agree with optical experiments.

The path integral formulation of radiative transfer was introduced in 1987 [58], intended as a way of describing propagation of light through participating media. It differs from traditional tracing techniques mainly in its mathematical basis in Feynman path integrals (FPI). The image is constructed from a sum over all possible paths that a given beam of light could have travelled from source to camera. To compute the sum, we sample over the entire space of possible paths, compute a weight for each path, and accumulate the path weights. There is no assumption regarding the number of scattering events, so there can be arbitrarily many. While the applications for the FPI approach in computer graphics are limited to rendering, new computational techniques for radiative transfer which

are physically accurate can have broader application. These applications span a number of disciplines including ocean and atmospheric science, medicine, nuclear engineering, and geophysics.

In previous work the FPI approach to radiative transfer had been developed mathematically [58, 59], its behavior in a numerical setting studied [60], and some approximations for it explored [61]. This dissertation focuses on using FPI approach in a computational setting. This includes establishing some of the first numerical techniques used for Monte Carlo evaluation of the FPI and surrounding spatial integrals (Chapter 4), evaluating the statistical behavior of the FPI quantity itself (Chapter 5), and replicating a result of an optics experiment (Chapter 6).

Chapter 2

Background

This chapter presents a brief overview of some of the concepts used extensively throughout the remaining parts of the dissertation. In Section 2.1 we briefly overview Feynman path integrals and their use in a numerical setting. Section 2.2 overviews the basic properties of the Frenet-Serret frame, and details a discretized space curve specification which is very integral to the rest of the dissertation. We introduce Monte Carlo integration in Section 2.3 including its basics, and some discussion about how it is used to evaluate a path integral. Finally, Section 2.4 describes the mathematics behind the FPI approach to radiative transfer.

2.1 Feynman path integrals

The Feynman path integral was introduced [10] as an unconventional alternative to compute the probability amplitude of a quantum mechanical particle instead of solving a wave equation. Roughly speaking, the probability amplitude is related to the probability density function of the states a quantum particle (like a photon) can assume. The path integral formulates this probability amplitude as a sum of the contributions of all paths a quantum mechanical particle could have taken. Feynman and Hibbs offer an introduction to path integrals in [11].

Path integrals have wide application and are useful for solving problems in physics, engineering, and chemistry. In a numerical setting, the use of Monte Carlo methods with path integrals is popular and has general application in physics. For instance, Miller and Clary [43] use a path integral Monte Carlo (PIMC) method to calculate internal energies for four carbon chain molecules. Herrero and Ramírez [17] use a PIMC method to study the structural and thermodynamic properties of diamonds, which they found agreement in some of the thermodynamic properties between the PIMC and experimental results. PIMC methods involve selecting random variates over a distribution of paths. Similar path-based techniques may generate random paths from previous paths through a kind of perturbation or mutation process [63] as a way of saving computer time. Often, this is not enough to achieve convergence in a reasonable amount of time, so many involve the use of the Metropolis algorithm [42].

Using the Metropolis algorithm phrases the problem of path generation as a Markov process where the next path drawn from the distribution is a function of the previously accepted path. In many cases for paths, this is a less expensive operation compared to selecting a totally random path. Kalos and Whitlock wrote a book [28] which overviews Monte Carlo methods from a more general perspective, though generally with applications to simulating physical systems. Khandekar et al. [33] elaborate on Monte Carlo methods and the Metropolis algorithm as it relates to path integrals.

2.2 Frenet-Serret curves

The Frenet-Serret frame is a definition of an orthonormal frame for continuous space curves, and it is usually an introductory topic in differential geometry textbooks [57]. The Frenet-Serret properties are defined for non-degenerate, arc length parameterized space curves $\mathbf{r}(s)$. From this, a Frenet-Serret frame of orthonormal vectors can be constructed:

$$\mathbf{T} = d\mathbf{r}/ds \tag{2.1}$$

$$\mathbf{N} = (d\mathbf{T}/ds)/\left\|d\mathbf{T}/ds\right\| \tag{2.2}$$

$$\mathbf{B} = \mathbf{T} \times \mathbf{N}. \tag{2.3}$$

$\mathbf{T}$ is the *tangent* vector, $\mathbf{N}$ is the *principal normal* (or less formally the *normal*), and $\mathbf{B}$ is the *binormal* vector. It can be shown that these three orthonormal vectors have the relation over all s

$$\begin{bmatrix} \mathbf{T}' \\ \mathbf{N}' \\ \mathbf{B}' \end{bmatrix} = \begin{bmatrix} 0 & \kappa & 0 \\ -\kappa & 0 & \tau \\ 0 & -\tau & 0 \end{bmatrix} \begin{bmatrix} \mathbf{T} \\ \mathbf{N} \\ \mathbf{B} \end{bmatrix} \tag{2.4}$$

with κ and τ being the curvature and torsion of the curve. We use prime notation to signify a derivative.

We work primarily with a discrete space curve for which these properties hold, which we refer to as a Frenet-Serret curve in this dissertation. The space curve is explained in depth in [60], reproduced here. In the discrete form, we are given a step size Δs, starting position $\vec{x}_0$ and a starting orthonormal basis $F_0 = [\mathbf{T}_0, \mathbf{N}_0, \mathbf{B}_0]$. We can then define a recurrence equation for the position sequence of the curve, $\vec{x}_i$, which follows directly from the step-wise integration of the space curve:

$$\vec{x}_i = \mathbf{T}_{i-1}\Delta s + \vec{x}_{i-1}. \tag{2.5}$$

Defining the sequence of orthonormal frames of the space curve, F_i, is a little more involved. We again define a recurrence equation

$$F_i = U_i F_{i-1} \tag{2.6}$$

and define U_i as a partial rotation matrix between F_{i-1} and F_i, based on the local curvature and torsion at that point:

$$U_i = \begin{bmatrix} 1 - \frac{\kappa_i^2}{\ell^2}(1 - \cos(\ell \Delta s)) & \frac{\kappa_i}{\ell}\sin(\ell \Delta s) & \frac{\kappa_i \tau_i}{\ell^2}(1 - \cos(\ell \Delta s)) \\ -\frac{\kappa_i}{\ell}\sin(\ell \Delta s) & \cos(\ell \Delta s) & \frac{\tau_i}{\ell}\sin(\ell \Delta s) \\ \frac{\kappa_i \tau_i}{\ell^2}(1 - \cos(\ell \Delta s)) & -\frac{\tau_i}{\ell}\sin(\ell \Delta s) & 1 - \frac{\tau_i^2}{\ell^2}(1 - \cos(\ell \Delta s)) \end{bmatrix} \tag{2.7}$$

with $\ell = \sqrt{\kappa_i^2 + \tau_i^2}$. Throughout the dissertation we will assume a curve has M segments.

The usefulness of the discrete Frenet-Serret formulation comes into play during the Monte Carlo integration over path configurations (Section 4.3). Our formulation gives us very good control over x_0, $\mathbf{T}_0$, M, and Δs since each are given. This is important for numerical reasons when we discuss the perturbation algorithm, as then it only has to preserve $\vec{x}_M$ and $\mathbf{T}_M$, the end point and end tangent.

2.3 Monte Carlo integration

The types of numerical integration of definite integrals normally discussed in a standard calculus course are informally referred to as quadrature methods. It includes such methods as the trapezoidal rule or Romberg's method. This type of integration typically involves sampling the target function at either fixed or dynamic intervals, computing the area of the small shape formed by these bounds. A simple visualization is provided by Figure 2.1.

A drawback to quadrature methods is the time complexity in higher dimensions, or the curse of dimensionality. Accurate quadrature methods sample the function at least at all boundaries of the integration bounds. For a single area of interest, this means a quadrature method would sample a one-dimensional integral twice, a two-dimensional integral four times, and so on. To formally understand the simple case where we always sample at the boundaries of the quadrature shape, suppose we estimate the value of an M-dimensional

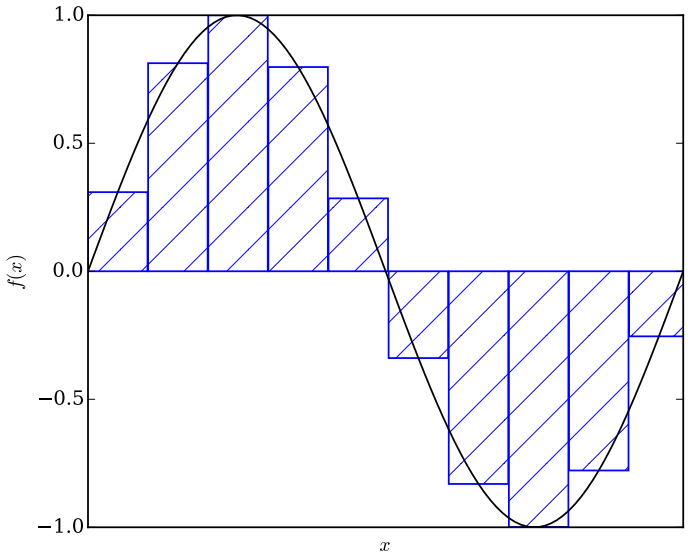


Figure 2.1: A quadrature of the function $f(x) = sin(x)$. In this example, the value of the function at the quadrature shape's midpoint is taken as the sampled value of the function. The value of the integral is estimated by summing the area of the quadrature shapes.

integral by forming N quadrature shapes. Suppose we use hypercubes as our quadrature shape. Then, for each of the N hypercubes, we must sample the target function $O(2^M)$ times. The time complexity required is then $O(N2^M)$. We can disregard information and take a fixed number of random samples around each region, but this is a trade-off between time and error. Typically, integral problems of very high dimensionality are not well suited to quadrature methods, and the Monte Carlo solutions outperform them [33, Section 11.2]. For a very thorough look at the curse of dimensionality with quadrature methods, see [45, Chapter 1].

As we are integrating over paths, our integration bounds are related to the parameters which describe the path. In the case of Frenet-Serret curves, these are the curvatures and torsions at the segments of the path. We can then expect dimensionality of our problem to be inversely proportional to Δs, the length of a path segment. For accuracy, we should require that Δs is very small. Thus, integrating over paths is a problem of very high dimensionality.

Monte Carlo integration is the preferred strategy for integrating high dimensional functions. It is helpful to understand from a lower-dimensional standpoint and extend to higher dimensions later. Kalos and Whitlock [28, Section 2.6] give an introduction to the basic idea of Monte Carlo integration. We apply this to a specific case here. Let $x_1, x_2, ..., x_N$ be a sequence of independent and identically distributed random variables from a distribution $p(x)$. We can then form an estimator based on these samples and the target function $f(x)$,

$$G_N = \frac{1}{N} \sum_{1}^{N} f(x_i), \tag{2.8}$$

and the expected value of G_N is related to our target function and the distribution we choose our samples from,

$$E(G_N) = \int f(x)p(x)\,dx. \tag{2.9}$$

In the case where $x \in [a, b]$ and $p(x)$ is uniform over $[a, b]$, then we have $p(x) = 1/(b - a)$, and therefore

$$(b - a)E(G_N) = \int_a^b f(x)\,dx. \tag{2.10}$$

In the limit of $N \to \infty$, then the variance of G_N decreases and approximates the expected value. The Monte Carlo integration of $f(x)$ then has the form

$$\int_a^b f(x)\,dx = (b - a) \lim_{N\to\infty} \frac{1}{N} \sum_{1}^{N} f(x_i). \tag{2.11}$$

In the single-dimensional case, we can see clearly that this scheme is $O(N)$. Scaling up dimensionality only slightly affects the complexity. If we take the dimensionality to be M, then x_i is an M-dimensional vector and takes $O(M)$ time per sample to generate. The time complexity for Monte Carlo integration is then $O(MN)$, which is preferable to the $O(N2^M)$ of quadrature integration. To compute the integral, we need to show convergence in the

limit.

In the case of integrals over paths, N is the interesting number since the dimensionality of a path is given, but it is suspected that N needs to be quite large to show convergence in the limit. As well, the $p(x)$ term is of interest, as it is the distribution our random variates follow. However, the bulk of this research assumes $p(x)$ is uniform and only the relative behavior is analyzed. The summation term

$$\lim_{N\to\infty} \frac{1}{N} \sum_{i}^{N} f(x_i) \tag{2.12}$$

is of primary interest since the character of its convergence determines a practical value for N to experimentally calculate a weight for a given curve.

Extending the Monte Carlo scheme of integration to curves is straightforward. We can represent a Frenet-Serret curve as a parameter set including an initial position ($\vec{x}_0$), initial orthonormal frame (F_0), a $2M$-dimensional vector of curvatures and torsions (K), and a path length (s or $M\Delta s$). Let these parameters be represented by C, and let $C_1, C_2, ..., C_N$ be a distribution of these path samples from a distribution $p(C)$. Let $f(C)$ be a weight function over path parameter sets. We can construct an estimator as before,

$$Q_N = \frac{1}{N} \sum_{1}^{N} f(C_i) \tag{2.13}$$

$$E(Q_N) = \int f(C')p(C')\, dC', \tag{2.14}$$

and in a Monte Carlo integration scheme, assuming a uniform $p(C)$, this may look like

$$\int f(C')\, dC' = D \lim_{N\to\infty} \frac{1}{N} \sum_{i}^{N} f(C_i) \tag{2.15}$$

where D is a product of the distribution analogous to $(b-a)$, which for the scope of the dissertation we ignore.

2.4 Path integrals for radiative transfer

Using Feynman path integrals for radiative transfer was first introduced by Tessendorf in 1987 [58] and the numerical behavior was first studied in 2009 [60]. This report gives a derivation of the mathematical model of volume rendering which is derived from the radiative transfer equation,

$$L(\vec{x}_1, \hat{n}_1) = \int_0^\infty ds \int d^3\vec{x}_0 \int_{4\pi} d\Omega_0 G(s, \vec{x}_1, \hat{n}_1, \vec{x}_0, \hat{n}_0) S(\vec{x}_0, \hat{n}_0), \tag{2.16}$$

where G is the transport kernel and S is a source function. The parameters are s, an arbitrary time variable with units of length; $\vec{x}_0$ and $\vec{x}_1$, the start and end (source and sink) position; and $\hat{n}_0$ and $\hat{n}_1$, the start and end (source and sink) direction. We shall refer to these as the *boundary constraints* as they are fixed when calculating the path integral. For the purposes of the FPI approach, we will assume s is the arc length of a path of interest. This equation for the radiance at $\vec{x}_1$ in direction $\hat{n}_1$ tells us we must integrate G over all space, over all directions, and over all arc lengths to compute an answer.

Even if the boundary constraints are fixed, there are several possible paths that satisfy these constraints. Therefore, calculating G, the transport kernel, is still a matter to discuss. The FPI approach says the transport kernel has a representation in the form of a Feynman path integral. We can define a path as a space curve $\mathbf{x}(s')$, $0 \leq s' \leq s$, having unit tangent vectors

$$\hat{\beta}(s') = \frac{d\mathbf{x}(s')}{ds'}. \tag{2.17}$$

The path integral portion only will accept space curves satisfying the boundary constraints set by the outermost integrals in Equation (2.16). Therefore, the constraints for the space curve are

$$\hat{\beta}(0) = \hat{n}_0 \tag{2.18}$$

$$\hat{\beta}(s) = \hat{n}_1 \tag{2.19}$$

$$\vec{x}_1 - \vec{x}_0 = \int_0^s ds' \hat{\beta}(s'). \tag{2.20}$$

We also define scalar field functions to represent the optical properties of the media. They are $a(\vec{x})$, the absorption coefficient; $b(\vec{x})$, the scattering coefficient; and $c(\vec{x}) = a(\vec{x}) + b(\vec{x})$, the total extinction. Though, in this dissertation we consider the case where these are not spatially varying. We simplify the notation by writing a, b, and c respectively in this case. The path integral form of G [60] is

$$\begin{aligned} G(s, \vec{x}_0, \hat{n}_0, \vec{x}_1, \hat{n}_1) = & \int [d\Omega][d\mathbf{p}] \\ & \times \delta\left(\hat{\beta}(0) - \hat{n}_0\right) \delta\left(\hat{\beta}(s) - \hat{n}_1\right) \delta\left(\vec{x}_1 - \vec{x}_0 - \int_0^s ds'\, \hat{\beta}(s')\right) \\ & \times \exp\left(-\int_0^s ds'\, c(\mathbf{x}(s'))\right) \exp\left(i \int_0^s ds'\, \mathbf{p}(s') \cdot \frac{d\hat{\beta}(s')}{ds'}\right) \\ & \times \exp\left(\int_0^s ds'\, b(\mathbf{x}(s')) \tilde{Z}(|\mathbf{p}(s')|)\right). \end{aligned} \tag{2.21}$$

We adopt the notation $[d\Omega]$ and $[d\mathbf{p}]$ to represent an integration over all the $\hat{\beta}(s)$ and $\mathbf{p}(s)$ functions. The delta functions limit the integration to those which satisfy Equations (2.18) to (2.20). The term $\tilde{Z}$ is the phase function of the media, transformed into a Fourier-like form and $\mathbf{p}$ is also introduced from this transformation. This dissertation only uses a forward-peaked Gaussian phase function for $\tilde{Z}$. The first exponential term captures the total extinction behavior (written $c(\vec{x})$) in part due to absorption. The latter two exponential terms model scattering behavior as mediated by the phase function.

Chapter 3

Radiative transfer

In this chapter we review the different applications of radiative transfer in several disciplines. For computer graphics, we will focus on techniques which seek to compute multiple scattering effects, most of them being global illumination solutions. We will also overview some of the software packages which use Monte Carlo and radiative transfer for particle physics in nuclear engineering. We review some of the work done studying multiple scattering in both medical imaging and ocean and atmospheric science. A seminal textbook on radiative transfer is available by Chandrasekhar [6].

3.1 Computer graphics

Multiple scattering techniques in computer graphics are typically Monte Carlo methods. Often, they have in common that paths are randomly traced from the scene lights to find interactions with the scene elements. Where these interactions are found, some lighting information is computed which is used in the rendering step to simulate multiple scattering effects. Often these techniques intend to solve the problem of global illumination, and we focus on these methods in this section. A thorough survey on rendering techniques for participating media is available from Cerezo et al. [5].

One of the earlier forms of global illumination solutions is path tracing first in-

troduced by Kajiya in 1986 [27]. It is a Monte Carlo alternative to traditional sampling procedures of ray tracing. Typically in ray tracing, when an interaction with an active ray is found the illumination is computed at this point and the ray is terminated. Special extensions can account for transmitted or reflected rays, but generally this is done for a handful of materials. In a path tracing renderer, at each ray-object intersection a Monte Carlo procedure determines if the ray scatters. If it scatters, paths are chosen randomly out of a solid angle distribution called the bidirectional reflectivity distribution function (BRDF). This process continues until all rays terminate, or some other stopping criteria halts execution. Image noise is a commonly cited problem in path tracing methods, but it also results in many ray-object intersection tests. Lafortune and Willems improved upon path tracing with bidirectional path tracing [36], where the ray-object intersection points and illumination information are reused for other path samples. With bidirectional path tracing, a path is traced from light sources, and scattering events with scene objects are computed as before. These light paths are reused when tracing paths from the camera. For each scattering event along a camera path, the path can be "joined" with each leg of the light path to create ensembles of valid paths. The technique was later extended to work for participating media [37].

Other global illumination solutions are two-pass in nature. One such solution is volumetric photon mapping, first posed by Jensen and Christensen [25]. Photon mapping was the subject of Jensen's PhD dissertation [23] and had been published earlier as a means to compute caustics [24]. The basic operation of photon maps is to cast particles ("photons") from the scene light and compute intersections with scene geometry. At each of these interaction, a Monte Carlo process referred to as "Russian roulette" is executed which determines if a scattering or absorption event occurs at the interaction point. The photon map is then computed as a result of tracing the paths taken by the photons. It is then used in the rendering step as supplementary lighting information. More recent work in photon mapping involves a change to the photon map where the radiance is estimated along a beam, as opposed to a point [22].

Instant radiosity [32] is another means of approximating multiple scattering. In brief, rays are drawn from the primary light source to find the point of first reflection, where a virtual point light (VPL) is placed. The scene is then rendered with the contributions of every light added together to produce a final image. This technique is similar to another method often used in production where lights are placed on the interior of participating media to produce a glow characteristic of multiple scattering methods, while still only using a single scatter renderer. Instant radiosity has a bidirectional extension [55], Metropolis extension [56], and is frequently a basis of GPU algorithms for global illumination [38]. The idea of VPLs was extended to participating media [9], and the related idea of virtual ray lights was born [46].

An alternative, path-based method called Metropolis Light Transport was introduced by Eric Veach [63] and was the subject of his PhD dissertation [62]. The technique was developed under the assumption of hard surfaces, so there is not support for participating media. However, there are many shared ideas in the formulation and this one, such as the application of the Metropolis algorithm and local exploration of the path space by perturbation. Wenzel Jakob also builds on the idea of path-space exploration focusing particularly on specular surfaces [21, 20], and his techniques are implemented as a part of his open-source Mitsuba renderer [19]. Premože et al. [53, 52] show a path integral based computer graphics renderer which handle materials commonly associated with multiple scattering phenomena, such as clouds and milk. The technique was later implemented as a part of an OpenGL rendering pipeline [16].

The work of Bouthors et al. [3, 4], also the subject of Bouthors's PhD dissertation [2], focuses on real time Monte Carlo rendering of slab-shaped cloud structures. The technique works by analyzing the results of Monte Carlo simulations to elicit the macroscopic behavior of light traveling through various slab structures. A fit function is computed for these simulations so that the model can be extended to non-slab-shaped volumes and executed on a GPU.

The path integral formulation of radiative transfer was first published in 1987 [58].

The original lacked time dependence and was formed as an approximation for small angles. Time dependence was added in [59]. An effort to implement the path integral formulation in a numerical setting was begun in [60]. An approximation with intended application in a wide variety of scenarios is outlined in [61], and this work presents a full derivation in the appendix. This dissertation largely builds on the previously developed numerical framework in [60].

3.2 Medical imaging

Perelman et al. [48] developed a path integral framework for photon transport in turbid media, showing good agreement between their model and a Monte Carlo simulation, with later extensions [49, 47], and also noted the technique's promise as a theoretical model for understanding optical tomography [66].

3.3 Nuclear engineering

The field of nuclear engineering devotes a large effort toward the understanding of propagation of general purpose particles through various kinds of media. There are many Monte Carlo particle transport suites. Two of the more popular suites include MCNP6 [12], developed primarily at Los Alamos National Laboratory, and Geant4 [1], developed primarily at the European Organization for Nuclear Research. While nuclide transport tends to be a large focus, several of these suites do support electromagnetic radiation, including visible light. The methods used in these suites are understandably very particle-oriented like the traditional multiple scattering techniques employed by computer graphics. As well, they have a much larger scope than multiple scattering for photons so the concerns of the field are not often in line with those of computer graphics. Generally Monte Carlo techniques are the driving force, so there is a shared interest.

3.4 Ocean and atmospheric science

Scientists studying the ocean and atmosphere have long been interested in the behavior of light in the water and air. Especially over the long distances considered in some problems, multiple scattering becomes an obstacle to be understood and dealt with by developing models to describe it. Experimental and computational work are both common. Seibert Duntley [8] performed some early experimental measurement of the propagation of lasers underwater, including measuring both the beam spread function and point spread functions in a simulated tank with highly scattering media. He gives a regression for the data he collected. Later, others would follow in the validation of models for point spread functions [18], including Duntley's regression, and the measurement of beam spread functions for sea ice [40]. Monte Carlo techniques are also common. Kattawar et al. used Monte Carlo methods to simulate atmospheric emission [31] and scattering at the atmosphere-ocean interface [30]. Kattawar also was the editor for a book of methods for multiple scattering in ocean and atmospheric science [29]. Mobley et al. provide comparative work for underwater scattering numerical models and suggest a set of benchmark problems for solutions of similar type [44].

3.5 Geophysics

Radiative transfer is used within geophysics to model the propagation of seismic waves through the Earth. One of the earliest descriptions for a geophysics application was by Wu [67]. As simulations became more commonplace software packages like Radiative3D came to be. A technical report for Radiative3D [7] gives more detailed background on the use of radiative transfer algorithms for geophysics.

