Light Transport Simulation in Participating Media using Spherical Harmonic Methods

Von der Fakultät Informatik, Elektrotechnik und Informationstechnik der Universität Stuttgart zur Erlangung der Würde eines Doktors der Naturwissenschaften (Dr. rer. nat.) genehmigte Abhandlung

Vorgelegt von

David Körner

aus Dresden

Hauptberichter:	Prof. Dr. Bernhard Eberhardt
Mitberichter:	Prof. Dr. Thomas Ertl
Mitberichter:	Prof. Dr. Wenzel Jakob

Tag der mündlichen Prüfung: 01.10.2021

Visualisierungsinstitut der Universität Stuttgart

2021

Contents

List of Figures 5

List of Tables 9

List of Symbols 11

Abstract 13

Zusammenfassung (German Abstract) 15

1 Introduction 17

1.1 Summary of Original Contributions . . . 23
1.2 Organization of the Dissertation . . . 24

2 Foundations of Light Transport Simulation 25

2.1 The Radiative Transfer Equation . . . 26
2.2 Analytical Solutions . . . 30
2.3 Stochastic Methods . . . 31
2.4 Deterministic Methods . . . 37

3 P_N-Method for Multiple Scattering in Participating Media 43

3.1 Spherical Harmonics . . . 44
3.2 Complex-valued P_N-equations . . . 48
3.3 Real-valued P_N-equations . . . 52
3.4 Two-dimensional problems . . . 56
3.5 P_N-Solver with Automated Stencil Generation . . . 58
3.5.1 Computer Algebra Representation . . . 62
3.5.2 Discretization . . . 63
3.5.3 Stencil Code Generation . . . 64
3.5.4 Boundary Conditions . . . 66
3.5.5 Solver Framework . . . 68
3.5.6 Staggered Grids . . . 70
3.5.7 System Matrix and Solution . . . 75
3.6 Integration into Rendering Software . . . 77
3.7 Results . . . 81
3.7.1 Point Source Problem . . . 81
3.7.2 Checkerboard Problem . . . 82
3.7.3 Procedural Cloud . . . 83
3.8 Discussion . . . 85

4 Flux-limited Diffusion for Multiple Scattering in Participating Media 87
4.1 Moment Expansion of the Radiance Field 88
4.2 Moment Expansion of the Radiative Transfer Equation 93
4.3 Moment Closure Problem and Isotropic Closure 97
4.4 Diffusion Approximation 100
4.4.1 Point Source Problem 101
4.4.2 Procedural Cloud 102
4.5 Moment Closure Problem Revisited 103
4.6 Streaming Limit Approximation 107
4.7 The Variable Eddington Factor 110
4.8 Flux-limiters 113
4.9 Non-Linear Gauss-Seidel Solver with Successive Overrelaxation 116
4.10 Results 119
4.10.1 Point Source Problem 120
4.10.2 Procedural Cloud 121
4.11 Discussion 122

5 Conclusion 125

Bibliography 131

A Appendix: Derivation of Real-valued P_N-equation Terms 141

B Appendix: Multigrid solver 159

List of Figures

1.1 Real world examples of participating media. Reproducing these optical phenomena with computers requires resource intensive light transport simulations for which deterministic methods are developed in this thesis[1]. 18
1.2 Light transport simulation for rendering in graphics shares problems with other domains such as astrophysics and nuclear sciences. This thesis draws ideas from the long history of research done in those fields and inspires new methods for application in graphics. 20
1.3 Organigram of this thesis showing original contributions in green. . . . 24

2.1 Directional derivatives of the radiance L along direction $\boldsymbol{\omega}$ model the interaction of light with participating media and constitue the radiative transfer equation. 27
2.2 Path sample and its contributing factors to energy loss of radiant energy from the light source to the sensor. 33
2.3 This result from Camminady et al [10] demonstrates the ray effects produced by the discrete ordinate method (left) in comparison to the ground truth (right) for the checkerboard problem. We also simulate the checkerboard problem for our methods (see 3.7.2). 40
2.4 Overview of deterministic methods for light transport simulation. . . . 42

3.1 Approximating a spherical function using spherical harmonics with increasing truncation order (from left to right). Like with the Fourier-transform, higher truncation order allows capturing higher frequencies. 45
3.2 Image of finite difference grid showing domain (white cells), boundary (gray cells), gridpoints (red), voxelsize h and interpolation. 59
3.3 Spatial discretization of partial differential equations will lead to stencil code which is executed to populate the system matrix A and right hand side vector b. 61
3.4 Overview of the method for solving the P_N-equations which is introduced in this chapter. The core idea is to automate the stencil code generation using a computer algebra representation of the equations. . 61
3.5 An Python-API is used to create the expression tree representation for mathematical expressions. 62
3.6 Algebraic manipulations of equations are expressed as operations on the expression tree given by the computer algebra representation. . . . 63