Chapter 4

Methodology

This chapter presents a numerical plan for implementing the path integral radiative transfer method. It is sufficient to integrate the spatial, angular, and path integrals, Equations (2.16) and (2.21), via Monte Carlo integration. In Section 4.1 we shall discuss geometries which we will use to model sensors and emitters. We need a way to generate *seed paths* based upon the geometry of the scene (Section 4.2). Creating seed paths can be expensive, so we use a perturbation method to create supplementary paths from a seed path (Section 4.3). Each path is then weighted and accumulated (Section 4.4) to produce the final image. An overview of the entire computational process is given in Figure 4.1.

4.1 Specifying geometry

In the world of Monte Carlo particle transport, emitters and sensors can be described by a *geometry*, a term we will borrow for this dissertation. The geometries we develop in this section will aid us in the integration over the $d\vec{x}_0$ and $d\Omega_0$ integrals of Equation (2.16). A geometry describes the set of points and directions an element of radiation can be emitted or sensed from, and the distribution they take on. Mathematically, we can think of a geometry as a joint probability density function (pdf), $G(\vec{x}, \hat{n})$, which gives the probability density over the entire possible set of positions and directions. G is defined over $\mathbb{R}^3$ and over the

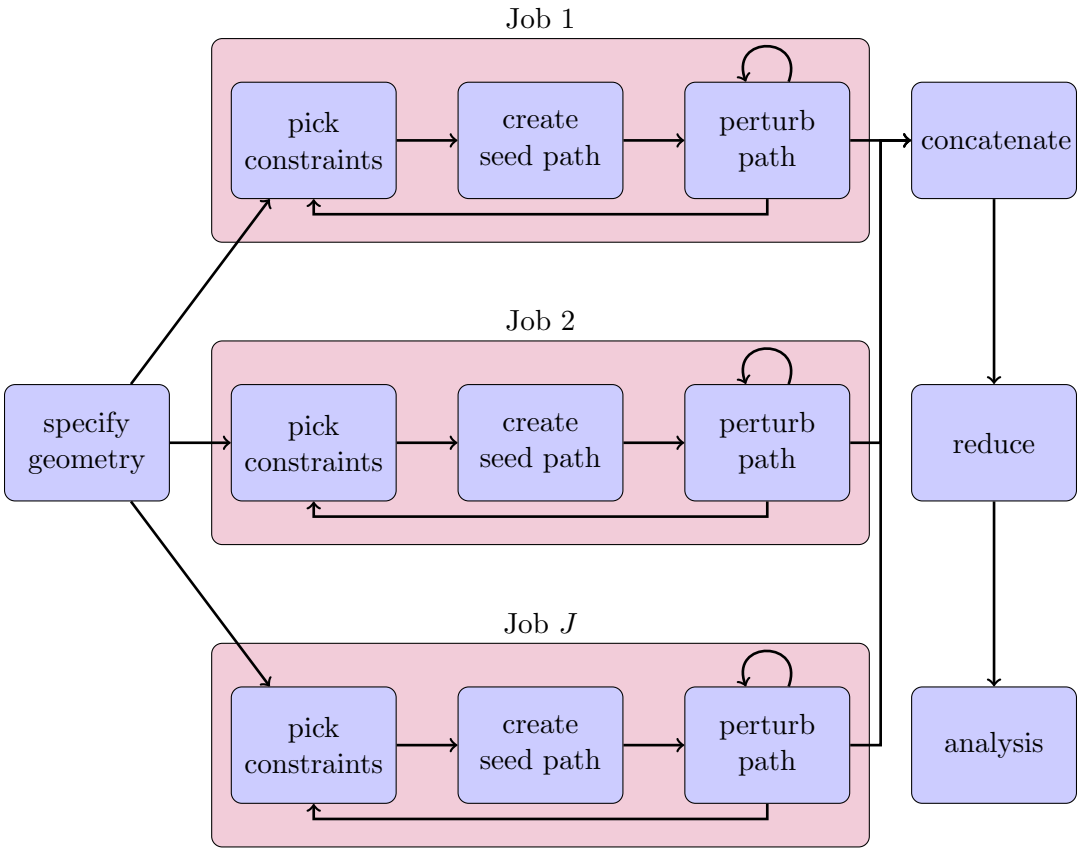


Figure 4.1: An overview of the computational process.
The experimenter specifies the desired emitter and sensor geometries via an input description file. Any desired number of compute jobs J can be attempted in parallel. Each sample will randomly pick boundary constraints from the geometry description, generate a seed path from the boundary constraints, and repeatedly perturb the seed path to calculate a single path integral sample. A job may repeat the process a configurable number of times. The jobs emits small sets of data which are concatenated. Each data item is the result of a single path integral calculation and accompanying metadata. The resulting data set is large, so it is reduced into a summary form suitable for analysis and plotting. Reduction depends on the type of analysis conducted but typically involves transforming and accumulating the data in terms of an alternate parameter, such as an (x, y) coordinate for 2D images or θ for beam spread functions.

solid angle sphere $\mathbf{\Omega}$. Note that G can be the product of two independent distributions. Conceptual examples of a geometry from computer graphics include point lights, which have fixed position and unconstrained direction and directional lights which have fixed direction but unconstrained position. Since G is a probability distribution we must have the condition

$$\int d\Omega \int d\vec{x}\ G(\vec{x}, \hat{n}) = 1. \tag{4.1}$$

We move on to formally specify some of the geometries used in this dissertation. The first geometry we will discuss is the simplest, which is the ray geometry. The ray geometry is conceptually similar to a laser: an emitter at a fixed position with a fixed direction. Its probability distribution is

$$G_{ray}(\vec{x}, \hat{n}) = \delta(\vec{x} - \vec{x}_r)\ \delta(\hat{n} - \hat{n}_r) \tag{4.2}$$

where $\vec{x}_r$ and $\hat{n}_r$ are parameters set by the experimenter. The ray is mostly useful as an emitter since it represents an ideal laser. It is also important from a numerical perspective as since it has only one defined value and can help reduce variance for simple experiments. Implementing the ray in a numerical setting is straightforward as it unconditionally returns $\vec{x}_r$ and $\hat{n}_r$.

A sphere geometry is also useful for recreating beam spread functions from optics (see Chapter 6). Conceptually we can think of the sphere as a radiometer tied by a taut string of length R from an anchor point $\vec{x}_s$. If we rotate our radiometer about the anchor point randomly, it forms the shape of the sphere. The radiometer is only sensitive in the direction of the string, and it spends an equal amount of time at all points of the sphere (i.e. it is spherically uniform). The pdf of the sphere geometry is

$$G_{sphere}(\vec{x}, \hat{n}) = \frac{1}{4\pi R^2}\ \delta(\|\vec{x} - \vec{x}_s\| - R)\ \delta\left(\frac{\vec{x} - \vec{x}_s}{\|\vec{x} - \vec{x}_s\|} - \hat{n}\right). \tag{4.3}$$

To generate this distribution numerically, we use the traditional spherical uniform sampling algorithm,

$$\hat{n} = \hat{x}\ \eta_1 + \hat{y}\ \cos(2\pi\xi_1)\ \sqrt{1-\eta_1^2} + \hat{z}\ \sin(2\pi\xi_1)\ \sqrt{1-\eta_1^2} \tag{4.4}$$

$$\vec{x} = \vec{x}_s + R\hat{n}, \tag{4.5}$$

where η_1 is uniform on $[-1, 1]$ and ξ_1 is uniform on $[0, 1]$.

There is one other alteration we can try with the sphere geometry. Suppose we assign a polar vector of interest on the sphere $\hat{p}$. The radiometer at any point of time is offset by an angle of $\theta = \arccos(\hat{n} \cdot \hat{p})$. Now suppose the radiometer spends an equal amount of time on the sphere for any given θ. We can simulate this numerically as well. Pick any vector perpendicular to $\hat{p}$ and call it $\hat{p}_\perp$. We can then calculate $\hat{n}$,

$$\hat{n} = R(\hat{p}, 2\pi\xi_1) \cdot R(\hat{p}_\perp, \pi\eta_1) \cdot \hat{p}, \tag{4.6}$$

where again η_1 is uniform on $[-1, 1]$, ξ_1 is uniform on $[0, 1]$, and $R(\hat{a}, \phi)$ gives the axis-angle rotation matrix for the axis $\hat{a}$ and angle ϕ. We calculate $\vec{x}$ as in Equation (4.5). This scheme is useful when you would like enforce uniform sampling with respect to θ, which can reduce variance for calculating beam spread functions in Chapter 6. However, this changes the pdf for the geometry to a non-uniform distribution. The effect of this is that the resulting calculation contains non-uniform bias and cannot be scaled uniformly for comparison.

Geometries are a useful convenience for concisely and precisely specifying how to generate the boundary constraints of our Monte Carlo sample. They are a reusable abstraction which can describe the pieces of a larger scenario. Similar, much more complete concepts exist in other particle transport packages [1, 12]. We impose the probability distribution function in our geometries to lay the framework for the incorporation of importance sampling, where it is important to be able to state the exact probability of a given sample being drawn.

4.2 Creating seed paths

We develop a strategy for creating seed paths. At this point, we will assume two pairs of directions and angles have been chosen using one of the geometries in Section 4.1. The emitter and sensor locations are at $(\vec{x}_0, \hat{n}_0)$ and $(\vec{x}_1, \hat{n}_1)$ respectively. The remaining arc length, s, is a free variable, but it is useful to control its distribution. Therefore we enforce that s is uniform on the interval $[\|\vec{x}_1 - \vec{x}_0\|, s_{max}]$ where s_{max} is left free. Ensuring s is adequately distributed will aid us in the integration over the ds integral of Equation (2.16). We seek to find a path having these end points, end directions, and arc length.

More formally, $\mathbf{r}(t)$ is a space curve defined on $t \in [0, 1]$ having the boundary conditions:

$$\mathbf{r}(0) = \vec{x}_0 \tag{4.7}$$

$$\mathbf{r}(1) = \vec{x}_1 \tag{4.8}$$

$$\frac{\partial}{\partial t}\mathbf{r}(0) = \hat{n}_0 \tag{4.9}$$

$$\frac{\partial}{\partial t}\mathbf{r}(1) = \hat{n}_1. \tag{4.10}$$

We impose the additional constraint that there is an arc length parameterization of the space curve $\mathbf{r}'(s')$ and it is defined on $s' \in [0, s]$. To be discretized properly we must have that

$$\vec{x}_1 = \vec{x}_0 + \Delta s \sum_{i=0}^{M-1} \frac{\partial}{\partial s'}\mathbf{r}'(i\Delta s) \tag{4.11}$$

where $\Delta s = s/M$, M being path subdivision count.

There are a number of different ways such a curve can be computed. We find using Bézier curves is an attractive option due to their ability to control both the position and tangent vector at the control vertices. The arc length may be varied by placing additional control vertices at different positions. We use a form for n-degree Bézier curves. $\mathbf{B}(t)$ is

defined on $t \in [0, 1]$,

$$\mathbf{B}(t) = \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n-i} t^i \vec{p}_i, \tag{4.12}$$

where $\vec{p}_i$ is one of $n+1$ control vertices. We can satisfy the constraints by constructing a Bézier curve with the following control vertices:

$$\vec{p}_0 = \vec{x}_0 \tag{4.13}$$

$$\vec{p}_1 = \vec{x}_0 + \ell_0 \hat{n}_0 \tag{4.14}$$

$$\vec{p}_2 = \vec{x}_1 - \ell_1 \hat{n}_1 \tag{4.15}$$

$$\vec{p}_3 = \vec{x}_1. \tag{4.16}$$

Here, ℓ_0 and ℓ_1 are both independent uniform random variables, each distributed over the interval $[0, 0.5 \times \|\vec{x}_1 - \vec{x}_0\|]$. This properly constrains both the position and tangent vector at the space curve's endpoints. The multiplication of the random scalars adds some measure of arc length variation.

There is one last consideration. Let us consider the case where there exists some d such that $\vec{x}_0 + d\hat{n}_0 = \vec{x}_1$, or less formally, where $\vec{x}_1$ lies along the ray based at $\vec{x}_0$ traveling in the direction of $\hat{n}_0$. This presents a problem when $\hat{n}_0 \approx \hat{n}_1$. It is difficult to generate anything but a straight path just with these four control vertices. Therefore, we should add a fifth control vertex as a handle for increasing the arc length. A reasonable choice is $(\vec{x}_0 + \vec{x}_1)/2 + \ell_2 \hat{n}_2$, where $\hat{n}_2$ is a spherically-uniform random unit vector and ℓ_2 is computed. The list of control vertices is then

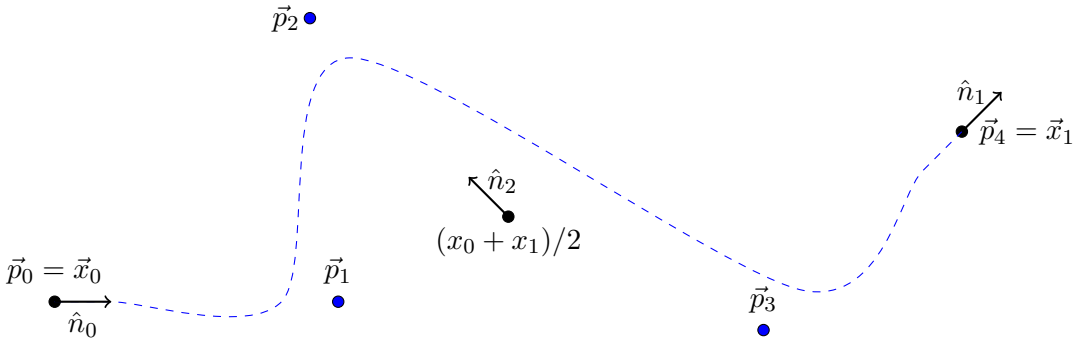


Figure 4.2: Diagram of Bézier curve control points. Here, $\vec{x}_0$ and $\vec{x}_1$ are chosen directly from the light and sensor geometry and serve as control points. We compute $\vec{p}_1$ and $\vec{p}_3$ as a function of these two points. $\vec{p}_2$ is chosen randomly by starting at the midpoint of $\vec{x}_0$ and $\vec{x}_1$ and tracing a random distance out from a spherically uniform vector $\hat{n}_2$.

$$\vec{p}_0 = \vec{x}_0 \tag{4.17}$$

$$\vec{p}_1 = \vec{x}_0 + \ell_0 \hat{n}_0 \tag{4.18}$$

$$\vec{p}_2 = (\vec{x}_0 + \vec{x}_1)/2 + \ell_2 \hat{n}_2 \tag{4.19}$$

$$\vec{p}_3 = \vec{x}_1 - \ell_1 \hat{n}_1 \tag{4.20}$$

$$\vec{p}_4 = \vec{x}_1. \tag{4.21}$$

Figure 4.2 offers a diagram showing the relationship between the terms.

The ℓ_2 parameter is used to constrain the arc length of $\mathbf{B}$ to s. Form a bracketing interval on s by allowing ℓ_2 to be any value in the range $[0, \ell_{max}]$. Once this bracketing interval is established, use any root finding mechanism to find an ℓ_2 such that the arc length of $\mathbf{B}$ is s. Brent's method is suitable for this step, and ℓ_{max} can be any sufficiently large static number. Alternatively, ℓ_{max} could be dynamically computed via a doubling procedure.

Now, we have selected a space curve $\mathbf{r}(t)$ which satisfies our constraints. This is sufficient information to define some of the initial parameters of the equivalent discrete

Frenet-Serret curve. The first parameter is the initial position:

$$x_0 = \mathbf{r}(0). \tag{4.22}$$

Similarly, we must find the initial tangent vector:

$$\mathbf{T}_0 = \hat{n}_0 = \mathbf{r}'(0). \tag{4.23}$$

On calculating the derivative of the space curve, it is sufficient to resort to numerical methods via finite difference, as the precision of analytic methods is not necessary in this case. The remainder of the orthonormal frame is the Frenet-Serret frame:

$$\mathbf{N}_0 = \mathbf{r}''(0) \tag{4.24}$$

$$\mathbf{B}_0 = \mathbf{T}_0 \times \mathbf{N}_0. \tag{4.25}$$

Finally, it is left to define the step size, Δs. We define this with respect to the number of desired path subdivisions, M, which remains a free parameter:

$$\Delta s = s/M. \tag{4.26}$$

We continue to find an arc length parameterization of the space curve. The arc length parameterization is necessary due to the path integral formulation requiring constant step sizes of Δs. Calculating arc length parameterizations is not trivial because the arc length function is not invertible. The arc length function $a(t)$ of our space curve is defined

$$a(t) = \int_0^t dt' \ \left\|\mathbf{r}'(t')\right\| \tag{4.27}$$

and given our space curve is defined by Equation (4.12) it is not invertible analytically. We have to rely on numerical methods.

Finding arc length parameterizations of Bézier curves and parametric curves in general is an area of much interest. Many approaches seek to find approximate solutions for a speed trade-off [39, 65]. Guenter and Parent [15] numerically calculate $a^{-1}(s)$ in a table to find bracketing intervals. The bracketing intervals aid a Newton solver in computing a more exact solution.

Our solution, which is similar to Guenter and Parent's, follows. Choose a positive integer k which becomes the size of the inverse arc length table. Ideally k should be a number much less than M, as the idea is to reduce the number of inverse arc length calculations from M down to k. For example, we use $k = M/20$. Choose a sequence of arc lengths to calculate $a_j = j \times s/k$. Form a table of the matching parameter-space values by solving the inverse arc length function numerically with Brent's method:

$$
\begin{aligned}
t_1 &= a^{-1}(a_1) \\
t_2 &= a^{-1}(a_2) \\
&... \\
t_{k-1} &= a^{-1}(a_{k-1}) \\
t_k &= a^{-1}(a_k).
\end{aligned}
$$

Brent's method can solve the inverse arc length function by optimizing $a(t) - s = 0$ for t. Once the table is calculated, form a second sequence of arc lengths $s_i = i \times s/M$. For each s_i there exists a a_j and a a_{j+1} such that $a_j < s_i \leq a_{j+1}$. The corresponding curve parameter can be approximated with linear interpolation. Let $q = (s_i - a_j)/(a_{j+1} - a_j)$:

$$t_i = qt_j + (1 - q)t_{j+1}. \tag{4.28}$$

At this point, our technique differs from Guenter and Parent. They use the interpolated value to serve as an estimate for a root solver, and find a precise value by root finding over

the subinterval. In practice, root-finding on this subinterval is quite slow for our purposes. Instead, we correct for the error after the curve is discretized. We sample the curvature function $\kappa(t)$ and the torsion function $\tau(t)$ along the sequence t_i. This generates the curve parameters

$$\kappa_i = \kappa(t_i) \tag{4.29}$$

$$\tau_i = \tau(t_i). \tag{4.30}$$

These form the last remaining discrete Frenet-Serret parameters. Once the space curve is discretized we can correct for the error introduced by the interpolation, particularly, the end point $\vec{x}_M$ and end direction $\mathbf{T}_M$. The perturbation algorithm in Section 4.3 is an excellent way to correct for this kind of error as it is highly tailored to this purpose.

4.3 Perturbing paths

We turn our attention to integrating over the path integral itself, Equation (2.21). At this point all the boundary constraints are fixed. For $s > \|\vec{x}_1 - \vec{x}_0\|$, there are an infinite number of *path configurations* which can satisfy the constraints. We can describe a path configuration as a vector of parameters which governs the path's trajectory, K. For a discrete Frenet-Serret curve (Section 2.2), this is a $2M$-dimensional vector consisting of the sequence of its segment curvatures and torsions, $K = \{(\kappa_0, \tau_0), (\kappa_1, \tau_1), ..., (\kappa_{M-2}, \tau_{M-2}), (\kappa_{M-1}, \tau_{M-1})\}$. To perform Monte Carlo integration over the path integral, we must find enough path configurations satisfying Equations (2.18) to (2.20) such that the variance of the estimate is below a reasonable tolerance value. To state the problem more precisely, we are given K_0 from Section 4.2. This chapter describes a method of efficiently generating a sequence K_i based on K_0 such that Equations (2.18) to (2.20) are preserved.

Previous work [60] proposes an algorithm for accomplishing this. A formal description of this work follows. We start by defining the points along the Frenet-Serret curve,

Equation (4.31), and the orthonormal frames at each of those points, Equation (4.32):

$$\vec{x}_i = \Delta s \mathbf{T}_i + \vec{x}_{i-1} \tag{4.31}$$

$$F_i = U_i F_{i-1}. \tag{4.32}$$

We take F_i to be a matrix consisting of the tangent (written as $\mathbf{T}_i$), normal ($\mathbf{N}_i$), and binormal ($\mathbf{B}_i$) vectors written as the rows of the matrix, $F_i = [\mathbf{T}_i, \mathbf{N}_i, \mathbf{B}_i]$. F_0 and $\vec{x}_0$ are given parameters of the curve, using the method described in Section 4.2. We take M to be the number of points defined along the curve. κ_i and τ_i are respectively the curvature and torsion at the i^{th} path segment. Recall that the U_i matrices describe an incremental transformation between orthonormal bases along the curve according to Equation (2.7). The algorithm is given as

1. Choose three random indices $0 \leq i_0 < i_1 < i_2 < M$.
2. Let $\kappa_{i_1} \in [a, b]$ such that $0 \leq a < b$.
3. Solve a five-dimensional equation

$$G(Y) = 0 \tag{4.33}$$

where

$$Y = \begin{bmatrix} \kappa_{i_0} \\ \kappa_{i_2} \\ \tau_{i_0} \\ \tau_{i_1} \\ \tau_{i_2} \end{bmatrix} \qquad G(Y) = \begin{bmatrix} x_M - x'_M \\ y_M - y'_M \\ z_M - z'_M \\ \|\mathbf{T}_M - \mathbf{T}'_M\|_1 \\ \|\mathbf{T}_M - \mathbf{T}'_M\|_2^2 \end{bmatrix}. \tag{4.34}$$

In this case, x_M, y_M, and z_M are the original, unperturbed components of the endpoint

coordinate of the curve; x'_M, y'_M, and z'_M are the perturbed components; and $\mathbf{T_M}'$ is the perturbed end tangent vector. We use $\|\cdot\|_i$ as notation for the i^{th} norm. G is a $\mathbb{R}^5 \mapsto \mathbb{R}^5$ function optimized when the curve is valid.

4. If the optimization failed to converge, restore the curve to its original state.

   A failure to converge can occur for many reasons, but often it is simply that the choice of indices or κ_{i_1} made it difficult for the root solver to resolve a solution.

5. Optionally reject the curve according to a Metropolis sampling algorithm.

4.3.1 Accelerated path generation

A typical root solver might look something like Algorithm 1. The *endpoint*() and *endtangent*() routines calculate $\vec{x}_M$ and $\mathbf{T}_M$ respectively. The *perturb*() routine creates a random perturbation of the curve's parameters. The *select*() routine chooses the i_0 and i_2 to use during optimization and *iterate*() is the root solver's specific method to update the optimized parameters. Then, *update*() actually applies the parameter update.

Algorithm 1 A typical root solver routine.

Require: $C = (\vec{x}_0, F_0, K, s)$
Ensure: $endpoint(C) = \vec{x}_{goal} \wedge\ endtangent(C) = \mathbf{T}_{goal}$
  $K' \leftarrow perturb(K)$
  $C' \leftarrow (\vec{x}_0, F_0, K', s)$
  $Y \leftarrow select(K')$
  while $\neg solve(C', Y)$ **do**
    $Y \leftarrow iterate(Y)$
    $update(C', Y)$
  end while
  if *converged*() **then**
    $update(C, Y)$
  end if

The *solve*() routine needs to calculate the end point and end tangent of the input curve, in order to satisfy the ensure clause. A first instinct might be to use Equation (4.31) and Equation (4.32) explicitly, which takes $O(M)$ matrix multiplications and vector additions. Let us assume that our root solver needs to evaluate the endpoint and tangent a

combined total of $O(P)$ times, therefore our current algorithm runs in $O(MP)$ time. In order to speed this up, we must make *endpoint*() and *endtangent*() faster. Let us consider the case of finding the end tangent with only one point of perturbation, index i_0. Expanding Equation (4.32), we get a chain of matrix multiplications,

$$F_M = \underbrace{U_M U_{M-1} ...}_{A_1} U_{i_0} \underbrace{...U_1 U_0}_{A_2} F_0. \tag{4.35}$$

Given that matrix multiplication is associative, we can precompute the annotated terms into A_1 and A_2 to save time. The idea easily extends to multiple points of perturbation:

$$F_M = \underbrace{U_M U_{M-1} ...}_{A_1} U_{i_2} \underbrace{...}_{A_2} U_{i_1} \underbrace{...}_{A_3} U_{i_0} \underbrace{...U_1 U_0}_{A_4} F_0. \tag{4.36}$$

For only an $O(M)$ penalty, we can precompute all of these matrices to make *endtangent*() run in $O(1)$ within a loop iteration. However, *endpoint*() still takes $O(M)$ time.

To speed up *endpoint*() is only slightly more difficult. At a desired point of perturbation i_0, there is a basis F_{i_0}. When this basis changes, the points $\vec{x}_{i_0+1}, \vec{x}_{i_0+2}, ..., \vec{x}_{M-1}, \vec{x}_M$ undergo a rigid rotation according to the change in the underlying basis. Also, notice one can find a straight line vector, or summary vector, of the path by taking the difference $\vec{x}_{i_1} - \vec{x}_{i_0}$, calling it $\vec{v}$. Then, by taking the sum of $\vec{x}_{i_0} + \vec{v}$, we find the next point of perturbation.

Since the curves undergo a rigid rotation, we can simply find a summary vector between the points of perturbation. It then appears that we could rotate these summary vectors with their respective bases to find the next point of perturbation. Mathematically, this means we project $\vec{v}$ into F_{i_0}, $\vec{v}_p = F_{i_0} \cdot \vec{v}$, and we remember these projections between iterations of the root solver. We allow the basis to become perturbed, written F'_{i_0}. This allows us to project the vector out of the modified basis to find the updated position of the next point of perturbation: $\vec{v}' = inv(F'_{i_0}) \cdot \vec{v}_p$. In summary, we find the following $O(1)$ means of calculating the perturbed end basis and tangent:

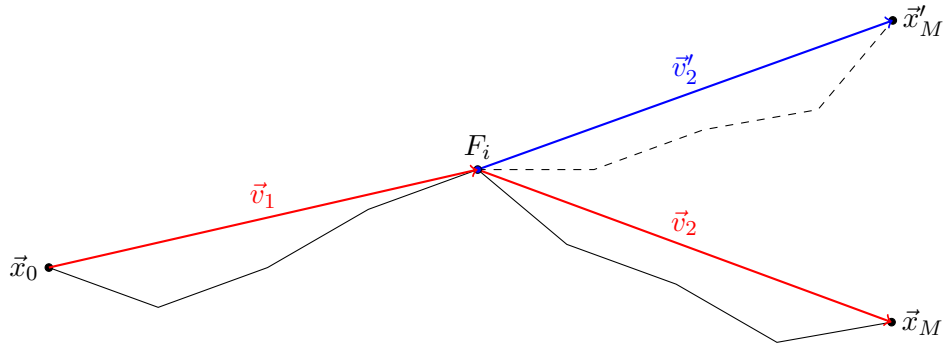


Figure 4.3: A diagram of the accelerated path perturbation algorithm, with a single point of perturbation i. Two vectors are drawn: $\vec{v}_1$ from $\vec{x}_0$ to $\vec{x}_i$ and $\vec{v}_2$ from $\vec{x}_i$ to $\vec{x}_M$. Imagining $\vec{v}_2$'s tail is attached to $\vec{x}_M$, then only $\vec{v}_2$ is affected by perturbation of the orthonormal basis F_i. By tracking the effect of the perturbation on the summary vectors $\vec{v}_i$ we avoid the need to trace along the potentially many path segments.

$$F'_M = A_4 U_{i_2} A_3 U_{i_1} A_2 U_{i_0} A_1 F_0 \tag{4.37}$$

$$\vec{x}'_M = \vec{x}_0 + \vec{v}_1 + inv(F'_{i_0}) \cdot \vec{v}_{2p} + inv(F'_{i_1}) \cdot \vec{v}_{3p} + inv(F'_{i_2}) \cdot \vec{v}_{4p}. \tag{4.38}$$

Algorithm 2 shows the revised root solver routine. The relevant additions are the $O(M)$ operations of $premultiplyBases()$ and $presumVectors()$, which are implementations of the optimization discussed previously. Given that there are $O(P)$ function evaluations in the root solver, we have reduced the algorithm to being $O(M + P)$ overall.

4.3.2 Root finder performance benchmark

An implementation of the described path generation algorithm was created using Python and SciPy [26]. The implementation includes the revised root solving algorithm and caches the computation of the U_i matrices where possible. A large portion of the computation is devoted to a root solving algorithm, and SciPy offers MINPACK's Powell hybrid method (*hybr*), MINPACK's Levenberg-Marquardt (*lm*), and several Newton methods. Among those, there are multiple methods to calculate the Jacobian, including Broyden's good method (*broyden1*), Broyden's bad method (*broyden2*), Anderson mixing (*anderson*),

Algorithm 2 Revised root solver routine.

Require: $C = (\vec{x}_0, F_0, K, s)$
Ensure: $endpoint(C) = \vec{x}_{goal} \wedge\ endtangent(C) = \mathbf{T}_{goal}$
  $K' \leftarrow perturb(K)$
  $C' \leftarrow (\vec{x}_0, F_0, K', s)$
  $A_1, A_2, A_3, A_4 \leftarrow premultiplyBases(C')$
  $\vec{v}_1, \vec{v}_2, \vec{v}_3, \vec{v}_4 \leftarrow presumVectors(C')$
  $Y \leftarrow select(K')$
  while $\neg solve(C', Y)$ **do**
    $Y \leftarrow iterate(Y)$
    $update(C', Y)$
  end while
  if $converged()$ **then**
    $update(C, Y)$
  end if

Method	Time (s)	Perturbations	CPCH	Evaluations
Broyden's good method	541.92	338.30	2250.93 ± 4.81	-
`excitingmixing`	1297.75	0.11	0.32 ± 0.37	-
Powell's hybrid method	**66.79**	667.34	**36108.04** ± 877.22	**197476.51**
Newton-Krylov	840.93	662.57	2855.83 ± 98.04	-
`linearmixing`	1202.42	0.17	0.51 ± 0.38	-
Levenberg-Marquardt	121.22	**828.17**	24662.43 ± 511.32	382549.86

Table 4.1: Results of the root solver performance benchmark. A confidence interval for CPCH is shown for $\alpha = 0.025$.

Krylov approximation (*krylov*), Diagonal Broyden approximation (*diagbroyden*), and others (*linearmixing* and *excitingmixing*).

An experiment was designed to test the efficacy of each of these methods for this particular algorithm. For each method, 35 single core jobs were launched on a supercomputer perturbing a curve 1000 times. For these jobs, the time spent generating paths, the number of successful perturbations, and the number of function evaluations (if available) were recorded. Table 4.1 shows the acquired results. The means of the job time and number of curves generated are shown. Curves generated per core-hour (CPCH) is derived by measuring a simple rate (curves generated per unit hour) over each individual job and finding the mean.

Of the methods available, many of them would diverge, suggesting that they are not a good fit for this situation; they are not included in the table. In terms of CPCH, Powell's hybrid method is a clear winner. Levenberg-Marquardt lags Powell's hybrid method in performance, but interestingly seems to converge more often.

4.4 Weighting paths

Once a path is generated we must be able to assign a weight to the path efficiently. In the derivation presented in [60], a weighting function is derived for a single node of a Frenet-Serret curve,

$$\omega_n = \exp(-c_n \Delta s) \int \frac{d^3\mathbf{p}}{(2\pi)^3} \exp(i\mathbf{p} \cdot \hat{\mathbf{N}}_n \kappa_n \Delta s + b_n \Delta s \tilde{Z}(|\mathbf{p}|)), \tag{4.39}$$

where the weight of the entire path is just the product over the nodes. In fact, an error was discovered in the original paper and several of the derived equations after being written in spherical coordinates are incorrect. Rewriting in spherical coordinates and taking $\mathbf{p} \cdot \hat{\mathbf{N}}_n = p\cos\theta$ will yield

$$\omega_n = \exp(-c_n \Delta s) \int_0^\infty \frac{p}{2\pi^2} \exp(b_n \Delta s \tilde{Z}(p)) \frac{\sin(p\kappa_n \Delta s)}{\kappa_n \Delta s}\, dp. \tag{4.40}$$

For a complete derivation, see Appendix A. We focus only on the regularized form,

$$\omega_n = \exp(-c_n \Delta s) \int_0^\infty \frac{p}{2\pi^2} \exp(b_n \Delta s \tilde{Z}(p) - \frac{\epsilon^2}{2} p^2) \frac{\sin(p\kappa_n \Delta s)}{\kappa_n \Delta s}\, dp. \tag{4.41}$$

It is desirable to normalize ω_n. Its value can be greater than 1 and cause overflow during multiplication to compute the path weight. For the case where b_n is a constant, ω_n peaks at $k\Delta s = 0$, which is used for normalization here. Many of the terms are discussed in greater length in Chapter 2. They are:

- Δs and κ_n which are the step size and the curvature of the Frenet-Serret curve;

- a_n, b_n and c_n, which are respectively absorption coefficient, scattering coefficient, and total extinction $(a_n + b_n)$ at position n;
- $\tilde{Z}(p)$, the Fourier transform of the phase function;
- ϵ, a tunable parameter to make the integral more numerically feasible.

While a_n and b_n can be any scalar field, here we focus on the case where they are constants. Smaller values of ϵ approach the theoretical value of the integral, but are also more sensitive to numerical noise. For $\tilde{Z}(p)$, we use the transformed Gaussian phase function presented in [61],

$$\tilde{Z}(p) = N_p \exp\left(\frac{-\mu p^2}{2}\right), \tag{4.42}$$

which is derived from the phase function,

$$P(\hat{n}, \hat{n}') = \frac{2N_p}{(2\pi\mu)^{3/2}} \exp\left(\frac{\hat{n} \cdot \hat{n}' - 1}{\mu}\right), \tag{4.43}$$

with the normalization constant

$$N_p = \frac{\sqrt{\pi\mu/2}}{1 - \exp(-2/\mu)}. \tag{4.44}$$

Cutting the integration bounds off at $10/\epsilon$ is sufficient for capturing the important parts of the integrand while avoiding numerical noise in the result. The bulk of the contribution is well within those bounds of integration, and anything beyond will likely only contribute numerical noise.

It is also helpful to understand how modifying the parameters affects the overall character of the weight function. Figure 4.4 shows increasing μ widens the Gaussian phase function and therefore the weight function tails off over a longer period of time. However, the difference between values of μ is quite small. Figure 4.5 shows that ϵ corresponds to the width of the peak of the delta function and increases the maximum of the function.

Figure 4.6 shows increasing $b\Delta s$ causes the weight in general to decrease at a slower rate, though we are not typically interested in large values of $b\Delta s$. This plot also shows that the smoothness of the weight curve does tend to break down at a particular shelf.

It is expensive to calculate ω_n numerically. However, we can observe that the integrand of Equation (4.41) is dependent on the terms $\kappa\Delta s$ and $b\Delta s$, evidenced by Figure 4.7. The phase function and other terms are not spatially varying. Inspection of Figures 4.6 and 4.7 shows ω_n is quite smooth for most reasonable values. Therefore we can bilinearly interpolate over $\kappa\Delta s$ and $b\Delta s$ to avoid repeated calculation of the integral portion of ω_n. This results in a fairly significant gain in performance.

The weight of a single path is a product of its segment weights:

$$\Omega = \prod_{i}^{M} \omega_i. \tag{4.45}$$

Here, ω_i is the i^{th} path segment's weight, as determined by the regularized path segment weight formula, Equation (4.41). Assuming a uniform distribution in the perturbation scheme, Monte Carlo integration over N paths satisfying the boundary constraints yields

$$D\frac{1}{N}\sum_{i=1}^{N}\Omega_i \tag{4.46}$$

where D is a normalizing factor. This quantity, in the limit of N, approaches the value of the transport kernel:

$$G(s, \vec{x}_1, \hat{n}_1, \vec{x}_0, \hat{n}_0) = \lim_{N\to\infty} D\frac{1}{N}\sum_{i=1}^{N}\Omega_i \tag{4.47}$$

This forms the numerical basis by which we evaluate the path integral. First we choose the boundary constraints to hold constant, which are the function parameters of the transport kernel G. We choose a seed path which satisfies the boundary constraints. From the seed path, we use the perturbation algorithm to create other paths which satisfy the boundary constraints, yielding a sequence of path weights Ω_i. These path weights may be integrated

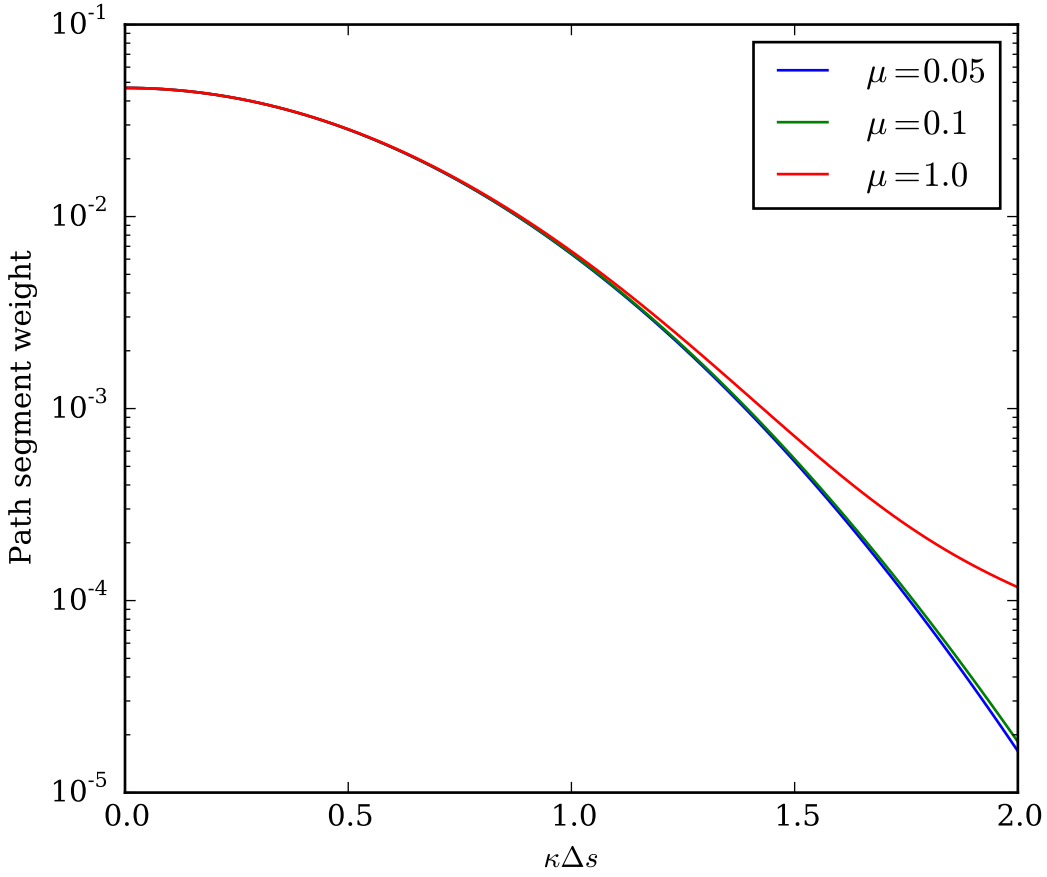


Figure 4.4: Path segment weight function with varying phase function μ. Here, we have the regularization value $\epsilon = 0.5$ and $b\Delta s = 0.08$. We observe that increasing the width of the Gaussian phase function causes the weight to fall off more slowly with increasing $\kappa\Delta s$, though the difference is very slight even for drastic changes in μ. Larger Gaussian widths correspond to more large angle scattering events, so we should expect higher curvature values to be penalized less.

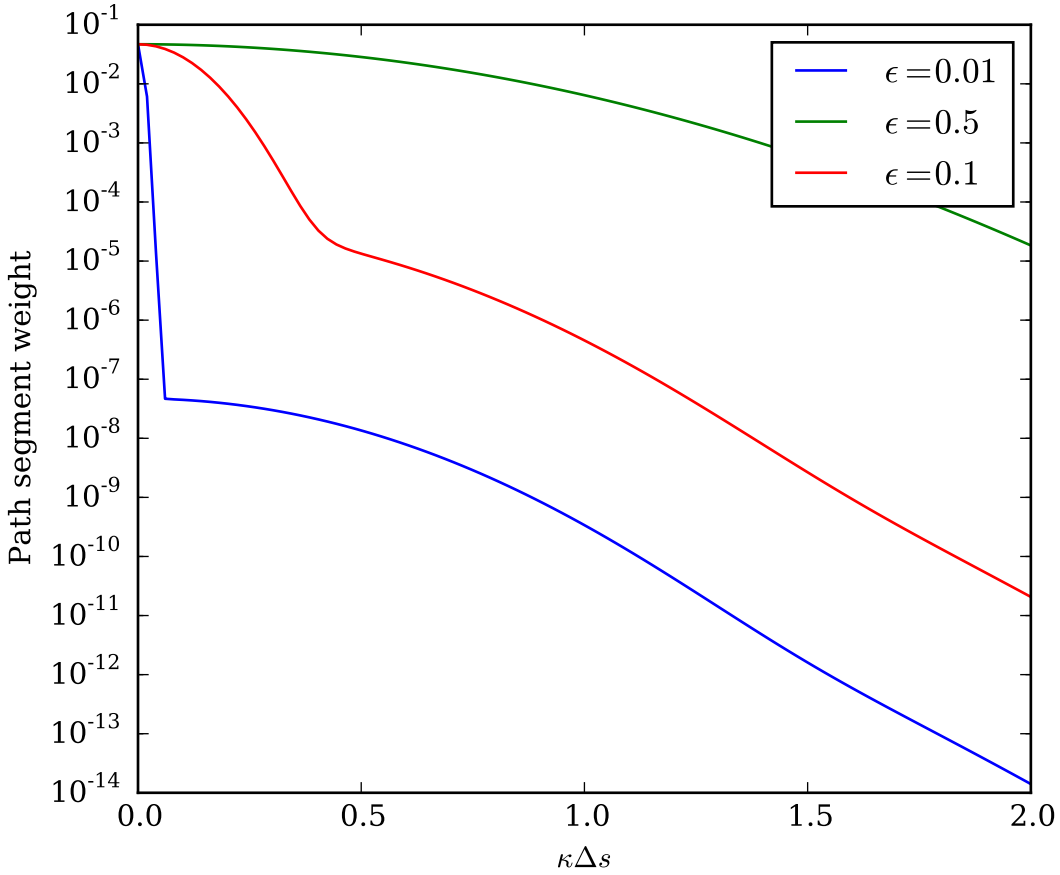


Figure 4.5: Path segment weight function with varying regularization value ϵ. Here, we have the Gaussian phase function parameter $\mu = 0.1$ and $b\Delta s = 0.08$. We can see as $\epsilon \to 0$, a delta function is more prominent at $k\Delta s = 0$, and higher values of $k\Delta s$ are weighted less.

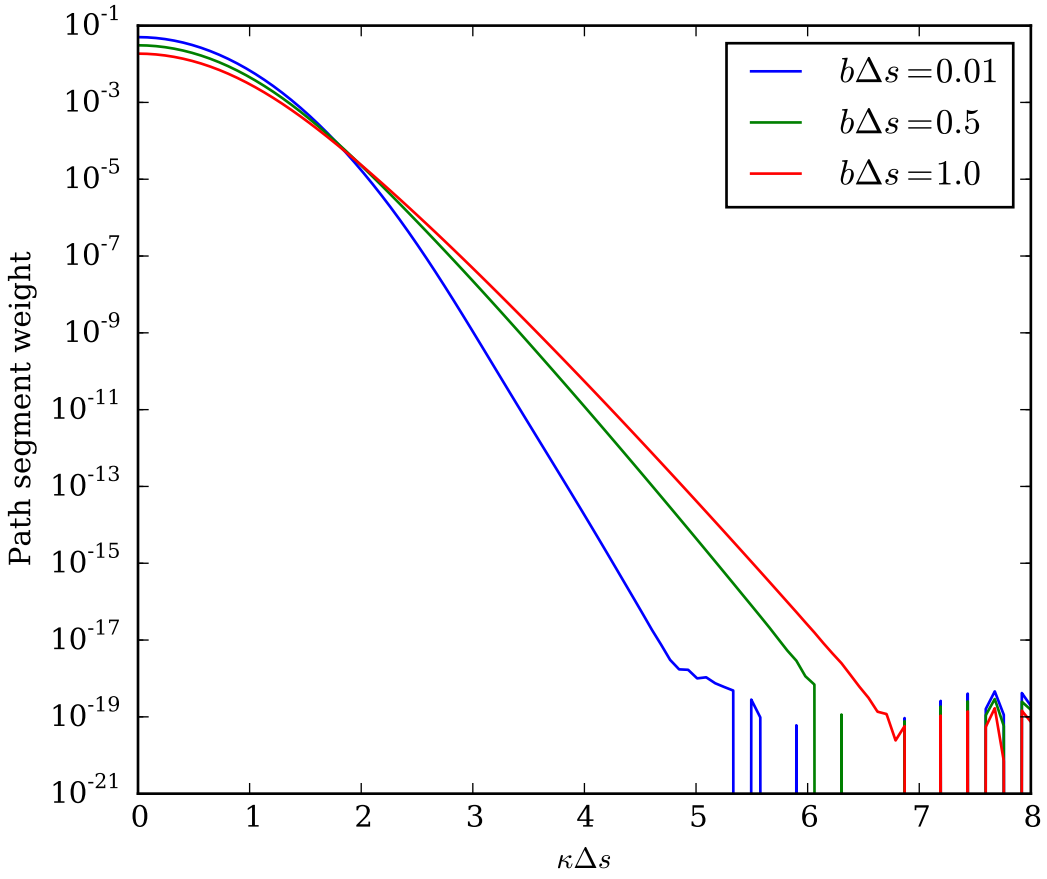


Figure 4.6: Path segment weight function varying the product of the scattering value and step size, $b\Delta s$. Here, the Gaussian phase function parameter $\mu = 0.1$, and the regularization value $\epsilon = 0.5$. As higher scattering values are reached, paths of higher curvature are weighted higher, but the peak decreases. Also shown here are some ranges where the path segment weight function breaks down numerically in higher areas of $\kappa\Delta s$.