3.7 Discretization is carried out as an operation on the expression tree where partial derivatives are replaced by finite difference subtrees. Continous variables are replaced by discrete counterparts, for which offsets are retrieved from a stack that is maintained during tree parsing. . . . 63
3.8 Continuous field variables are turned into discrete counterparts by replacing them with an expression subtree which realizes an interpolation. 64
3.9 Rendering frontends allow turning expression trees into source code. . 65
3.10 The discretized equation in canonical form allows straightforward generation of stencil code from the different parts of the expression tree. . 65
3.11 Stencil code generation assumes an API for querying voxel fields of radiative quantities and accessing entries in coefficient matrix A and right hand side b 66
3.12 Dirichlet boundary conditions with zero value are modelled by ignoring coefficient assignments to boundary voxels. 67
3.13 Neumann boundary conditions with zero value are modelled by remapping indices towards the adjacent inner voxel when assigning coefficients. 68
3.14 High level overview over the P_N-solver architecture. 68
3.15 Artefacts in the solution (left) due to shortcomings of collocated grid locations. These are addressed using staggered grids (exemplified right). 71
3.16 Staggered grid configuration in 2d. Extending the concept to 3d is straightforward. 72
3.17 The process of finding staggered grid locations for unknowns ϕ_i. The P_N-equations relate the unknowns through spatial derivatives, which imply an order on the finite difference grid. The first unknown is placed at the voxel center and the remaining unknown locations are found by following the spatial derivatives through the equations. 72
3.18 Dealing with staggered grid locations per voxel (left) implies a boundary interface which produces asymmetric results as seen for the solution of the checkerboard problem (right). 74
3.19 Placing staggered grid coefficients such that the defined boundary interface is symmetric (left) leads to a correct symmetric solution of the checkerboard problem (right). 74
3.20 Structure of coefficient matrix A and solution vector $\mathbf{u}$ after discretization of the P_N-equations on a finite difference grid. 75
3.21 Overview over the full rendering process. The single scattering solution is precomputed and projected into RHS Q. The coefficient matrix A is assembled from voxelized RTE quantities and used to compute solution u. That solution is then un-projected and used to evaluate the multiple scattering contribution during final rendering step. 80
3.22 Comparison of P_N-results for the pointsource problem with different truncation order N against groundtruth solution (Grosjean). The disagreement on the right comes from zero dirichlet boundary conditions. 82

3.23 Solver P_5-result for the checkerboard problem (left) in comparsison to StaRMAP by Seibold et al. [81] (right). The solution of the solver, developed as part of this thesis, is in very good agreement. 83
3.24 P_N-results for the procedural cloud dataset (multiple scattered light) with different truncation order N. 85
3.25 Convergence of P_5 for different choices of minimum threshold σ_{min} and its effect on the conditional number for the coefficient matrix A. 86

4.1 Ficks first law of diffusion (visualized on the right) states that the diffusive flux can be related to the difference in concentration (of fluence ϕ), allowing to approximate a global quantity (left) by a local quantity. This is the fundamental idea behind diffusion approximation. 100
4.2 Solution of the classical diffusion approximation (CDA) for the point source problem (red line) compared against ground truth (black), P_1 (red dots) and P_5 (green). 101
4.3 Diffusion approximation results for the procedural cloud dataset (multiple scattered light) compared against Monte-Carlo reference, P_1 and P_2. 102
4.4 Intuition behind parameterizing the transport regime according to mean free path (average distance between two consecutive scattering events). Small mean free paths imply higher medium density, while larger mean free paths imply lower density or vacuum in case of infinity. 105
4.5 The intuition behind parameterizing the transport regime according to the ratio between the total amount of light arriving at a point and the directionality given by the magnitude of the flux vector (red in the figure). A small ratio results from small directionality in relation to large amounts of light arriving. This implies a uniform distribution of light coming from all directions, which is the case for diffusive transport. The ballistic transport regime is characterized by increased directionality in relation to the amount of light. In free streaming transport, the magnitude of the flux vector equals the amount of light arriving. This is the case when light comes from a single direction. 106
4.6 Illustration of diffusive (left) and advective (right) transport of particles. With diffusion, particles spread locally depending on the gradient of the particle distribution, which for flux-limited diffusion is the zero moment, the fluence. Advection is the movement of particles along a directed force field. For flux-limited diffusion, this force field is defined by the first moment, the flux vector. 110
4.7 Radiative transfer in typical scenes contains a mix of transport regimes as shown in this figure. Diffuse transport in dense media with high absorption and advective transport in thin media. The idea behind the Variable Eddington Factor is to express radiative transfer as a mix between those two extreme transport modes. 111

4.8 Plot of the various flux-limiters shown in table 4.1. 116
4.9 Comparing the solution from flux-limited diffusion (blue) against classical diffusion approximation (red), P_5-solution (green) and groundtruth (black). Flux-limited diffusion much closer to the groundtruth and P_5-result at much lower computational cost. 121
4.10 Flux-limited diffusion results for the procedural cloud dataset compared against previous methods. FLD produces results which are similarily close to the pathtraced solution than P_5 at a fraction of its computational cost. 123