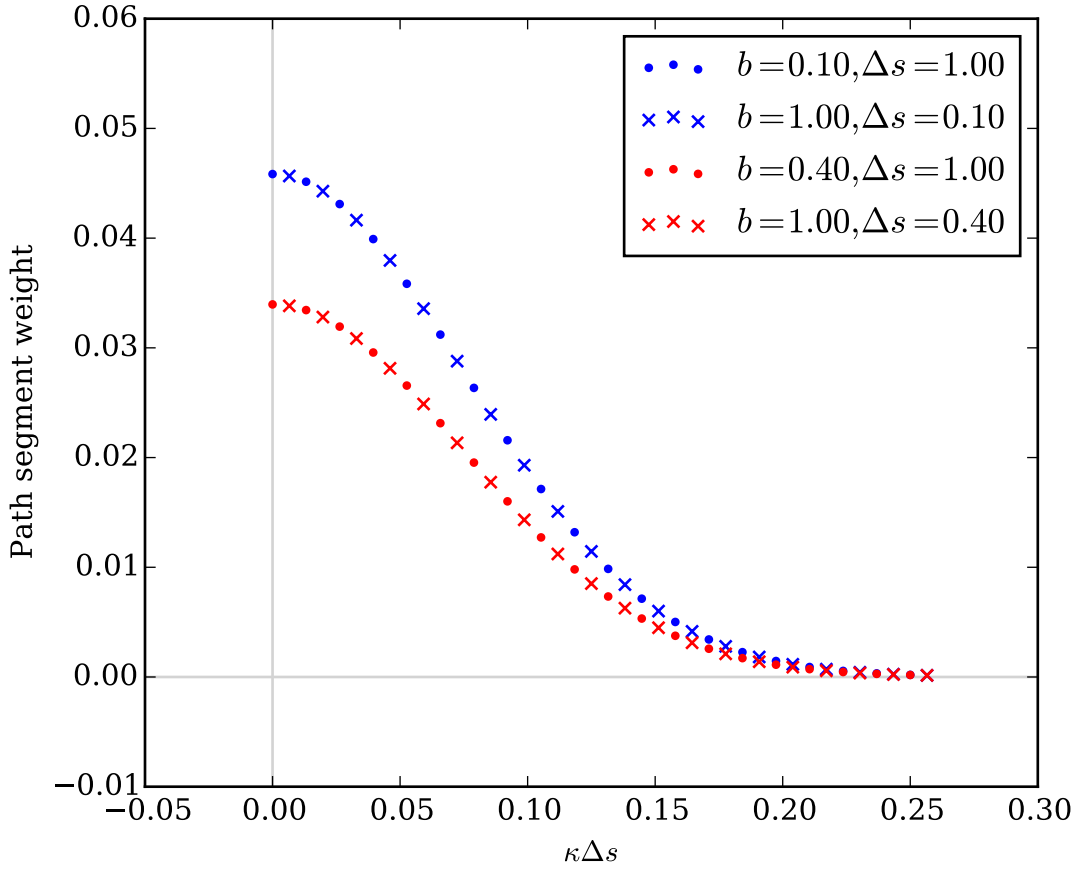


Figure 4.7: Plot of path segment weight function where the product $b\Delta s$ is the same. In the case where $b\Delta s$ is equal the data are colored the same. Notice that the functions are equivalent for equal $b\Delta s$. This implies it is possible to cache calculations of the integral in Equation (4.41) for some significant performance gains.

via Monte Carlo integration as in Equation (4.47) to approximate the value of the transport kernel.

Chapter 5

Experiment: Statistics of path integral

Before conducting full scale experiments, the statistical behavior path integral should be known. The numerical integration of the path integral should be repeatable for a given set of boundary constraints. Equation (4.47) gives a plan for integration, however leaves the normalizing factor D undefined. For this and other experiments we ignore D as we are interested in the relative behavior. As we are working interested in the statistics of the numerical methods themselves, we are free to choose any boundary constraints we choose. For this experiment we (1) set $\vec{x}_0$, $\hat{n}_0$, s, κ_i and τ_i arbitrarily to generate a seed path and (2) repeatedly perturb the seed path k times. For a given job j we can form a sum of the path weights to represent them in a more compact form:

$$P_j = \sum_{i=1}^{k_j} \Omega_i. \tag{5.1}$$

Then between the j jobs there is a total number of perturbations

$$K_j = \sum_{i=1}^{j} k_i \tag{5.2}$$

Parameter	Value
Absorption coefficient (a)	0.0
Scattering coefficient (b)	1.0
Initial curvature (κ_i)	0.1
Initial torsion (τ_i)	0.1
Path segments (M)	200
Arc length (s)	16.0
Gaussian width (μ)	0.1
Regularization constant (ϵ)	0.5

Table 5.1: Experimental parameters used to study the statistics of the path integral calculation.

and considering the results of j jobs we define an estimator for the path weight to be

$$Q_j = \left(\sum_{d=1}^{j} P_d \right) / K_j. \tag{5.3}$$

A summary of experimental parameters are available in Table 5.1. The process can be repeated many times to generate independent perturbation sequences.

Monte Carlo integration of the path weights produces a running plot like Figure 5.1. Visually, the variance appears to decrease as the sample count increases. However, this is only one large sample set, and the estimate of the intensity could be biased. Therefore, we are left with a question if the particular sequence of samples biases the result of the calculation. To explore this question further, we can divide our large sample set into smaller slices, integrate the slices, and analyze the results.

Figure 5.2 illustrates the character of convergence if we sum over slices of varying size across the whole data set and create a histogram of the result. What we find is that the means of the histograms line up with the convergence predicted by Figure 5.1. A fit of the data places the mean of the log-normal distribution roughly in the same place. This suggests that if we choose any set of randomly generated curves and sum their weights, then they should roughly converge to the same mean.

The width of the distribution is also of interest since it will gauge how close to convergence the calculation is. Figure 5.3 shows the trend of the σ parameter for the

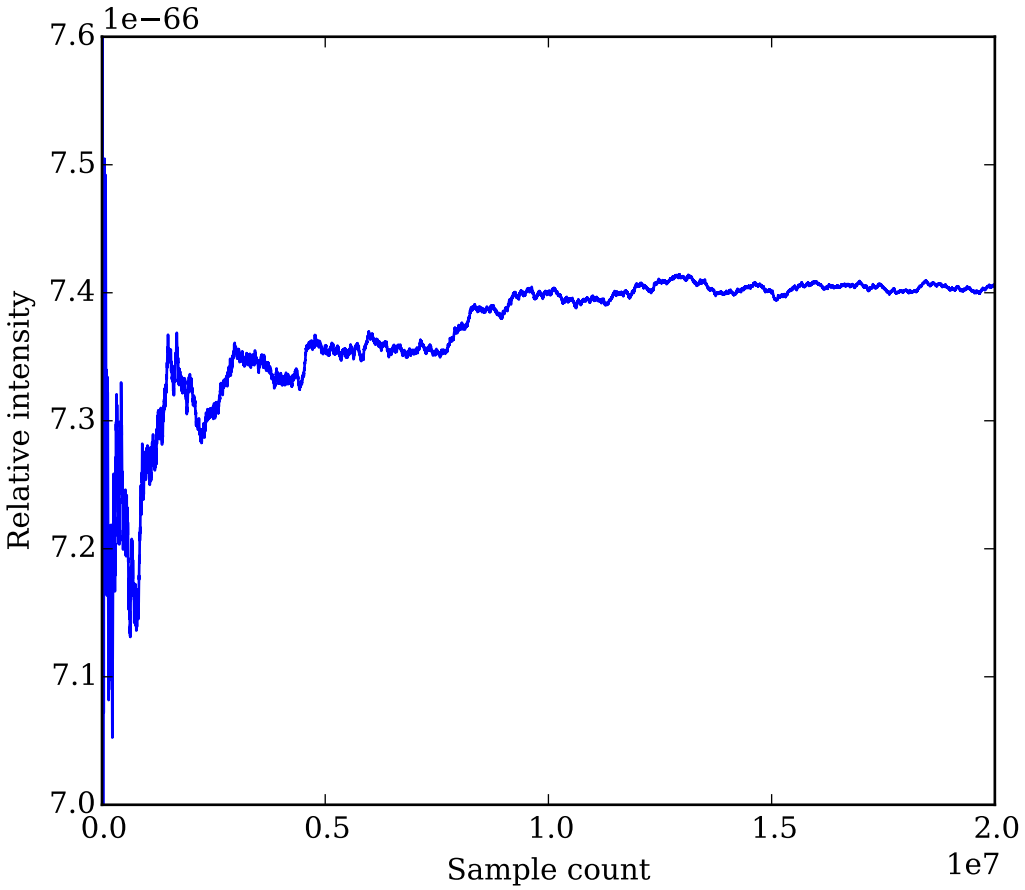


Figure 5.1: A running estimate of a path weight for a very large sample count. The relative intensity is Q_j and sample count is K_j. The estimated mean seems to reduce in noise after several million perturbation samples.

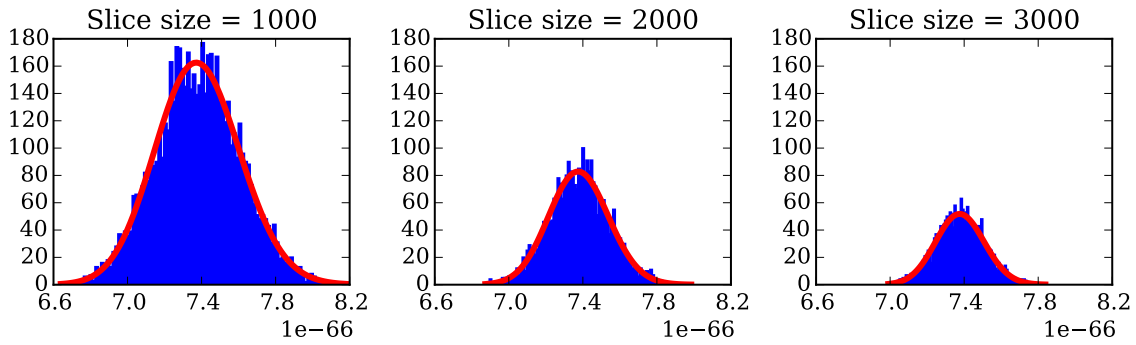


Figure 5.2: Log-normal fits for three different slice sizes: 1000, 2000, and 3000. The fit mean μ is consistent regardless of which sequence of paths, so there is not bias in this regard. Each of the means agrees with Figure 5.1. The width parameter σ decreases as we increase the number of sample runs. This decrease suggests we decrease the uncertainty of the mean estimate as we take more samples.

resulting log-normal fits against increasing slice size. We see a decreasing trend as expected, but this offers insight to the accuracy of the prediction of the mean as a function of the slice size. For instance, there is an obvious difference in the fit width between a slice size of 500 and 1000, but the difference is subtle between slices of 2000 and 3000.

This log-normal fit is useful for the Metropolis algorithm, as it provides a method of determining the acceptance probability of a given path weight. Let C_i be a given curve in the sampling sequence and Ω_i its weight; likewise C_c is a candidate curve and Ω_c its weight. Using the fit data we obtain a log-normal probability density function $p(x)$, and compute an acceptance probability $\alpha = min(1, p(\Omega_c)/p(\Omega_i))$. We let C_{i+1} be C_c with probability α, or C_i otherwise. Figure 5.4 shows a running integration using the Metropolis algorithm alongside the original approach. While the Metropolis approach does seem to stabilize a little earlier than the standard approach, the two converge to different values. It is possible there is some bias in one of the approaches. However, all other experiments do not use Metropolis acceleration with the motivation of finding a brute force answer to which future approaches can be compared.

In conclusion, this experiment studies the statistical behavior of the path integral calculation (Equation (2.21)) for an arbitrary set of boundary constraints. Analyzing the

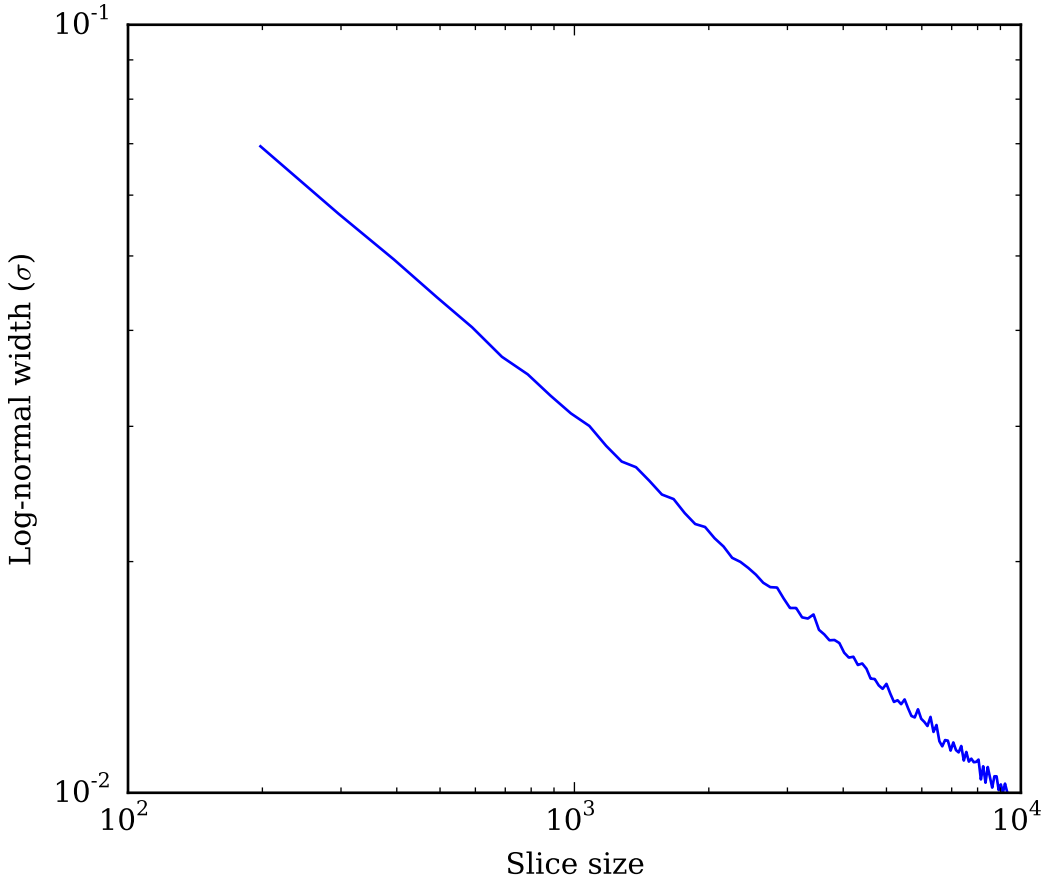
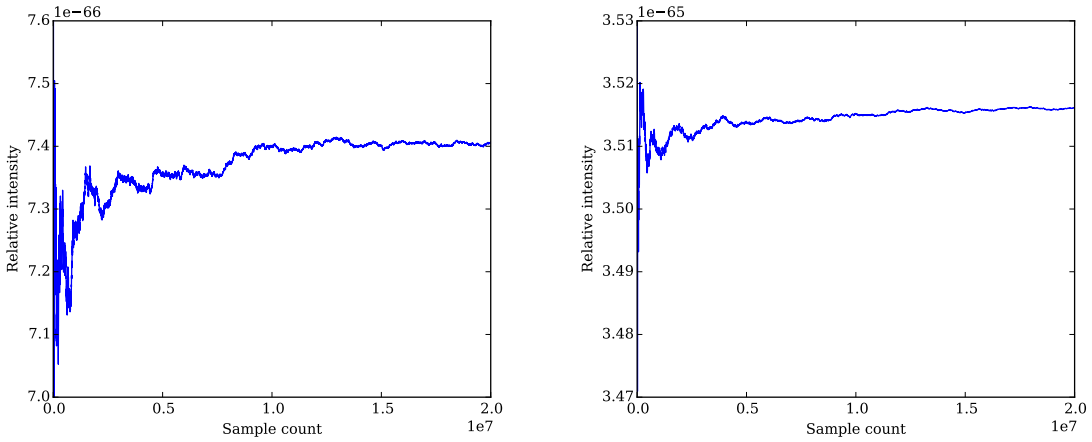


Figure 5.3: Log-normal width parameter σ for increasing slice size. This reaffirms what is shown in Figure 5.2, but plots the σ parameter with higher resolution with respect to the number of sample runs. We do see the expected trend in the decrease of σ.



(a) Standard sampling algorithm

(b) Metropolis acceleration scheme

Figure 5.4: Comparison of the standard sampling to Metropolis-accelerated scheme. The two approaches appear to converge upon different values almost an order of magnitude apart. However, the Metropolis-assisted scheme seems to stabilize a little earlier.

data set piece-wise shows the integration follows a log-normal distribution whose variance decreases as more samples are collected. This yields a possible basis for a Metropolis acceleration scheme for the perturbation algorithm. The results of this experiment proves that our path integral calculations are repeatable given enough samples. This information informs that the approach is ready to be tested against integration over solid angle and surface distributions, i.e., to calculate Equation (2.16). This allows comparison against real phenomena such as beam spread functions.

Chapter 6

Experiment: Beam spread functions

A beam spread function (BSF) is an experimentally measurable property of scattering media. Imagine a laser suspended at the origin in cloudy water, and a sphere located at the origin having radius R. The laser is illuminated, and the radiation travels through the cloudy water and eventually reaches the sphere's surface. If absolutely no scattering occurred, we should expect the sphere to be illuminated at a single point, $\vec{p}$. Scattering, however, directs the radiation such that the illumination pattern is a small spot which has a peak at $\vec{p}$ and gradually falls off as we move away from $\vec{p}$ on the sphere's surface.

Generally a BSF is measured by immersing a laser (or any collimated light source) and a radiometer within the scattering media and sweeping the laser across a range of angles to measure the irradiance with respect to angle θ. BSFs are an attractive experimental property for a few reasons. They are reasonably simple to specify geometrically. There is experimental data for simple optical cases which can be approximated by infinite media of uniform optical properties. Since only one angular variable, θ, is randomized there is far less variance to contend with in Monte Carlo applications such as the FPI approach. All the while, multiple scattering effects are visible in the larger values of θ where the expectation is that there is gradually decreasing intensity as θ increases.

Point spread functions (PSFs) are equivalent to BSFs [13] and are often measured experimentally as well. The seminal paper on BSFs and PSFs and their applications in optics is by Mertens and Replogle [41]. To summarize the scope of work completed in the measurement of both BSFs and PSFs, each has been measured for various types of media in both laboratory and field settings. This has yielded some results which computational techniques can use for validation. Some of the earliest work was by Duntley [8] with laboratory PSF measurements in highly scattering water tanks. Voss [64] measured the PSF of the Sargasso Sea water. Schoonmaker et al. [54] measured the BSF of laboratory-grown sea ice, and were able to fit a Gaussian model to their data. Maffione et al. [40] measured the BSF of Alaskan sea ice and attempted a Gaussian fit but found a Lorentzian form function which fit their data.

According to Mertens and Replogle [41], the BSF is the normalized irradiance distribution parameterized by the polar angles on the sphere's surface and the sphere's radius R, written $BSF(\theta, \phi, R)$. However, in the literature we find several different forms of the BSF. Assuming spherical symmetry allows for dimensionality reduction, as Maffione et al. [40] drop the ϕ to give $BSF(R, \theta)$. Schoonmaker et al. [54] show a different approach of measuring the BSF with a Gaussian beam by transforming the irradiance patterns of the Gaussian beam and the pattern on the sea ice surface. This defines the BSF as a function of R and z, the radial distance from the spot center as measured on the sea ice interface, $BSF(R, z)$.

Our approach is most similar to Maffione et al. The original experiment places the laser and radiometer in holes bored into the sea ice, and the laser is swept across an angle θ. The radiometer measures intensity with respect to this angle. Our modified approach is three-dimensional. A laser is placed at a fixed position and the radiometer is placed at any point on a sphere a distance R away from the laser, fixing its direction to the surface normal of the sphere. This is a geometrically equivalent description and has the benefit of checking for spherical symmetry in the BSF. We take $\theta = cos^{-1}(\hat{n}_0 \cdot \hat{n}_1)$. This gives the BSF as a function of R and θ, $BSF(R, \theta)$. Figure 6.1 provides an illustration of the experimental

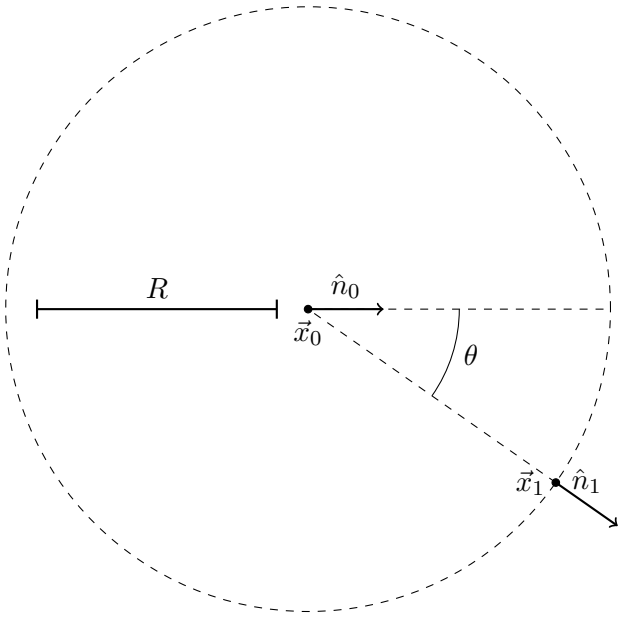


Figure 6.1: Illustration of beam spread function experimental setup. $\vec{x}_0$ and $\hat{n}_0$ are the laser position and direction, which are both constants. $\vec{x}_1$ is the radiometer position, which can be located on a sphere a distance R away from $\vec{x}_0$. $\hat{n}_1$ is the direction normal to the sphere's surface.

setup.

In analyzing BSFs, typically they are singly-peaked functions at 0 degrees and can have varying shape. We refer to Maffione et al. for their analysis. They use both a Gaussian and a Lorentzian function to fit their beam spread function data. The Gaussian has the form

$$Gauss(\theta) = A \exp\left(-\frac{\theta^2}{2\sigma^2}\right) \tag{6.1}$$

where they found the fit was most appropriate when A is constrained to the peak of the measured data. The Lorentzian function has the form

$$Lor(\theta) = \frac{A}{(\sigma/\theta)^2 + 1}. \tag{6.2}$$

Each of these are singly-peaked functions which can be used to fit the collected data, though this work focuses on using a Gaussian. Before fitting, a calculated BSF is normalized such

that its peak value is 0.025 and the θ domain is in degrees. These conventions are arbitrary, but the intent is to be able to compare with the data collected by Maffione et al. as directly as possible.

Specifically for our calculated data, there is the notion of *efficiency*, *sample count*, and *wall time*. The efficiency is the ratio of successful samples to the total number of samples. A sample in this case refers to a choice of boundary constraints, seed path, and the 1000 perturbations of the seed path. The wall time is the total sum of time spent calculating all of the samples (CPU + IO), but does not consider overhead incurred by the scheduler.

6.1 Scattering, absorption, and path length

There is a relationship between the scattering coefficient (b) and the path length value, written R in the BSF experimental setup. Both are related to distance values: b has units of inverse distance and R has units of distance. Conceptually, the quantity $(1/b)$ corresponds to an expected distance between scattering events and this is known as the *scattering length*. Therefore, the quantity Rb is related to the expected number of scattering events for a path length R. In the absence of absorption (i.e. $a = 0$) and barring changes to all other parameters, two BSFs should be equivalent when the quantity Rb is equal between them. ‘

To see how the FPI approach behaves in cases of equivalent scattering lengths, an experiment was designed. We vary the scattering coefficient b and path length R such that there are several levels of the experiment where Rb is equal. To limit total extinction, the absorption coefficient a is set to 0. The static parameters are available in Table 6.1 and the experiment levels are given in Table 6.2. The hypothesis for the experiment is that the FPI approach should produce identical BSFs when Rb is the same between the two, a consequence which should arise because of the path segment weight ω_n being dependent on the product $b\Delta s$ (see Figure 4.7 on page 40).

Parameter	Value
Absorption coefficient (a)	0.0
Maximum arc length (s_{max})	$\approx 1.5R$
Gaussian width (μ)	0.1
Regularization constant (ϵ)	0.075

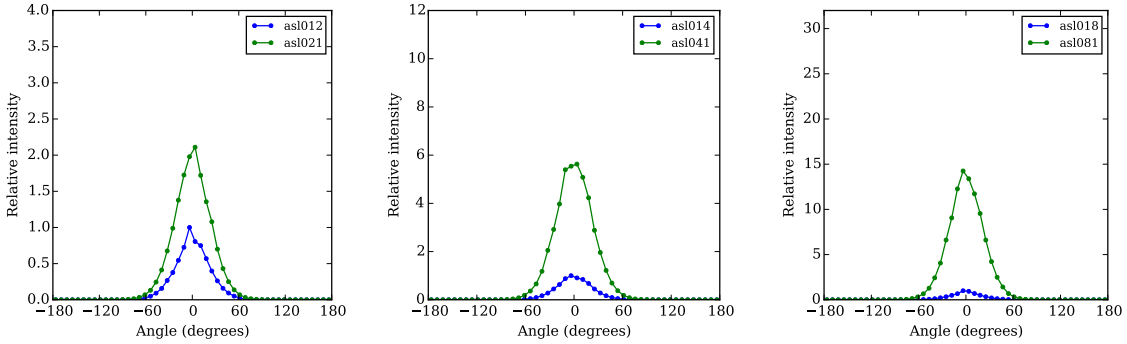
Table 6.1: The static set of parameters used for the scattering versus path length experiment. This experiment was conducted when s_{max} was relatively computed with respect to $\|\vec{x}_1 - \vec{x}_0\|$, and tended to produce arc lengths close to that value.

Label	Scattering (b)	Path length (R)	Scattering lengths (Rb)
asl011	1	1	1
asl012	1	2	2
asl014	1	4	4
asl018	1	8	8
asl021	2	1	2
asl041	4	1	4
asl081	8	1	8

Table 6.2: Table of varied parameters for the scattering versus path length experiment. These cases are constructed such that there are cases where the path lengths are equivalent multiples of scattering lengths.

	Efficiency	Gaussian width	Sample count	Wall time
asl011	64.69%	22.09	3.47e+05	0.95 years
asl012	64.31%	18.49	3.99e+05	1.18 years
asl014	63.62%	20.30	2.80e+05	0.83 years
asl018	62.92%	16.56	2.94e+05	0.87 years
asl021	64.77%	21.27	3.98e+05	1.16 years
asl041	64.83%	22.57	3.97e+05	1.17 years
asl081	64.52%	20.50	3.99e+05	1.16 years

Table 6.3: A summary of the collected data for the scattering versus path length experiment. There appears to be an inverse relationship between Gaussian width σ and path length R.



(a) $Rb = 2$ relative comparison (b) $Rb = 4$ relative comparison (c) $Rb = 8$ relative comparison

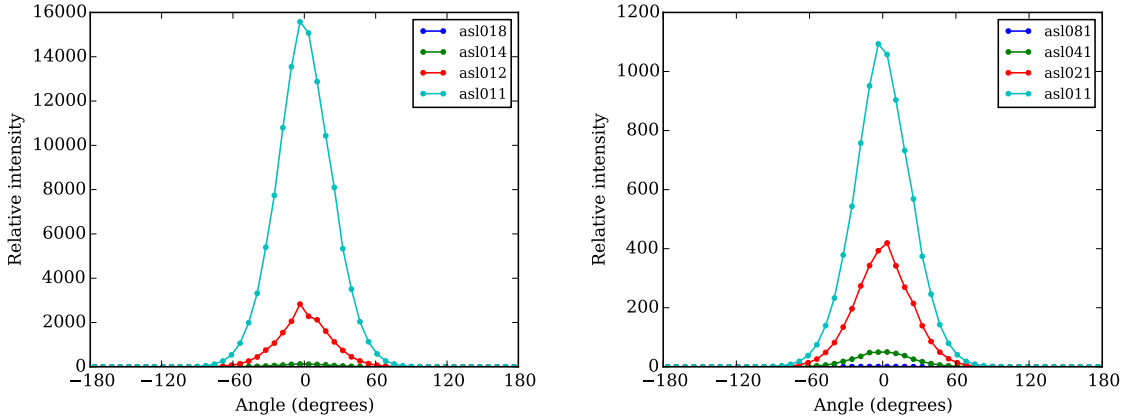
Figure 6.2: Three plots of BSFs whose path lengths are identical multiples of the scattering length. Each of these plots has the longer path length's peak intensity normalized to 1 and the other BSF is plotted with respect to that normalization factor. In each case, the BSF with the longer path length appears squashed by a factor on the order of $2R - 2$.

BSFs with equivalent Rb are plotted in Figure 6.2. However, it appears that the hypothesis is not valid. Instead, the BSFs with longer path length are squashed by a factor related to the path length. This factor disagrees with the factor of difference if we only vary path length or scattering coefficient (Figure 6.3).

This experiment exposed a flaw in the methodology. An earlier form of seed path generation was used to collect the data. It specified how to vary arc length completely in terms of the curve's parameter space, and did not control the arc length distribution. This led to the distribution clustering around R, having a maximum of about $1.5R$ (Figure 6.4), biasing the data set towards paths having arc length R. This means arc lengths in this data set are limited to 0 to 10 scattering lengths, and they are vastly undersampled at the higher scattering lengths. Having appropriate sampling at these larger scattering lengths is important for observing multiple scattering, and the trend present in Figure 6.2 could be only present in these paths of just a few scattering lengths.

6.1.1 Retrial

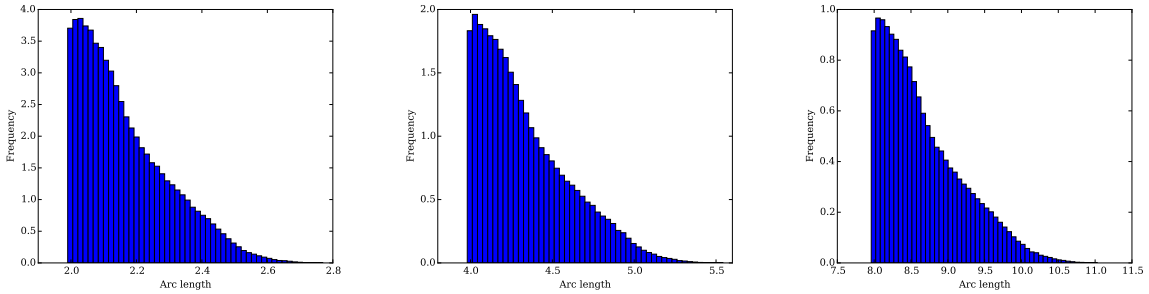
Because of the flaw discovered, a retrial of the $Rb = 2$ data set was conducted with two modifications. We finely control the arc length's distribution by forcing a uniform



(a) BSFs with varying path length

(b) BSFs with varying scattering coefficient

Figure 6.3: Beam spread functions with varying path length and scattering coefficient. The 8 scattering lengths data set is normalized, and each BSF is plotted using this normalization factor. There is a far greater decay in the intensity with respect to path length than the scattering coefficient, and it is suggests the trend seen in Figure 6.2 is not the same trend for path length seen here.



(a) $R = 2$ arc length distribution

(b) $R = 4$ arc length distribution

(c) $R = 8$ arc length distribution

Figure 6.4: Distribution of arc lengths generated for various path lengths. A flaw in an earlier form of the seed path generation method causes the arc lengths to cluster around R and to be limited in the maximum arc length.

Label	Scattering (b)	Path length (R)	Rb	s_{max}
asl012-uniform-theta	1	2	2	10
asl021-uniform-theta	2	1	2	5

Table 6.4: Parameters for scattering vs. path length retrial.

	Efficiency	Gaussian width	Sample count	Wall time
asl012-uniform-theta	62.11%	27.94	1.02e+07	27.18 years
asl021-uniform-theta	62.43%	28.36	1.03e+07	26.25 years

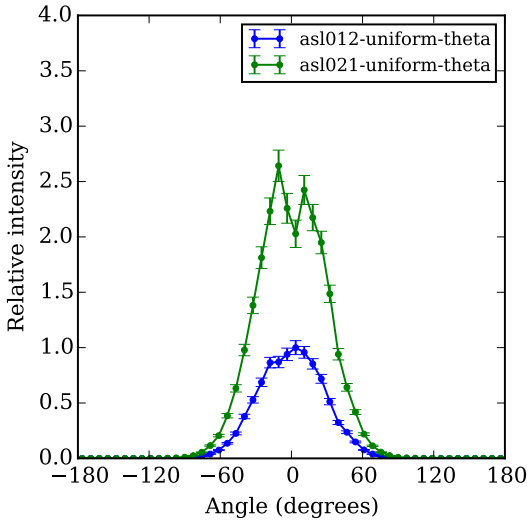
Table 6.5: A summary of collected retrial data. Roughly the same trends in efficiency are seen. Likely the increase in s_{max} has raised the Gaussian fit's width, but the possibility for bias introduced by the altered sample pattern casts doubt on that conclusion.

distribution on the interval $[0, s_{max}]$. Also, we sample uniformly over θ to combat variance with fewer samples. Each trial's s_{max} was set to exactly ten scattering lengths, so for the $b = 2$ trial we have $s_{max} = 5$ and for the $b = 1$ trial we have $s_{max} = 10$. A table of parameters is given in Table 6.4.

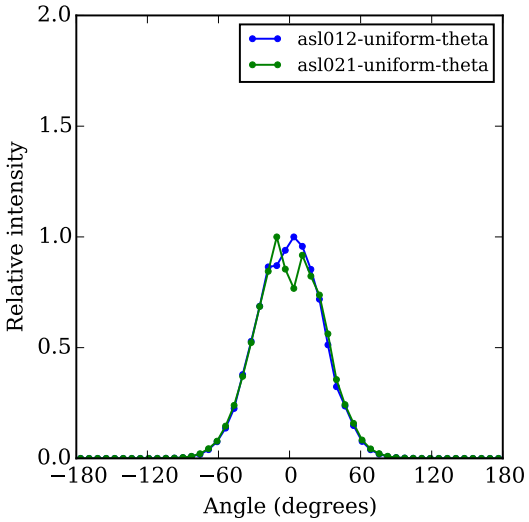
A summary of the data collected is available in Table 6.5. One interesting observation about this data set is that the fitted Gaussian width has increased fairly substantially as it is in Table 6.3. This is very likely due to the increase in s_{max}, as broadness in the BSF would be an effect of multiply scattered radiation. Multiply scattered radiation tends to have longer flight paths. However, because we also modified the sampling distribution in this trial, this observation can serve as a hypothesis for a later experiment.

The hypothesis that the BSFs should be identical is not viable in this case either, as seen in Figure 6.5a. The $R = 2$ data set continues to have a peak value different from the $R = 1$ data set. The relative behavior of the peaks of these two data sets are roughly the same as in the original experiment (see Figure 6.2a). Normalizing the two BSFs such the peaks are at unity shows that they are the same shape (Figure 6.5b).

We compare the repeated experiments to the original calculations in Figure 6.6. We observe a great difference in the peak values between the repeated and original experiments as evident in Figures 6.6a and 6.6c. The difference is likely due to the sample bias from



(a) $Rb = 2$ relative comparison



(b) $Rb = 2$ with peaks normalized to unity

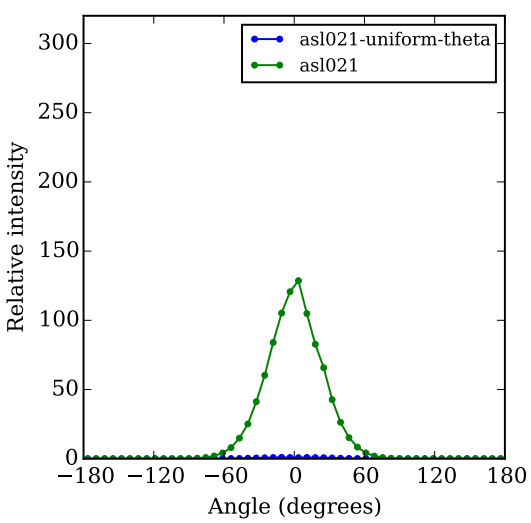
Figure 6.5: Varying Rb experiment retried with $Rb = 2$ under the revised methodology. The error bars shown are standard error of the mean. A difference in the peak value of the two BSFs is evident in (a). Comparing Figures 6.2a and 6.5a we see roughly the same relative behavior in the peaks of the BSFs. Normalizing the BSFs to unity (b) shows they are nearly the same width. This is also evidenced by the nearly identical Gaussian fit width parameters in Table 6.5.

the original experiment, which tended to select shorter arc lengths. These shorter arc lengths tends to contribute more than longer paths due to less total extinction. However, when normalizing the two BSFs (Figures 6.6b and 6.6d) we see that the repeated experiment calculated a wider BSF. This is likely due to a tendency to include longer arc lengths, which are more likely to contribute in the larger angles of the BSF due to multiple scattering. We also see this same behavior in the different Gaussian width values shown in Tables 6.3 and 6.5.

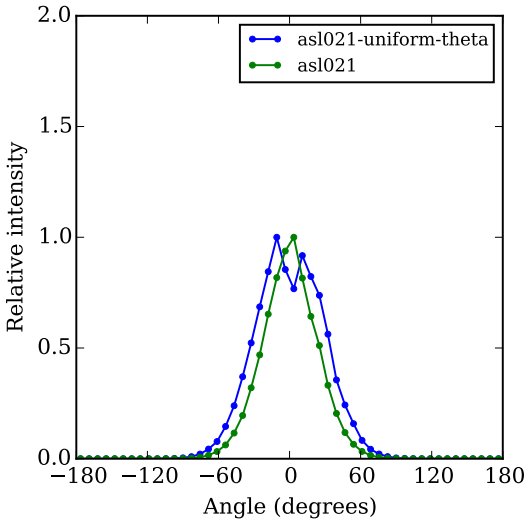
We next examine the effect of uniform sampling over θ in Figure 6.7. The most notable observation is that uniform sampling is not actually achieved in this case. The distribution sinks significantly at the extreme large and small angles (Figure 6.7b). Likely this is due to a tendency for samples to be aborted in these regions as it is difficult to generate seed paths in these cases. It is not expected that the non-uniform distribution causes problems in our case, as changing the sample distribution was only done as a convenience to deal with variance. However, a distribution closer to uniform might be achieved by retrying seed path generation several times before aborting.

6.1.2 Conclusion

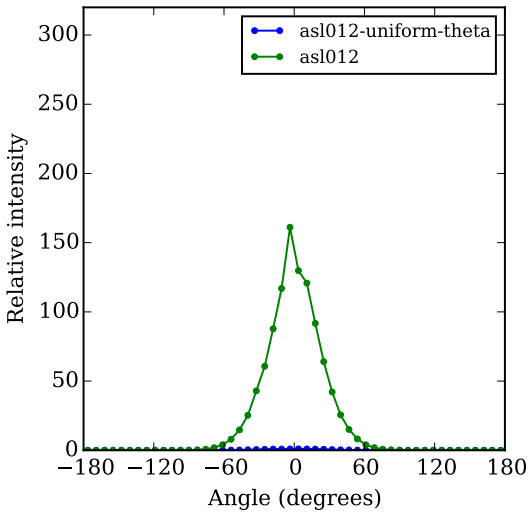
The hypothesis for this experiment was that for identical values of Rb the calculated BSF should be equivalent, a direct consequence which extends from the path segment weight function (see Figure 4.7 on page 40). All trials suggested that this hypothesis was false. Instead, we found that for BSFs with identical Rb, the BSF with the larger R tends to have a smaller peak value. Other observations from this experiment showed a bias in how the methodology as originally proposed selected path arc lengths. Originally, it was biased to produce paths with arc lengths roughly in the interval $[R, 1.5R]$ with a bias towards arc lengths of R. This observation led to a revision in the methodology which allows arc lengths to be arbitrarily distributed, provided $s > R$. A retrial of the experiment was conducted with the modification and the arc lengths distributed over the interval $[0, s_{max}]$ where s_{max} is ten scattering lengths. The retrial yielded conclusive results for the $Rb = 2$ case, and



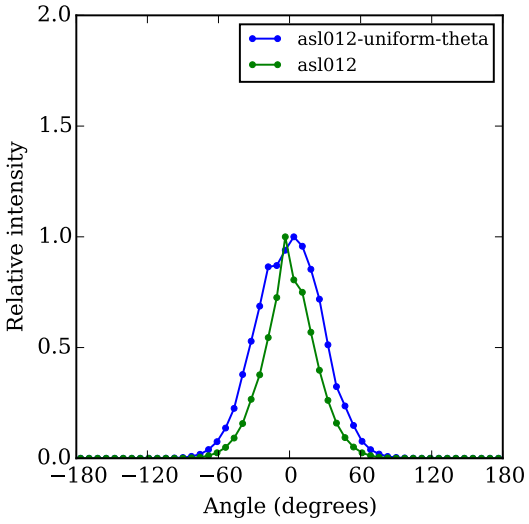
(a) $R = 1$ relative comparison



(b) $R = 1$ normalized to unity

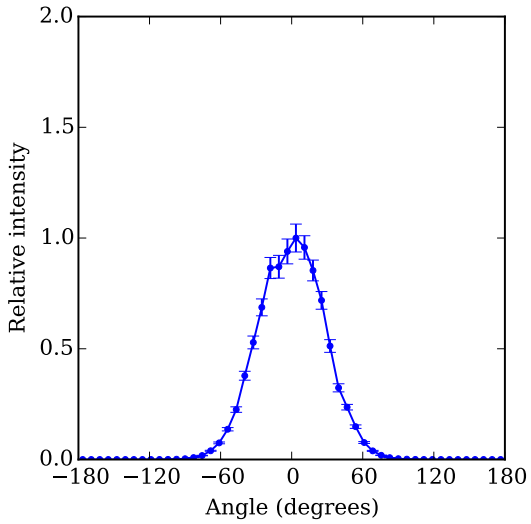


(c) $R = 2$ relative comparison

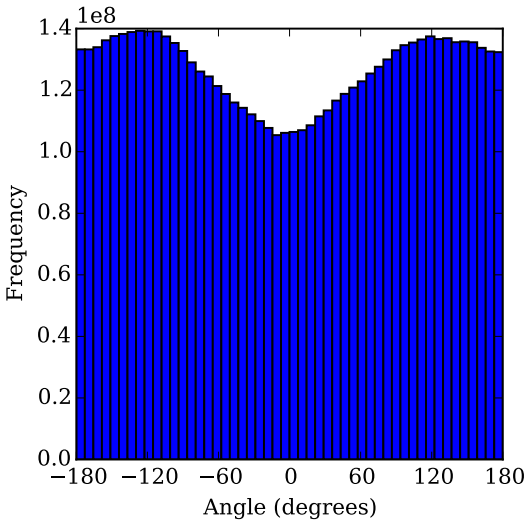


(d) $R = 2$ normalized to unity

Figure 6.6: Repeated and original scattering versus path length experiments compared. The peaks in the original experiments are much higher than in the repeated experiment, likely due to a bias towards shorter arc lengths. When normalized, we see repeated experiments calculate a broader BSF, but both still show Gaussian behavior with respect to θ.



(a) $R = 2$ with standard error of the mean error bars



(b) Sample pattern over θ

Figure 6.7: A view of the effect of uniform sampling over θ on the error of the calculated BSF. Error is reasonably kept under control especially within the small angles. However we quickly see true uniform sampling is not actually achieved, as there are far fewer samples at the smaller angles.

also suggested that s_{max} has an effect on the width of the BSF. This could be a useful relationship to understand in future work. Future work may also study the peak value of the BSF with respect to R, as this may suggest why the hypothesis was violated.

6.2 Comparison with experimental data

Showing equivalence of a calculation by the FPI approach to an experimental result would be a milestone for the validation of the numerical techniques used by the FPI approach. It marks the first time the numerical technique has been used to simulate a real phenomenon. There are inherent difficulties matching to experimental results. Optical parameters such as the absorption (a) and scattering (b) values are not known. As well the phase function is not known. To estimate the absorption and scattering coefficients the assumption is that the media is infinite and spatially uniform, so a and b are both constants. This is not true for sea ice as there are often anisotropies such as crystal alignments

Parameter	Value
Absorption coefficient (a)	0.004 cm^{-1}
Scattering coefficient (b)	0.1 cm^{-1}
Path length (R)	30 cm
Maximum arc length (s_{max})	100 cm
Gaussian width (μ)	0.5
Regularization constant (ϵ)	0.075

Table 6.6: Table of experiment parameters used to reconstruct Maffione beam spread function measurements. Note that s_{max} is set to about ten scattering lengths in this case. This follows from the product $b \times s_{max}$.

and fractures in the media. Such anisotropies do not have drastic effect on the measured BSF and can be remediated by measuring the BSF in the positive and negative directions and checking for symmetry [40]. The isotropic simplification allows us to estimate values for a and b more easily, but for the phase function there is far less information. Precise experimental measurement of phase functions is difficult [50]. A true measurement should isolate only a single scattering event, but for a highly scattering substance like sea ice this is very challenging. Nonetheless there have been attempts to measure the phase function and scattering coefficient of sea ice [14] which yielded a very forward-peaked phase function and scattering coefficients in the range of 0.089 cm^{-1} to 0.196 cm^{-1} dependent on the age and conditions of the ice. Since there is not general agreement on the phase function we choose a forward-peaked Gaussian phase function having width μ as an approximation. This is a reasonable approximation, but it penalizes backscattering which would be present to some degree in a real substance. A table of parameters used in the experiment is available in Table 6.6. Tabulated forms of the data collected in this experiment are available in Appendix B, and some additional analysis of the error present in each data set is in Appendix C along with Gaussian fits and fit residuals.

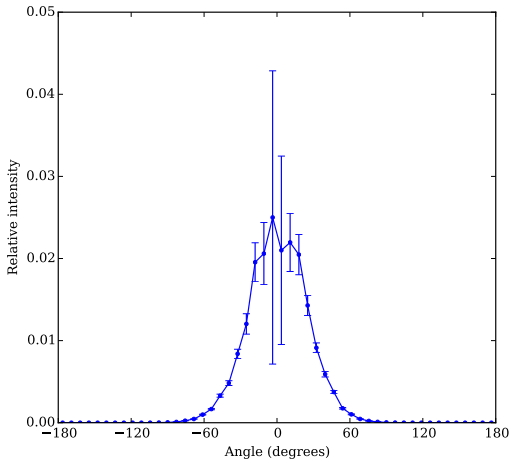
Using these parameters, we calculate BSFs using the FPI approach to compare to the experimental BSFs measured by Maffione et al. [40] (see Figure 6.8). There are shared characteristics between the calculated and experimental BSFs: the peak at 0 degrees and gradual decrease in the relative intensity in the mid-range angles. The gradual falloff

suggests some of the multiple scattering behavior is being captured. The calculated BSF falls off to about 0 by the 90 degree mark, whereas the experimental results still measure some amount of intensity at the extremes. There are several possible explanations for this. For example, the extreme high angles correspond to higher amounts of backscattering, and the phase function has a strong forward peak. It is also possible field experimental conditions led to external factors contributing to the data set. In the calculated BSF there is a large amount of variance at the smaller angles, likely due to undersampling evident in Figure 6.8c.

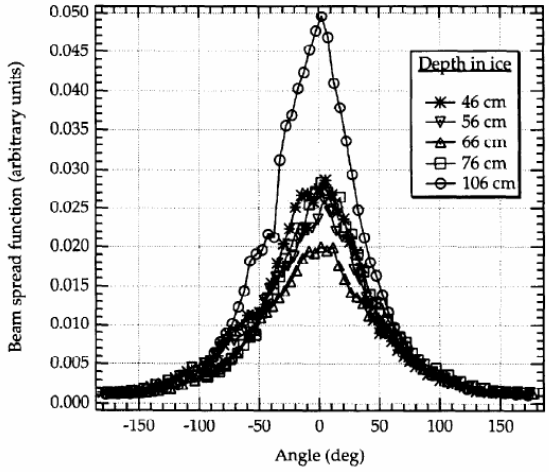
Maffione et al. [40] notes that the laboratory BSF measured by Schoonmaker et al. [54] regressed more easily to a Gaussian than their data set. We use least squares to determine parameters for both a Gaussian and a Lorentzian fit in Figure 6.9. It appears that the calculated BSFs follow a Gaussian much more closely. This means the numerical FPI approach disagrees with field measurements not just at angle extremes but in general shape as well. This may suggest our assumption of infinite, uniformly scattering media follows laboratory conditions more closely than field conditions if the data are to be trusted.

One parameter of interest is M, the number of path segments. Higher values of M correspond to a smaller Δs and therefore smaller minimum distance between scattering events. For highly scattering media like sea ice, there are many scattering events so it is beneficial to decrease Δs to capture this high number of scattering events. More scattering events corresponds to a broader BSF. Therefore, increasing M should broaden the calculated BSF. A larger M leads to more floating point multiplications during the calculation of the path weight. Currently this leads to floating point underflow. A small value of M leads to higher numbers of aborted samples, decreasing the efficiency. Facing these difficulties, we calculate BSFs for $M = 80$, $M = 120$, $M = 160$, and $M = 200$ using the parameters in Table 6.6 to observe the effect on the BSF. Each of the BSFs is plotted in Figure 6.10. A summary of the data collected is available in Table 6.7.

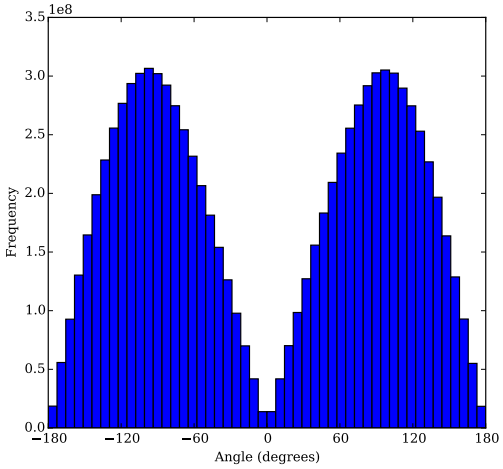
By visually inspecting the plots in Figure 6.10 we can immediately see there is some broadening in the calculated data, especially in the less noisy larger angles. As a metric



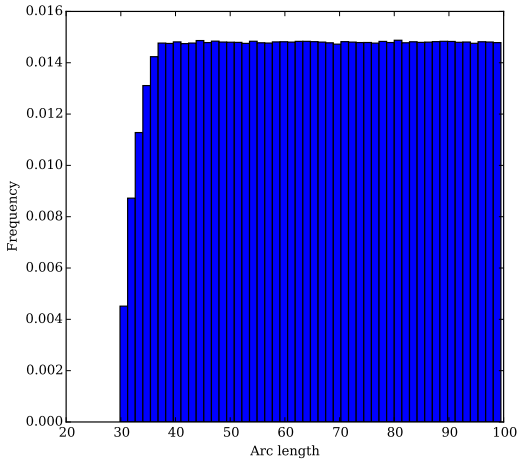
(a) Calculated BSF for $M = 200$.



(b) Experimental BSFs (Figure 5 from [40].)

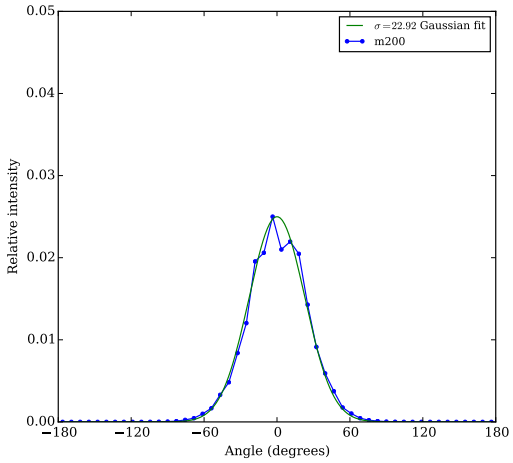


(c) Sample pattern for (a).

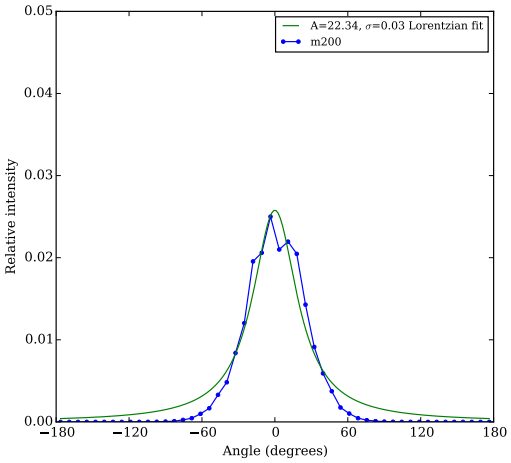


(d) Arc length distribution for (a).

Figure 6.8: A calculated BSF (a) using the numerical FPI approach, normalized such the peak value is 0.025. The sample pattern shown in (c) is the reason for the increased standard error of the mean in (a) at the smaller angles. In (b) we see similar trends as in (a) but especially at the 50 degree mark there is disagreement in the shape. The calculated BSF does not have the same gradual falloff of the experimental BSFs. In (d) we see a roughly uniform arc length distribution, except for perfectly straight paths. This could be due to continued numerical difficulty in constructing perfectly straight paths.



(a) Gaussian fit for $M = 200$ data set.



(b) Lorentzian fit for $M = 200$ data set.

Figure 6.9: Gaussian and Lorentzian fits for the $M = 200$ data set. Maffione et al. [40] found a Lorentzian was a better fit for their data, however the calculated BSF follows a Gaussian fit much more closely. They also note Schoonmaker et al. [54] fit a Gaussian to their laboratory BSF data. This may mean the calculated BSFs follow laboratory conditions more closely than field conditions, possibly due to assumptions of infinite, uniformly scattering media.

	Efficiency	Gaussian width	Sample count	Wall time
m80	53.92%	10.08	2.08e+07	76.72 years
m120	56.54%	14.64	2.16e+07	68.36 years
m160	58.97%	22.57	1.48e+07	48.58 years
m200	60.74%	22.92	1.50e+07	48.56 years

Table 6.7: Summary of collected BSF data. Note the drastic increase in computation time compared to the data in Section 6.1, which is to lessen the variance added by sampling paths of many more scattering lengths. The Gaussian width shows an increase with respect to M, and interestingly so does the efficiency. The efficiency increases because the perturbation algorithm is less likely to fail when there are more path segments to select.

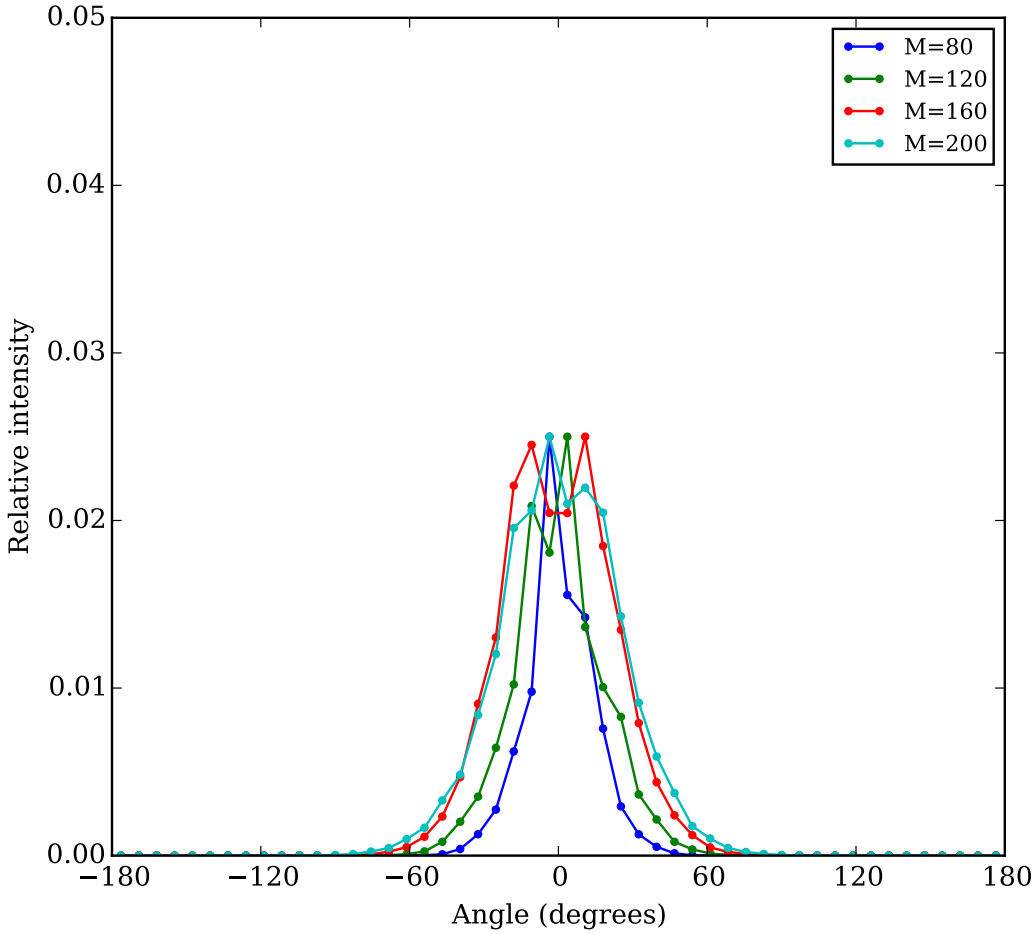


Figure 6.10: A plot of several BSFs with varying path segments (M). The static parameters used here are as in Table 6.6. There is a clear trend in the broadening of the BSF as M increases. This suggests a larger value of M can more accurately account for multiple scattering effects.

for the broadening, we can use the fitted Gaussian width σ which is available in Table 6.7. The Gaussian width also increases with M, and also seems to be approaching a value. We can use a step function such as

$$\sigma = A(M^p/(B + M^p)) \tag{6.3}$$

to predict what value σ converges to in the limit of M. This is the fit parameter A. We find $p = 4$ is a fairly decent step function to use for our calculated data. Fitting by least squares yields a value of $A = 23.76$, suggesting that the $M = 200$ data set is already a close approximation in the limit of M (Figure 6.11). The projection may not be reliable with a small data set like this one, or the step function could be inappropriate. In any case, more data is needed, particularly for large values of M, make accurate conclusions about decent values of M for this experiment.

In conclusion, we compare the FPI approach's calculated beam spread functions to experimental beam spread functions measured from sea ice [40] shows that the FPI approach's calculated beam spread function and the measured beam spread functions differ fundamentally in shape: the FPI approach follows a Gaussian shape while the experimental data follows a Lorentzian shape. However, [40] is a field measurement, and lab measurements of Gaussian shape have been reported which follow a Gaussian shape [54]. Beam spread functions for varying path segment counts show that the Gaussian width parameter σ appears to be approaching a shelf value in the limit of M. However, attempts to project that shelf value are currently inconclusive due to the amount of computer time required to calculate a single beam spread function, limiting the amount of data that can be collected with current approaches. The validity of the projection equation is not known either. Future extensions to this experiment might explore the use of larger M values than those used here.

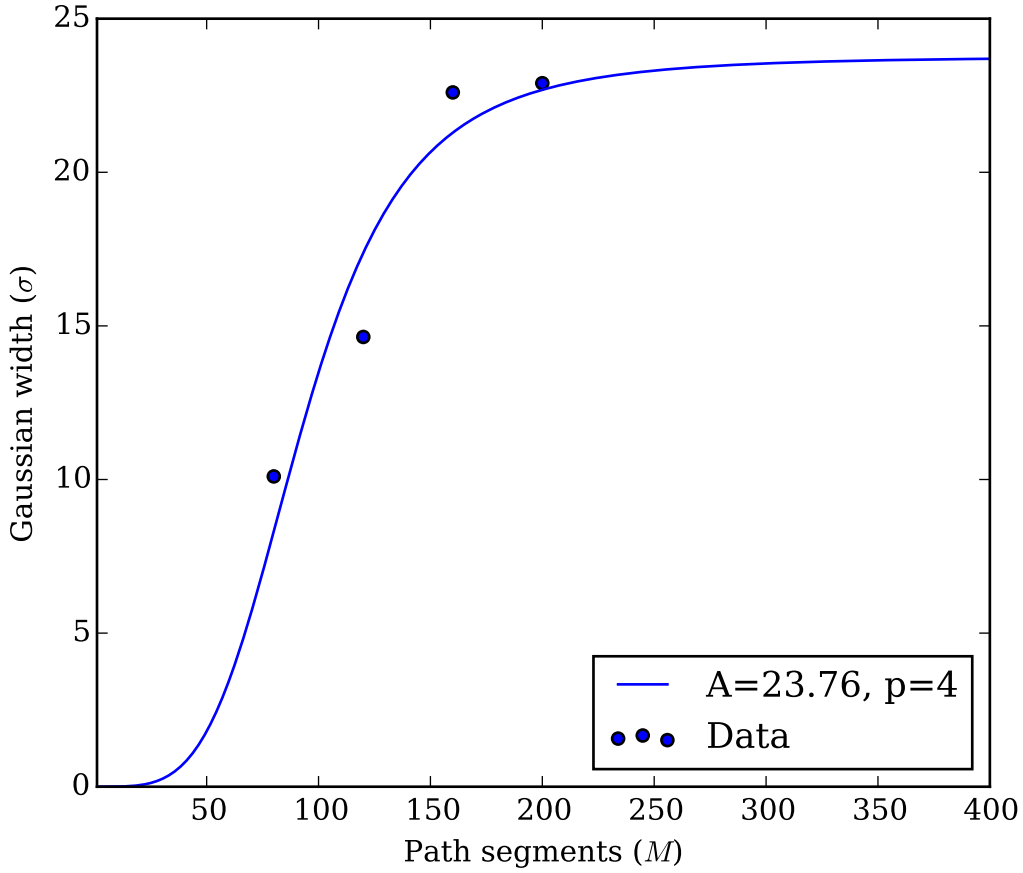


Figure 6.11: Projection of the Gaussian fit parameter σ for calculated BSFs with respect to the number of path segments M with a least squares fit of Equation (6.3), and a manual constraint of $p = 4$. The fit suggests there is not much broadening beyond what is calculated for the $M = 200$ data set.

6.3 Spherical symmetry

An advantage to computing a BSF in three dimensions is that we can check for spherical symmetry within the calculation. To aid in analysis we use a simple 2D projection for the data. We define two unit vectors $\hat{c}_{up}$ and $\hat{c}_{right}$, somewhat alike a camera facing the laser. We can define an $x-$ and $y-$coordinate like so:

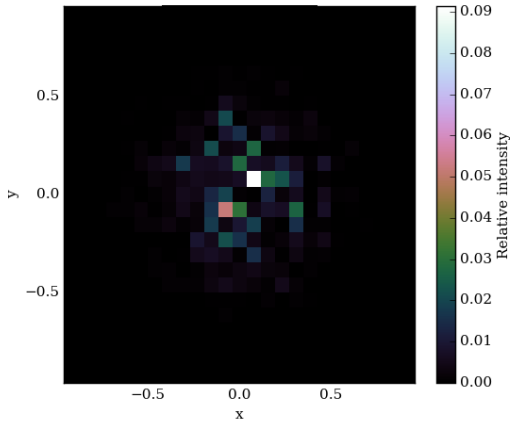
$$x = \hat{n}_1 \cdot \hat{c}_{right} \tag{6.4}$$

$$y = \hat{n}_1 \cdot \hat{c}_{up} \tag{6.5}$$

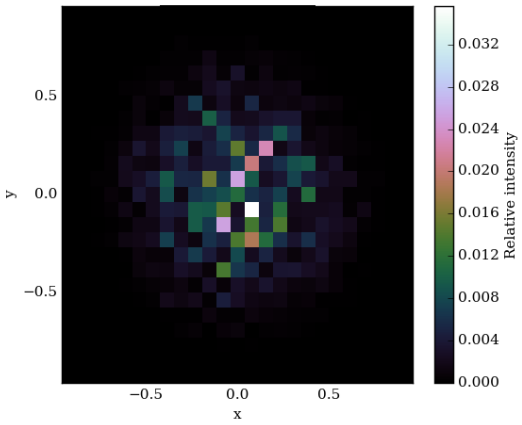
Using these coordinates, we can divide samples into bins and integrate as done for θ in previous analysis. In the case of the "varying-M" data sets from Section 6.2, we choose $\hat{c}_{up} = \hat{y}$ and $\hat{c}_{right} = -\hat{z}$, producing four 2D plots (Figure 6.12).

In producing such images we use a fairly fine bin count compared to analysis of just θ so variance is amplified. The plots show sufficient spherical symmetry which is the expectation, especially for $M = 200$ and $M = 160$. The other data sets produce a much more finely peaked BSF with generally higher variance. The variance dominates in these plots and spherical symmetry is not clear. However, earlier experiments suggest larger values of M are desirable for better accuracy, so spherical symmetry is not as important for these small values of M.

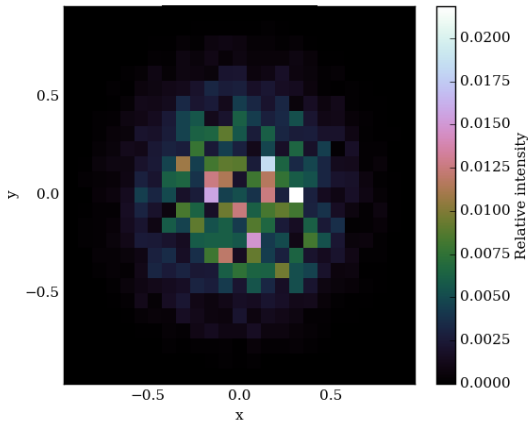
In conclusion, spherical symmetry is apparent at least for higher values of M. The effect of this conclusion is we can be reasonably certain the approach is robust in three dimensions and can possibly be used for other sensor geometries.



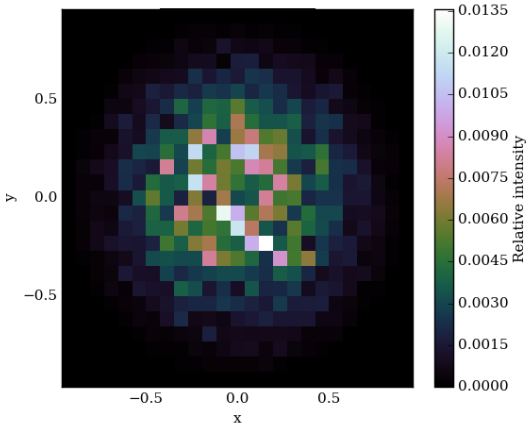
(a) 2D intensity plot of $M = 80$



(b) 2D intensity plot of $M = 120$



(c) 2D intensity plot of $M = 160$



(d) 2D intensity plot of $M = 200$

Figure 6.12: Normalized 2D intensity plots for each of the varying M data sets. The plots suggest spherical symmetry is present in the data, especially with $M = 160$ and $M = 200$. It is difficult to claim spherical symmetry for the smaller values of M, but generally smaller values of M should not be of much concern for accurate calculations.

Chapter 7

Conclusions

This chapter summarizes the main findings of the dissertation, and gives a more detailed summary of the dissertation contents. The guiding hypothesis for this research is that there exist numerical methods for solving the FPI approach to radiative transfer, and these methods can faithfully model multiple scattering effects in participating media. To summarize the entire dissertation, numerical methods were developed and tested in experimental setups meant to parallel experiments where beam spread functions are measured. The results disagreed with the chosen data set, however similar experiments where beam spread functions are measured have also disagreed.

7.1 Summary of contributions

The contributions of this dissertation lie in both methodology in experiment. In methodology, this dissertation presents a method for asymptotic speedup of a path perturbation algorithm originally introduced in 2009 [60] which is necessary for numerically integrating path integrals in the FPI method of radiative transfer. This adjustment in the methodology allows for larger scale experiments to be conducted with FPI radiative transfer. Also, the dissertation introduces new methods for integrating over the spatial integrals of Equation (2.16) to create paths which are eventually used for numerically computing

the path integral. This enables construction of experiments meant to reproduce real world phenomena like beam spread functions. The experimental contributions of this dissertation are a thorough analysis of the path integral calculation which proves repeatability and lack of bias with enough samples. The knowledge that the calculations are repeatable enables larger scale experiments involving the path integral calculation like beam spread functions. Finally, the last experimental contribution is a calculation of several different beam spread functions to explore the effect of certain parameters on these beam spread functions, compare against experimental measurements, and to check for robustness in three dimensions. Initial parameter studies showed a flaw in the proposed methodology and led to more careful control of the arc length of seed paths. A retrial with revised methodology yielded similar results, but also showed significant differences in the width of the calculated beam spread function with the change in s_{max}. Later calculations showed fundamental differences in shape between the measured and calculated beam spread functions, but the shape difference could be the difference of Schoonmaker et al.'s lab-measured beam spread functions [54], which fit the Gaussian shape like that calculated by FPI radiative transfer, and field-measured beam spread functions [40], which instead fit a Lorentzian function. It is important to note that Schoonmaker et al's experimental setup measures intensity across the sea ice interface, and not with respect to angle like Maffione et al. The difference in these two conditions could be due to a lack of backscattering, which the forward-peaked Gaussian phase function does not support, and Maffione et al. may have measured more backscattering than the young sea ice grown by Schoonmaker et al. Since the calculated beam spread function fits one of the suggested models of [40] there is promise that other experimental data sets could be reproduced. The experiment also revealed a tendency in M, the number of path segments, to affect the width of the calculated beam spread function. The fit saw diminishing returns for values of M close to 200. This discovery may inform future best practices for choosing an M which has the desired accuracy without uselessly slowing down the calculation. Finally, calculated beam spread functions showed spherical symmetry in several data sets when projected onto a plane. This improves confidence in

the FPI approach's robustness in three dimensions, however high variance continues to be a larger problem because higher resolution is generally required to make meaningful judgment of the results.

The contained contributions support the thesis that Feynman path integrals as a basis for radiative transfer shows promise in a numerical setting to model multiple scattering in participating media realistically. Though fundamental differences in shape were found in the calculated beam spread functions, other researchers have measured Gaussian-shaped beam spread functions [54] across the sea ice interface. It is numerically well-behaved, evidenced by repeatability of the calculation and spherical symmetry, and ready for further study. The current challenges are chiefly dealing with the enormous computation time, a lack of parameter studies, and a lack of reference data to compare with.

7.2 Summary of dissertation

In Chapter 1 we discuss the introduce the basic concepts of radiative transfer, particularly scattering, and discuss briefly the history of the FPI approach to radiative transfer. Chapter 2 gives background needed to understand the FPI approach, including the introduction of Feynman path integrals and their applications, the Frenet-Serret frame and a discretized curve used by the FPI approach, the basics of Monte Carlo integration in single and higher dimensions, and the mathematics behind the FPI approach itself. In Chapter 3 we introduce relevant related work including several approaches to solving global illumination in computer graphics, and the scientific applications of radiative transfer in medicine, nuclear engineering, ocean and atmospheric science, and geophysics. Chapter 4 gives a precise explanation of the currently best known methods for solving the FPI approach to radiative transfer, including creating seed paths, the perturbation of paths and its accelerated form, and calculating path weights. Chapter 5 outlines the methods and discoveries of some early work of numerically computing the path integral without the spatial and angular integrals, which finds that the perturbation numerical method is generally

well-behaved and reproducible. We continue on to integrate over the spatial and angular integrals as well in Chapter 6, where beam spread functions are calculated, compared to experimentally-measured beam spread functions, found to be Gaussian-shaped, and dependent on the number of path segments used. In addition, beam spread functions calculated for path lengths of identical scattering lengths were shown to differ, and exposed an error in the methodology as originally proposed. Beam spread functions are also shown to be spherically symmetric for higher numbers of path segments.

7.3 Future directions

Currently using the FPI approach to compute very simple cases like beam spread functions takes a significant amount of compute time. Left unchecked, it will be infeasible to test the approach in cases where the participating media is spatially varying or with more complex emission geometries, like area lights or volume lights. Therefore, methods to reduce the computation time should be a primary future direction.

Approximations are a possible strategy. This can take form as an assumption within the mathematics that makes computation easier, or adjusting an existing parameter. For example, the s_{max} parameter controls the maximum arc length for a seed path, and decreasing this parameter also decreases variance. A large s_{max} is necessary to allow for multiple scattering, but eventually there is a diminishing return. Identifying this limit in a specific case like beam spread functions is useful, and may inform the general case too. As well, approximations can target the path integral itself, as it is done via stationary phase and steepest descents in [61].

Importance sampling can reduce variance for the general Monte Carlo application. Applying it to FPI radiative transfer has not been studied in depth. Importance sampling conceptually recommends sampling values which are more likely to contribute to the approximation of a mean by selecting probability distributions which encourage these "important" values. An example of how this could apply is the selection of arc length s. An

assumption in the current methodology is that the distribution of s should be uniform on the interval $[\|\vec{x}_1 - \vec{x}_0\|, s_{max}]$. However, paths of length close to s_{max} will not contribute as much as the shorter paths, so it may yield less variance to use a distribution other than the uniform distribution – for example, the exponential distribution. Of course, this introduces bias which eventually must be accounted for in the calculation. Other strategies should be explored to prevent undersampling. For example, the effect of sampling uniformly in θ could be investigated for beam spread functions, which would help contend with the variance in some of the otherwise undersampled area.

In a similar vein, use of the Metropolis algorithm could be investigated for variance reduction purposes. This is explored partially in Chapter 5 and preliminary methodology is proposed. However, it is not explored on larger scale experiments such as beam spread function computation. The Metropolis algorithm's effect on bias in the solution could be compared with the data collected in this dissertation.

A number of other modifications to the methodology could be tested. For instance, part of the methodology requires an arc length parameterization once a seed path is generated. However, the solution is error-prone and expensive. Other arc length parameterization algorithms could be investigated and evaluated, or the elimination of the arc length parameterization step altogether could be tested, as the error from finding an approximate arc length parameterization is eventually corrected for by the perturbation algorithm. Root-finding makes up a large portion of the work done during a job, and there is not much investigation in how many of its parameters related to accuracy affect error and performance. For instance, most root finding methods have controllable tolerance values or iteration counts. Relaxing the tolerance and lowering iteration count can result in performance benefits sacrificing some accuracy of the solution – but how much error is acceptable to justify the performance benefits?

Monte Carlo solutions such as these are quite amenable to parallel architectures. Because of the low cost and minimal effort to adopt, the bulk of the work contained in this dissertation was either conducted on local campus computing resources or the Open Science

Grid [51]. Both of these resources are shared and are primarily oriented to computation on the CPU. Shared resources have the natural problem of policy preventing total use of the resources. Given the large scale computational resources required for this solution, CPU computation is not practical or cost-effective in the long term. GPGPU architectures have increased in prevalence and popularity, and their cost per unit of throughput makes them an attractive option. A well-written GPU implementation could drastically reduce the turnaround time for experiments, which is about two weeks for a CPU solution on Open Science Grid. This would in turn allow experiments to be iterated upon faster.

Moving away from performance-related concerns, there are many other possible parameter studies we could choose to conduct. For example, the effect on path segment count M on beam spread function calculation was studied in Chapter 6. Using s_{max} as a means to eliminate classes of paths was mentioned earlier, but its effect on beam spread function calculation could be studied. Chapter 6 attempts a projection for large values of M, but there are current precision issues which should be resolved so beam spread functions can be calculated for larger values of M. The results of the experiment in Section 6.1 were not according to expectation, and the results also suggested a relation between the s_{max} parameter and the width of the beam spread function. Either of these two topics could be investigated further. The effect of s_{max} on the peak of the beam spread function could be investigated. For example, there appeared to be an inverse relationship between s_{max} and the peak of the beam spread function, which could explain why the results of the experiment were not to expectation.

A central concern of radiative transfer with FPIs is physical correctness. At this point, all analysis conducted with collected data has not been using physical quantities. Instead, the values are scaled. This is out of necessity since experimental data is only available in relative units [40], and there are very few beam spread function data sets available which are not scaled. Currently, there are a few assumptions which could affect physical correctness. In Section 2.3 we reference a term D that arises from Monte Carlo integration of the path integral in Equation (2.15) under the assumption the perturbation

algorithm produces uniformly-distributed paths. However, very little investigation into the actual distribution in path space or path parameter space has been conducted, and given that the perturbation algorithm only produces related paths we should not expect that this distribution is uniform in most senses. The distribution of generated paths for both the seed path generation algorithm (Section 4.2) and perturbation algorithm (Section 4.3) should be investigated and characterized. In Section 4.4 we discuss a normalization procedure for ω_n, the path segment weight, that is for numerical ease, and the effect of this on physical quantities is not known.

Appendices

Appendix A

Derivation of a path segment's weight

The following presents the derivation of Equation (4.40) from Equation (4.39), which is rewritten here for convenience:

$$\omega_n = \exp(-c_n \Delta s) \int \frac{d^3\mathbf{p}}{(2\pi)^3} \exp(i\mathbf{p} \cdot \hat{\mathbf{N}}_n \kappa_n \Delta s + b_n \Delta s \tilde{Z}(|\mathbf{p}|)).$$

We transform the equation into spherical coordinates. We replace the differential as usual,

$$\omega_n = \exp(-c_n \Delta s) \int_0^{2\pi} d\phi \int_0^{\infty} \int_0^{\pi} \frac{p^2 \sin\theta}{(2\pi)^3} \exp(i\mathbf{p} \cdot \hat{\mathbf{N}}_n \kappa_n \Delta s + b_n \Delta s \tilde{Z}(p))\, d\theta\, dp, \quad \text{(A.1)}$$

and integrate the $d\phi$ integral:

$$\omega_n = \exp(-c_n \Delta s) \int_0^{\infty} \int_0^{\pi} \frac{p^2 \sin\theta}{(2\pi)^2} \exp(i\mathbf{p} \cdot \hat{\mathbf{N}}_n \kappa_n \Delta s + b_n \Delta s \tilde{Z}(p))\, d\theta\, dp. \quad \text{(A.2)}$$

Next we take $\mathbf{p} \cdot \hat{\mathbf{N}}_n = p\cos\theta$, split the exponential into a product, and rearrange terms to

get

$$\omega_n = \exp(-c_n \Delta s) \int_0^\infty \frac{p^2}{(2\pi)^2} \exp(b_n \Delta s \tilde{Z}(p)) \int_0^\pi \sin\theta \exp(ip \cos\theta \kappa_n \Delta s)\, d\theta\, dp. \tag{A.3}$$

Now the $d\theta$ integral may be calculated,

$$\omega_n = \exp(-c_n \Delta s) \int_0^\infty \frac{p^2}{(2\pi)^2} \exp(b_n \Delta s \tilde{Z}(p)) \frac{2\sin(p\kappa_n \Delta s)}{p\kappa_n \Delta s}\, dp, \tag{A.4}$$

and the final terms cancelled to yield

$$\omega_n = \exp(-c_n \Delta s) \int_0^\infty \frac{p}{2\pi^2} \exp(b_n \Delta s \tilde{Z}(p)) \frac{\sin(p\kappa_n \Delta s)}{\kappa_n \Delta s}\, dp$$

which is Equation (4.40).

Appendix B

Tabulated beam spread function data

This appendix contains the raw tabulated data collected for the Maffione beam spread function replication experiment, for path segment counts $M = \{80, 120, 160, 200\}$. Each table is presented as a function of angle (written θ in Chapter 6). We present the mean intensity and standard deviation (written "s.d." in the table) which is the raw value of the transport kernel after Monte Carlo integration. The mean wall time of computation for each θ value is also given in seconds. The path count given is the number of paths generated for the θ value. Note that each sample generates 1000 paths, and therefore the number of successful samples is path count divided by 1000.

Angle (degrees)	Mean intensity	Intensity s.d.	Mean time (s)	Time s.d.	Path count
-176.4	2.18e-131	1.97e-126	116.97	114.79	21668000
-169.2	1.44e-130	1.88e-125	116.84	115.26	64666000
-162.0	6.51e-129	1.51e-123	117.46	114.81	107760000
-154.8	3.11e-127	6.26e-122	118.48	114.32	149282000
-147.6	5.87e-127	1.33e-121	119.94	113.97	191417000
-140.4	6.08e-125	1.33e-119	121.02	113.41	230435000
-133.2	1.07e-124	1.99e-119	122.43	113.06	267803000
-126.0	8.42e-124	1.37e-118	123.28	112.59	300448000
-118.8	7.15e-123	1.09e-117	124.03	112.47	328342000
-111.6	3.65e-121	9.66e-116	124.51	112.34	350548000
-104.4	2.23e-120	2.88e-115	123.97	112.48	364802000
-97.2	1.87e-119	2.31e-114	122.44	111.75	369427000
-90.0	1.30e-118	1.61e-113	120.62	112.88	366565000
-82.8	1.51e-117	1.46e-112	118.39	112.07	357377000
-75.6	8.40e-117	6.91e-112	116.19	112.56	341153000
-68.4	4.70e-116	3.84e-111	113.78	112.57	319344000
-61.2	2.65e-115	2.11e-110	111.61	112.69	294909000
-54.0	7.61e-115	5.27e-110	109.18	112.54	265841000
-46.8	2.40e-114	1.30e-109	107.59	112.53	235413000
-39.6	1.20e-113	6.54e-109	105.56	112.39	202458000
-32.4	3.89e-113	1.95e-108	104.07	112.28	166809000
-25.2	8.40e-113	3.86e-108	102.44	112.20	130381000
-18.0	1.90e-112	9.81e-108	101.00	111.86	93349000
-10.8	2.99e-112	1.12e-107	100.42	112.56	56011000
-3.6	7.64e-112	3.36e-107	99.85	112.42	18651000
3.6	4.75e-112	3.21e-107	100.35	112.03	18858000
10.8	4.35e-112	1.77e-107	99.66	112.06	55586000
18.0	2.32e-112	8.59e-108	101.02	111.81	93652000
25.2	8.97e-113	4.36e-108	102.42	111.92	130932000
32.4	3.88e-113	2.36e-108	104.26	111.78	167956000
39.6	1.61e-113	8.94e-109	105.95	111.54	204878000
46.8	4.21e-114	2.44e-109	107.60	111.70	237664000
54.0	6.52e-115	3.89e-110	109.66	111.39	269911000
61.2	2.21e-115	1.70e-110	111.85	111.70	297718000
68.4	2.97e-116	3.92e-111	114.01	111.88	321550000
75.6	8.66e-117	5.97e-112	115.96	111.64	342945000
82.8	9.42e-118	8.04e-113	118.54	111.92	357846000
90.0	1.62e-118	1.78e-113	120.81	111.77	368180000
97.2	1.40e-119	1.50e-114	122.72	112.03	370114000
104.4	1.02e-120	1.03e-115	123.72	112.69	361212000
111.6	3.50e-121	8.03e-116	123.85	113.14	347108000
118.8	1.15e-122	1.62e-117	123.82	113.75	324454000
126.0	2.28e-123	4.41e-118	123.07	113.83	297678000
133.2	1.16e-124	3.87e-119	122.47	114.31	265132000
140.4	1.63e-125	4.24e-120	121.13	114.33	229036000
147.6	6.35e-126	2.01e-120	120.25	114.56	190466000
154.8	4.27e-128	8.22e-123	119.05	114.96	149476000
162.0	3.13e-129	5.50e-124	117.90	115.05	107377000
169.2	1.15e-130	1.91e-125	116.75	115.11	64317000
176.4	1.09e-135	1.58e-127	116.35	115.09	21257000

Table B.1: Computed beam spread function data for $M = 80$.

Angle (degrees)	Mean intensity	Intensity s.d.	Mean time (s)	Time s.d.	Path count
-176.4	2.02e-177	1.54e-172	101.74	90.88	24472000
-169.2	1.19e-175	2.35e-170	102.23	97.53	72850000
-162.0	1.41e-175	1.51e-170	103.05	96.57	121162000
-154.8	1.37e-174	1.65e-169	104.28	94.33	169231000
-147.6	9.16e-174	1.20e-168	105.24	94.77	215383000
-140.4	3.48e-173	4.65e-168	106.09	93.80	260251000
-133.2	3.91e-172	5.71e-167	107.21	92.61	301604000
-126.0	2.02e-171	2.94e-166	107.46	92.20	337632000
-118.8	8.71e-171	7.73e-166	107.91	93.36	366339000
-111.6	5.39e-170	8.21e-165	107.70	91.14	388809000
-104.4	2.42e-169	1.76e-164	106.85	92.68	402293000
-97.2	1.04e-168	7.02e-164	105.66	91.91	407688000
-90.0	5.26e-168	3.04e-163	103.71	92.19	403581000
-82.8	2.48e-167	1.23e-162	101.00	92.61	388173000
-75.6	8.46e-167	5.15e-162	98.28	93.11	368776000
-68.4	2.05e-166	9.18e-162	95.95	92.09	344177000
-61.2	6.39e-166	2.27e-161	93.78	95.02	315525000
-54.0	1.83e-165	6.66e-161	90.90	92.81	281849000
-46.8	6.02e-165	2.08e-160	89.20	93.75	249149000
-39.6	1.47e-164	5.04e-160	87.23	92.19	213244000
-32.4	2.56e-164	8.18e-160	85.90	95.45	175073000
-25.2	4.69e-164	1.53e-159	84.48	94.27	137094000
-18.0	7.44e-164	2.53e-159	83.62	96.54	97426000
-10.8	1.52e-163	4.69e-159	82.05	90.23	58116000
-3.6	1.32e-163	4.62e-159	82.86	98.82	19774000
3.6	1.82e-163	4.34e-159	81.91	90.13	19602000
10.8	9.94e-164	2.85e-159	82.24	91.67	58416000
18.0	7.33e-164	2.23e-159	83.08	94.47	97676000
25.2	6.03e-164	1.79e-159	84.32	92.58	136776000
32.4	2.66e-164	9.92e-160	85.52	92.57	175687000
39.6	1.57e-164	5.68e-160	87.37	93.03	214646000
46.8	6.01e-165	2.07e-160	89.29	92.98	251683000
54.0	2.70e-165	1.05e-160	91.25	92.07	287277000
61.2	9.46e-166	4.11e-161	93.80	91.43	318289000
68.4	2.67e-166	1.10e-161	96.02	90.96	347607000
75.6	8.61e-167	4.29e-162	98.19	91.89	370690000
82.8	2.39e-167	1.54e-162	100.98	93.10	390694000
90.0	5.75e-168	3.04e-163	103.69	94.24	403967000
97.2	1.09e-168	7.62e-164	105.25	91.21	406363000
104.4	2.10e-169	2.03e-164	106.71	90.56	400695000
111.6	5.10e-170	5.15e-165	107.79	94.44	385845000
118.8	6.98e-171	6.80e-166	107.87	93.04	362100000
126.0	1.09e-171	1.12e-166	106.99	94.40	331721000
133.2	1.01e-172	1.11e-167	107.11	93.17	297899000
140.4	2.28e-173	2.61e-168	106.04	93.07	257731000
147.6	1.08e-173	2.11e-168	105.65	95.54	214801000
154.8	6.47e-175	1.05e-169	104.81	95.72	168383000
162.0	2.98e-175	4.26e-170	103.34	95.39	120678000
169.2	1.06e-176	1.03e-171	102.79	95.81	72868000
176.4	1.32e-176	1.35e-171	102.13	91.82	24546000

Table B.2: Computed beam spread function data for $M = 120$.

Angle (degrees)	Mean intensity	Intensity s.d.	Mean time (s)	Time s.d.	Path count
-176.4	5.79e-226	2.90e-221	107.49	90.92	17828000
-169.2	1.24e-224	1.67e-219	107.89	90.75	52682000
-162.0	1.17e-224	8.27e-220	108.56	92.87	88050000
-154.8	6.55e-224	5.08e-219	109.46	92.16	122718000
-147.6	3.99e-223	5.22e-218	110.59	90.91	156795000
-140.4	1.67e-222	1.33e-217	110.80	90.02	187077000
-133.2	6.88e-222	5.99e-217	111.97	88.64	217716000
-126.0	2.60e-221	2.29e-216	112.08	88.53	243914000
-118.8	9.91e-221	7.32e-216	112.33	89.09	263035000
-111.6	3.11e-220	1.42e-215	112.21	88.80	279603000
-104.4	1.54e-219	7.32e-215	111.13	88.43	289448000
-97.2	4.16e-219	1.77e-214	109.82	90.00	292936000
-90.0	1.81e-218	7.25e-214	107.71	89.95	289492000
-82.8	4.53e-218	1.50e-213	104.69	89.40	277747000
-75.6	1.18e-217	3.82e-213	101.76	90.35	262297000
-68.4	3.42e-217	1.04e-212	99.01	90.68	244605000
-61.2	7.50e-217	2.01e-212	96.49	91.84	222949000
-54.0	1.66e-216	4.21e-212	93.45	90.70	199805000
-46.8	3.44e-216	8.96e-212	91.46	91.80	175479000
-39.6	6.91e-216	1.71e-211	89.32	90.30	148779000
-32.4	1.34e-215	3.16e-211	87.59	90.80	122851000
-25.2	1.92e-215	4.32e-211	86.57	91.05	96267000
-18.0	3.26e-215	7.34e-211	84.82	91.15	67956000
-10.8	3.62e-215	8.13e-211	83.69	91.66	40561000
-3.6	3.02e-215	6.65e-211	82.76	91.02	13488000
3.6	3.02e-215	5.94e-211	83.39	90.71	13415000
10.8	3.69e-215	9.76e-211	83.77	90.25	40914000
18.0	2.73e-215	5.89e-211	85.04	90.20	68685000
25.2	1.99e-215	4.55e-211	86.20	91.06	95820000
32.4	1.17e-215	2.42e-211	87.54	90.43	123242000
39.6	6.47e-216	1.66e-211	89.05	90.07	150651000
46.8	3.55e-216	8.16e-212	91.16	89.71	177414000
54.0	1.81e-216	5.25e-212	93.39	89.15	202499000
61.2	7.22e-217	1.88e-212	96.41	90.28	225022000
68.4	3.28e-217	9.53e-213	98.97	90.25	246302000
75.6	1.36e-217	4.22e-213	101.48	89.85	264211000
82.8	5.10e-218	1.66e-213	104.39	89.54	278519000
90.0	1.62e-218	6.34e-214	107.49	90.62	287927000
97.2	4.16e-219	1.77e-214	109.88	89.65	291975000
104.4	1.07e-219	4.94e-215	111.23	89.20	288125000
111.6	2.70e-220	1.35e-215	112.31	91.45	277406000
118.8	8.20e-221	5.93e-216	112.53	89.85	261332000
126.0	2.08e-221	1.91e-216	111.97	90.11	240511000
133.2	3.41e-222	2.43e-217	111.95	90.42	215911000
140.4	2.32e-222	2.21e-217	111.56	91.46	186929000
147.6	4.00e-223	5.89e-218	110.72	91.99	155333000
154.8	2.18e-224	2.59e-219	109.56	90.74	122157000
162.0	2.37e-224	1.89e-219	108.55	91.10	87648000
169.2	6.74e-225	4.24e-220	107.62	90.66	52775000
176.4	7.07e-226	6.13e-221	106.20	90.59	17651000

Table B.3: Computed beam spread function data for $M = 160$.

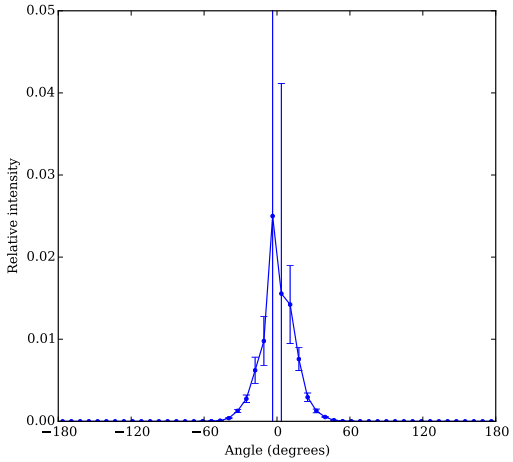
Angle (degrees)	Mean intensity	Intensity s.d.	Mean time (s)	Time s.d.	Path count
-176.4	3.92e-275	2.48e-270	107.55	83.95	18631000
-169.2	8.88e-275	5.18e-270	107.38	83.85	55816000
-162.0	3.29e-274	2.83e-269	107.68	83.84	92737000
-154.8	6.41e-274	3.74e-269	109.25	83.50	130324000
-147.6	1.47e-273	9.43e-269	109.50	82.88	164555000
-140.4	5.36e-273	2.64e-268	110.46	82.84	198825000
-133.2	1.64e-272	1.18e-267	110.74	81.73	228454000
-126.0	6.15e-272	2.64e-267	110.78	81.15	255655000
-118.8	2.01e-271	7.77e-267	111.34	81.77	276743000
-111.6	7.31e-271	3.03e-266	110.87	81.81	293640000
-104.4	1.99e-270	5.76e-266	109.59	82.27	302293000
-97.2	6.79e-270	1.98e-265	108.24	81.86	306578000
-90.0	1.71e-269	5.24e-265	106.08	83.11	302069000
-82.8	3.87e-269	8.98e-265	103.23	83.15	292286000
-75.6	1.02e-268	2.49e-264	100.13	83.92	274705000
-68.4	1.90e-268	4.10e-264	97.47	84.52	254224000
-61.2	4.13e-268	8.69e-264	94.68	85.00	231678000
-54.0	6.97e-268	1.34e-263	91.36	84.42	206580000
-46.8	1.38e-267	2.67e-263	89.37	84.91	181418000
-39.6	2.02e-267	3.64e-263	87.18	84.47	153987000
-32.4	3.52e-267	5.89e-263	85.60	85.23	126278000
-25.2	5.04e-267	1.04e-262	84.15	84.82	97834000
-18.0	8.19e-267	1.45e-262	82.79	84.90	69988000
-10.8	8.63e-267	1.40e-262	82.16	84.74	41875000
-3.6	1.05e-266	2.22e-262	80.88	84.27	13912000
3.6	8.80e-267	1.43e-262	81.26	85.19	13920000
10.8	9.20e-267	1.31e-262	81.68	84.79	41865000
18.0	8.58e-267	1.50e-262	82.80	84.32	70175000
25.2	5.99e-267	1.04e-262	83.95	85.04	98451000
32.4	3.83e-267	6.29e-263	85.56	84.93	127192000
39.6	2.48e-267	4.29e-263	87.02	83.73	155872000
46.8	1.57e-267	2.85e-263	89.18	83.40	183242000
54.0	7.36e-268	1.43e-263	91.62	83.34	209357000
61.2	4.27e-268	8.75e-264	94.45	83.64	234305000
68.4	1.91e-268	4.34e-264	97.11	83.58	255567000
75.6	9.06e-269	2.08e-264	99.72	83.15	275322000
82.8	4.28e-269	1.01e-264	102.96	83.47	291766000
90.0	1.51e-269	3.60e-265	106.06	82.35	302749000
97.2	5.91e-270	1.60e-265	108.07	82.34	305156000
104.4	2.35e-270	8.38e-266	109.82	82.22	302511000
111.6	6.95e-271	3.00e-266	110.63	82.79	289730000
118.8	2.11e-271	7.93e-267	111.44	83.06	274625000
126.0	6.22e-272	2.89e-267	110.89	82.72	253000000
133.2	1.74e-272	1.27e-267	110.80	84.11	226861000
140.4	3.91e-273	2.05e-268	110.66	83.53	196713000
147.6	1.86e-273	1.87e-268	110.09	83.33	163766000
154.8	6.57e-274	4.66e-269	109.10	84.60	128758000
162.0	1.01e-274	7.18e-270	108.32	84.12	92875000
169.2	4.56e-275	3.23e-270	106.88	85.62	55059000
176.4	3.47e-275	2.58e-270	106.04	83.21	18463000

Table B.4: Computed beam spread function data for $M = 200$.

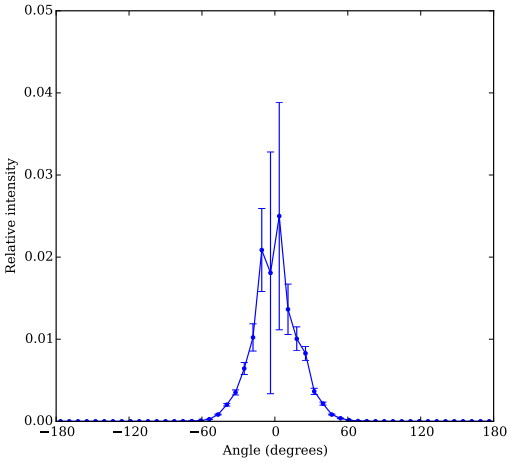
Appendix C

Beam spread function data analysis

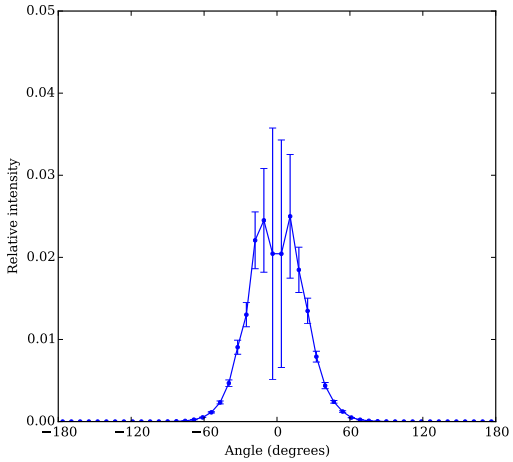
This appendix contains supplementary analysis of the data in Appendix B, speaking toward the error contained in each data set along with error present in the fit models. Figure C.1 contains each of the four "varying-M" data sets plotted as a function of angle with standard error of the mean error bars. Figures C.2 and C.3 plot the same data with Gaussian fits and the fit residuals.



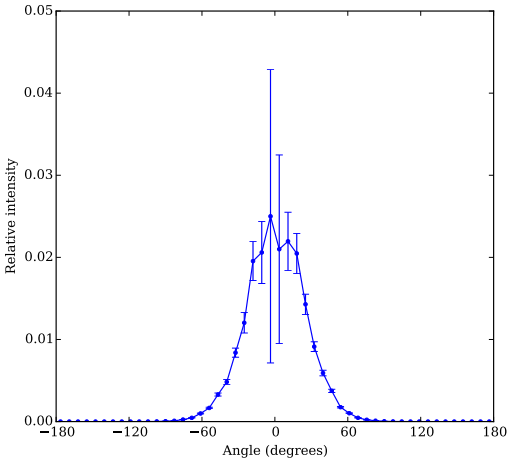
(a) $M = 80$



(b) $M = 120$

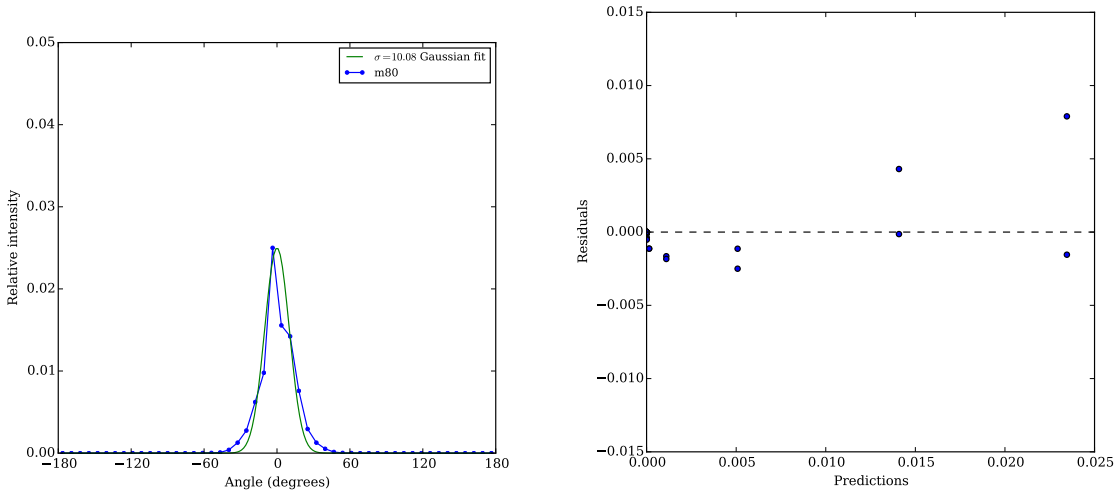


(c) $M = 160$

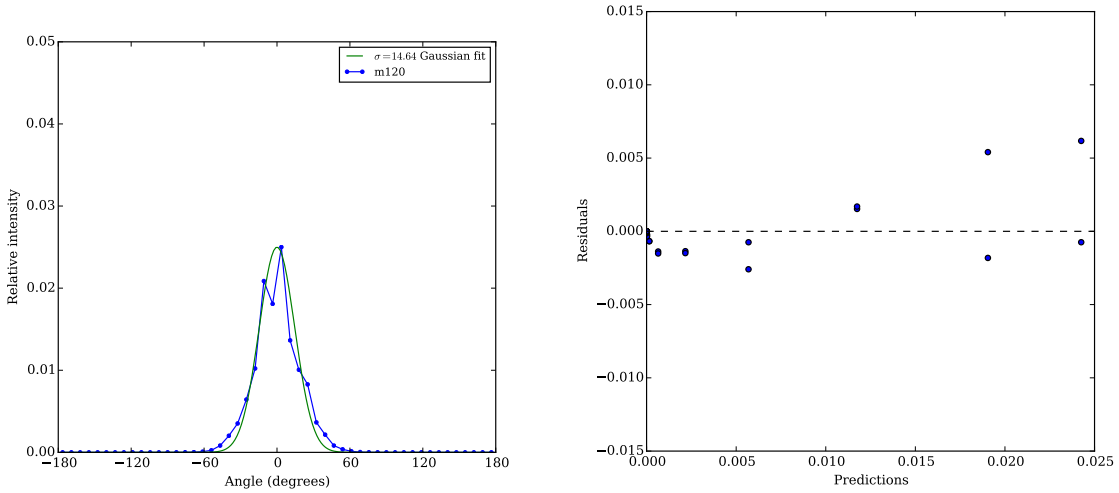


(d) $M = 200$

Figure C.1: Calculated beam spread functions with varying M and standard error of the mean used for error bars.

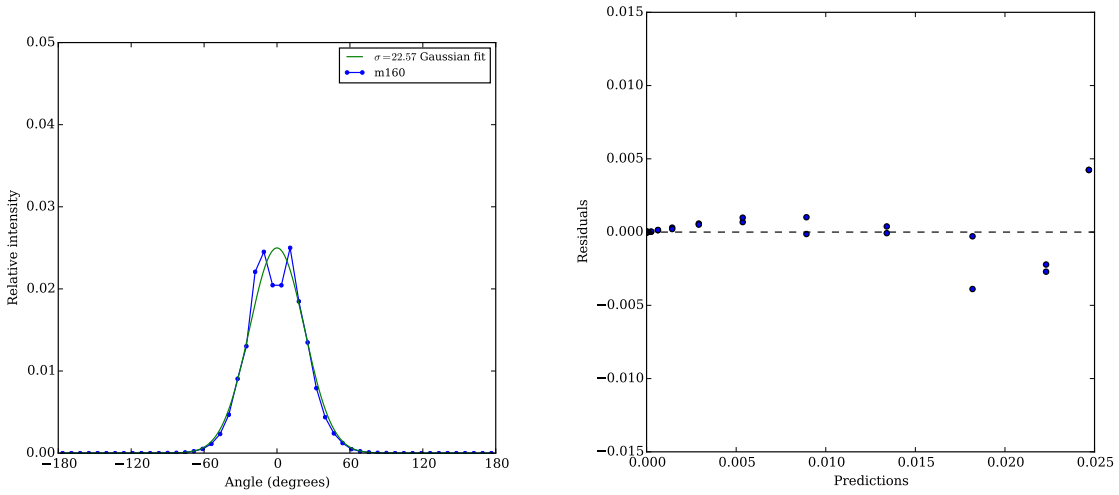


(a) $M = 80$

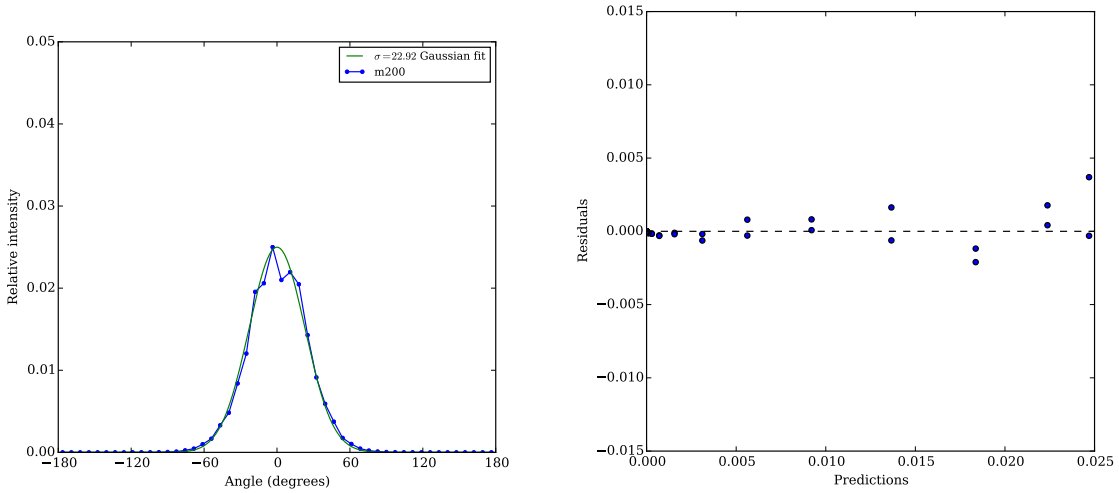


(b) $M = 120$

Figure C.2: Gaussian fit and residuals for $M = 80$ and $M = 120$ beam spread functions.



(a) $M = 160$



(b) $M = 200$

Figure C.3: Gaussian fit and residuals for $M = 160$ and $M = 200$ beam spread functions.

Bibliography

[1] S. Agostinelli, J. Allison, K. Amako, J. Apostolakis, H. Araujo, P. Arce, M. Asai, D. Axen, S. Banerjee, G. Barrand, F. Behner, L. Bellagamba, J. Boudreau, L. Broglia, A. Brunengo, H. Burkhardt, S. Chauvie, J. Chuma, R. Chytracek, G. Cooperman, G. Cosmo, P. Degtyarenko, A. Dell'Acqua, G. Depaola, D. Dietrich, R. Enami, A. Feliciello, C. Ferguson, H. Fesefeldt, G. Folger, F. Foppiano, A. Forti, S. Garelli, S. Giani, R. Giannitrapani, D. Gibin, J.J. Gómez Cadenas, I. González, G. Gracia Abril, G. Greeniaus, W. Greiner, V. Grichine, A. Grossheim, S. Guatelli, P. Gumplinger, R. Hamatsu, K. Hashimoto, H. Hasui, A. Heikkinen, A. Howard, V. Ivanchenko, A. Johnson, F.W. Jones, J. Kallenbach, N. Kanaya, M. Kawabata, Y. Kawabata, M. Kawaguti, S. Kelner, P. Kent, A. Kimura, T. Kodama, R. Kokoulin, M. Kossov, H. Kurashige, E. Lamanna, T. Lampén, V. Lara, V. Lefebure, F. Lei, M. Liendl, W. Lockman, F. Longo, S. Magni, M. Maire, E. Medernach, K. Minamimoto, P. Mora de Freitas, Y. Morita, K. Murakami, M. Nagamatu, R. Nartallo, P. Nieminen, T. Nishimura, K. Ohtsubo, M. Okamura, S. O'Neale, Y. Oohata, K. Paech, J. Perl, A. Pfeiffer, M.G. Pia, F. Ranjard, A. Rybin, S. Sadilov, E. Di Salvo, G. Santin, T. Sasaki, N. Savvas, Y. Sawada, S. Scherer, S. Sei, V. Sirotenko, D. Smith, N. Starkov, H. Stoecker, J. Sulkimo, M. Takahata, S. Tanaka, E. Tcherniaev, E. Safai Tehrani, M. Tropeano, P. Truscott, H. Uno, L. Urban, P. Urban, M. Verderi, A. Walkden, W. Wander, H. Weber, J.P. Wellisch, T. Wenaus, D.C. Williams, D. Wright, T. Yamada, H. Yoshida, and D. Zschiesche. Geant4 – a simulation toolkit. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 506(3):250 – 303, 2003.

[2] Antoine Bouthors. *Realistic Rendering of Clouds in Realtime.* PhD thesis, Université Joseph Fourier, June 2008.

[3] Antoine Bouthors, Fabrice Neyret, and Sylvain Lefebvre. Real-time realistic illumination and shading of stratiform clouds. In *Eurographics Workshop on Natural Phenomena*, 2006.

[4] Antoine Bouthors, Fabrice Neyret, Nelson Max, Eric Bruneton, and Cyril Crassin. Interactive multiple anisotropic scattering in clouds. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games*, pages 173–182. ACM, 2008.

[5] Eva Cerezo, Frederic Pérez, Xavier Pueyo, Francisco J Seron, and François X Sillion. A survey on participating media rendering techniques. *The Visual Computer*, 21(5):303–328, 2005.

[6] Subrahmanyan Chandrasekhar. *Radiative Transfer*. Dover, New York, N.Y., 1960.

[7] Vernon F Cormier, Christopher J Sanborn, Michele Fitzpatrick, Steven Walsh, and Nil Mistry. Modeling the combined effects of deterministic and statistical structure for optimization of regional monitoring. Technical report, DTIC Document, 2015.

[8] S. Q. Duntley. Underwater lighting by submerged lasers and incandescent sources. Technical report, Scripps Institution of Oceanography, June 1971.

[9] Thomas Engelhardt, Jan Novak, and Carsten Dachsbacher. Instant multiple scattering for interactive rendering of heterogeneous participating media. Technical report, Karlsruhe Institut of Technology, December 2010.

[10] Richard P Feynman. Mathematical formulation of the quantum theory of electromagnetic interaction. *Physical Review*, 80(3):440, 1950.

[11] Richard P. Feynman and Albert R Hibbs. *Quantum Mechanics and Path Integrals*. Dover Publications, Mineola, New York, NY, 2010.

[12] T. Goorley, M. James, T. Booth, F. Brown, J. Bull, L. J. Cox, J. Durkee, J. Elson, M. Fensin, R. A. Forster, J. Hendricks, H. G. Hughes, R. Johns, B. Kiedrowski, R. Martz, S. Mashnik, D. Pelowitz G. McKinney, R. Prael, J. Sweezy, L. Waters, T. Wilcox, and T. Zukaitis. Initial MCNP6 release overview. *Nuclear Technology*, 180(3):298 – 315, December 2012.

[13] Howard R Gordon. Equivalence of the point and beam spread functions of scattering media: a formal demonstration. *Applied optics*, 33(6):1120–1122, 1994.

[14] Thomas C Grenfell and David Hedrick. Scattering of visible and near infrared radiation by NaCl ice and glacier ice. *Cold Regions Science and Technology*, 8(2):119–127, 1983.

[15] B. Guenter and R. Parent. Computing the arc length of parametric curves. *IEEE Computer Graphics and Applications*, 10(3):72–78, May 1990.

[16] Kyle Hegeman, Michael Ashikhmin, and Simon Premože. A lighting model for general participating media. In *Proceedings of the 2005 symposium on Interactive 3D graphics and games*, pages 117–124. ACM, 2005.

[17] Carlos P Herrero and Rafael Ramírez. Structural and thermodynamic properties of diamond: A path-integral Monte Carlo study. *Physical Review B*, 63(2):024103, 2000.

[18] Weilin Hou, Deric J Gray, Alan D Weidemann, and Robert A Arnone. Comparison and validation of point spread models for imaging in natural waters. *Optics Express*, 16(13):9958–9965, 2008.

[19] Wenzel Jakob. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org.

[20] Wenzel Jakob. *Light Transport on Path-Space Manifolds*. PhD thesis, Cornell University, August 2013.

[21] Wenzel Jakob and Steve Marschner. Manifold exploration: A Markov chain Monte Carlo technique for rendering scenes with difficult specular transport. *ACM Transactions on Graphics (TOG)*, 31(4):58, 2012.

[22] Wojciech Jarosz, Derek Nowrouzezahrai, Iman Sadeghi, and Henrik Wann Jensen. A comprehensive theory of volumetric radiance estimation using photon points and beams. *ACM Transactions on Graphics*, 30(1):5:1 – 5:19, February 2011.

[23] Henrik Wann Jensen. *The Photon Map in Global Illumination*. PhD thesis, Technical University of Denmark, September 1996.

[24] Henrik Wann Jensen. Rendering caustics on non-Lambertian surfaces. In *Proceedings of Graphics Interface '96*, pages 116–121, 1996.

[25] Henrik Wann Jensen and Per H. Christensen. Efficient simulation of light transport in scenes with participating media using photon maps. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 311–320, New York, NY, USA, 1998. ACM.

[26] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[27] James T. Kajiya. The rendering equation. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '86, pages 143–150, New York, NY, USA, 1986. ACM.

[28] Malvin H. Kalos and Paula A. Whitlock. *Monte Carlo Methods*, volume I. John Wiley & Sons, Ltd., 1986.

[29] George W. Kattawar, editor. *Selected Papers on Multiple Scattering in Plane Parallel Atmospheres and Oceans: Methods*, volume MS 42 of *SPIE Milestone Series*. SPIE Optical Engineering Press, 1991.

[30] George W Kattawar and Charles N Adams. Stokes vector calculations of the submarine light field in an atmosphere-ocean with scattering according to a Rayleigh phase matrix: Effect of interface refractive index on radiance and polarization. *Limnology and Oceanography*, 34(8):1453–1472, 1989.

[31] GW Kattawar and GN Plass. Thermal emission from haze and clouds. *Applied optics*, 9(2):413–419, 1970.

[32] Alexander Keller. Instant radiosity. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, pages 49–56, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[33] D C Khandekar, S V Lawande, and K V Bhagwat. *Path-Integral Methods and their Applications*. World Scientific Publishing Co. Pte. Ltd., P O Box 128, Farrer Road, Singapore 912805, 1993.

[34] Paul Kilgo and Jerry Tessendorf. Accelerated path generation and visualization for numerical integration of Feynman path integrals for radiative transfer. In *Joint International Conference on Mathematics and Computation, Supercomputing in Nuclear Applications and the Monte Carlo Method*, April 2015.

[35] Paul Kilgo and Jerry Tessendorf. Toward validation of a Monte Carlo rendering technique. In *Special Interest Group on Graphics and Interactive Techniques*, August 2015. Poster session.

[36] Eric P Lafortune and Yves D Willems. Bi-directional path tracing. In *Proceedings of CompuGraphics*, volume 93, pages 145–153, 1993.

[37] Eric P Lafortune and Yves D Willems. Rendering participating media with bidirectional path tracing. In *Rendering Techniques 96*, pages 91–100. Springer, 1996.

[38] Samuli Laine, Hannu Saransaari, Janne Kontkanen, Jaakko Lehtinen, and Timo Aila. Incremental instant radiosity for real-time indirect illumination. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*, pages 277–286. Eurographics Association, 2007.

[39] Mohsen Madi. Closed-form expressions for the approximation of arclength parameterization for Bézier curves. *International Journal of Applied Mathematics and Computer Science*, 14(1):33–42, 2004.

[40] Robert A. Maffione, Jeff M. Voss, and Curtis D. Mobley. Theory and measurements of the complete beam spread function of sea ice. *Limnology and Oceanography*, 43(1):34–43, 1998.

[41] LE Mertens and FS Replogle Jr. Use of point spread and beam spread functions for analysis of imaging systems in water. *JOSA*, 67(8):1105–1117, 1977.

[42] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[43] Thomas F. Miller and David C. Clary. Torsional path integral Monte Carlo method for calculating the absolute quantum free energy of large molecules. *The Journal of Chemical Physics*, 119(1):68–76, 2003.

[44] Curtis D Mobley, Bernard Gentili, Howard R Gordon, Zhonghai Jin, George W Kattawar, Andre Morel, Phillip Reinersman, Knut Stamnes, and Robert H Stavn. Comparison of numerical models for computing underwater light fields. *Applied Optics*, 32(36):7484–7504, 1993.

[45] Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*, volume 63 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Capital City Press, Montpelier, Vermont, 1992.

[46] Jan Novák, Derek Nowrouzezahrai, Carsten Dachsbacher, and Wojciech Jarosz. Virtual ray lights for rendering scenes with participating media. *ACM Transactions on Graphics*, 31(4):60:1–60:11, July 2012.

[47] Lev T Perelman, Joshua Winn, Jun Wu, Ramachandra R Dasari, and Michael S Feld. Photon migration of near-diffusive photons in turbid media: a Lagrangian-based approach. *JOSA A*, 14(1):224–229, 1997.

[48] Lev T Perelman, Jun Wu, Irving Itzkan, and Michael S Feld. Photon migration in turbid media using path integrals. *Physical Review Letters*, 72(9):1341, 1994.

[49] Lev T Perelman, Jun Wu, Yang Wang, Irving Itzkan, Ramachandra R Dasari, and Michael S Feld. Time-dependent photon migration using path integrals. *Physical Review E*, 51(6):6134, 1995.

[50] Donald K Perovich. The optical properties of sea ice. Technical report, DTIC Document, 1996.

[51] Ruth Pordes, Don Petravick, Bill Kramer, Doug Olson, Miron Livny, Alain Roy, Paul Avery, Kent Blackburn, Torre Wenaus, Frank Würthwein, Ian Foster, Rob Gardner, Mike Wilde, Alan Blatecky, John McGee, and Rob Quick. The Open Science Grid. *Journal of Physics: Conference Series*, 78(1):012057, 2007.

[52] Simon Premože, Michael Ashikhmin, Ravi Ramamoorthi, and Shree Nayar. Practical rendering of multiple scattering effects in participating media. In *Proceedings of the Fifteenth Eurographics conference on Rendering Techniques*, pages 363–374, 2004.

[53] Simon Premože, Michael Ashikhmin, and Peter Shirley. Path integration for light transport in volumes. In *Proceedings of the 14th Eurographics workshop on Rendering*, pages 25–27. Citeseer, 2003.

[54] JS Schoonmaker, KJ Voss, and GD Gilbert. Laboratory measurements of optical beams in young sea ice. *Limnology and oceanography*, 34(8):1606–1613, 1989.

[55] Benjamin Segovia, Jean Claude Iehl, Richard Mitanchey, and Bernard Péroche. Bidirectional instant radiosity. In *Rendering Techniques*, pages 389–397, 2006.

[56] Benjamin Segovia, Jean Claude Iehl, and Bernard Péroche. Metropolis instant radiosity. In *Computer Graphics Forum*, volume 26, pages 425–434. Wiley Online Library, 2007.

[57] Michael Spivak. *A Comprehensive Introduction to Differential Geometry*, volume II. Publish or Perish, Inc., Berkeley, second edition, 1979.

[58] Jerry Tessendorf. Radiative transfer as a sum over paths. *Physical Review A*, 35(2):872, 1987.

[59] Jerry Tessendorf. Time-dependent radiative transfer and pulse evolution. *JOSA A*, 6(2):280–297, 1989.

[60] Jerry Tessendorf. Numerical integration of the Feynman path integral for radiative transport. In *International Conference on Mathematics, Computational Methods and Reactor Physics*, Saratoga Springs, NY, May 2009.

[61] Jerry Tessendorf. Angular smoothing and spatial diffusion from the Feynman path integral representation of radiative transfer. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 112(4):751 – 760, 2011.

[62] Eric Veach. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, 1997.

[63] Eric Veach and Leonidas J Guibas. Metropolis light transport. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, pages 65–76. ACM Press/Addison-Wesley Publishing Co., 1997.

[64] Kenneth J Voss. Variation of the point spread function in the Sargasso Sea. In *San Diego,'91, San Diego, CA*, pages 97–103. International Society for Optics and Photonics, 1991.

[65] Hongling Wang, Joseph Kearney, and Kendall Atkinson. Arc-length parameterized spline curves for real-time simulation. In *5th International Conference on Curves and Surfaces*, 2002.

[66] Joshua N Winn, Lev T Perelman, Kun Chen, Jun Wu, Ramachandra R Dasari, and Michael S Feld. Distribution of the paths of early-arriving photons traversing a turbid medium. *Applied optics*, 37(34):8085–8091, 1998.

[67] Ru-Shan Wu. Multiple scattering and energy transfer of seismic waves – separation of scattering effect from intrinsic attenuation – I. Theoretical modelling. *Geophysical Journal International*, 82(1):57–80, 1985.