5.1 Flux-limited diffusion is used to compute multiple scattering in Elementacular[2], a commercial plugin for interactive cloud modelling. Due to the incremental nature of interactive edits, the result from previous frames is a great initial solution for the solver, which then only needs a few iterations to re-solve the underlying equation system. 127

B.1 The staggered coarse grids for the multigrid v-cycle are computed from interpolating coefficients at their coarse grid locations. 162

List of Tables

3.1 Performance characteristics of the P_N-method for the point source problem for a 64×64×64 sized grid. 82

3.2 Performance characteristics of the P_N-method for the procedural cloud problem for a 64×64×64 sized grid with a minimum extinction coefficient threshold of 4. 84

4.1 Various prominent flux-limiters which all respect the Eddington factor limits and represent different flux-limited diffusion theories. 115

4.2 Performance characteristics of flux-limited diffusion and P_1 for the point source problem. 121

List of Symbols

l	mean free path $[\mathrm{m}]$
σ_a	absorption coefficient $\left[\mathrm{m}^{-1}\right]$
σ_s	scattering coefficient $\left[\mathrm{m}^{-1}\right]$
$\sigma_t = \sigma_a + \sigma_s$	extinction coefficient $\left[\mathrm{m}^{-1}\right]$
σ_{min}	extinction coefficient minimum threshold $\left[\mathrm{m}^{-1}\right]$
α	single scattering albedo σ_s/σ_t
g	spherical function mean cosine
$\mathbf{x}$	position vector
$\bar{\mathrm{x}}$	path sample
$\boldsymbol{\omega}$	unit length direction vector
$D(\mathbf{x})$	diffusion coefficient $[\mathrm{m}]$
$R(\mathbf{x})$	transport regime measure
$L(\mathbf{x},\boldsymbol{\omega})$	radiance field $\left[\frac{\mathrm{W}}{\mathrm{srm}^2}\right]$
$L_u(\mathbf{x},\boldsymbol{\omega})$	unscattered light radiance field $\left[\frac{\mathrm{W}}{\mathrm{srm}^2}\right]$
$L_s(\mathbf{x},\boldsymbol{\omega})$	single scattered light radiance field $\left[\frac{\mathrm{W}}{\mathrm{srm}^2}\right]$
$L_m(\mathbf{x},\boldsymbol{\omega})$	medium radiance field (single and multiple scattered light) $\left[\frac{\mathrm{W}}{\mathrm{srm}^2}\right]$
$\widehat{L}(\mathbf{x},\boldsymbol{\omega})$	radiance field truncated spherical harmonics reconstruction $\left[\frac{\mathrm{W}}{\mathrm{srm}^2}\right]$

$\phi(\mathbf{x})$	fluence (zero moment of the radiance field) $\left[\frac{\mathrm{W}}{\mathrm{m}^2}\right]$
$\mathbf{E}(\mathbf{x})$	flux-vector (first moment of the radiance field) $\left[\frac{\mathrm{W}}{\mathrm{m}^2}\right]$
$P(\mathbf{x})$	radiative pressure tensor (second moment of the radiance field)
$\rho(\mathbf{x},\boldsymbol{\omega}'\cdot\boldsymbol{\omega})$	phase function $[\mathrm{sr}]$
$Q_e(\mathbf{x},\boldsymbol{\omega})$	emission field $\left[\frac{\mathrm{W}}{\mathrm{srm}^2}\right]$
$E[f]$	expected value of random variable f
$\mathbf{u}$	discretized radiance field (solution vector)
A	coefficient matrix from discretized radiative transfer operators
$L^{l,m}$	radiance field spherical harmonics coefficients
$\rho_{R(\boldsymbol{\omega})}(f)$	operator for rotating radial functions about vector $\boldsymbol{\omega}$
Ω	solid angle domain of the unit sphere
h_x,h_y,h_z	finite difference grid voxelsize
$\odot$	tensor contraction operator
$\mu_n[\cdot]$	n-th moment expansion of a given spherical function
$\mathbf{I}$	identity matrix
χ	variable Eddington factor

Abstract

This thesis introduces novel numerical methods for light transport simulation in computer graphics which are based on the spherical harmonics discretization of angular variables in the governing equations.

Light transport simulation is required for generating realistic images from virtual scenes and therefore ubiquitous in digital media today. Also, an increasing presence of light transport simulation in the domain of scientific visualization can be observed. This can be explained by the fact that the visual system of a human being is optimized for reading real world cues, like surface smoothness from specularity, or a shape from indirect illumination. The early stage of realistic image generation in computer graphics relied on ad-hoc techniques and heuristics due to limitations in computing power. With the growth in computing power, a steady increase in model accuracy and physical rigor has become feasible.

Today, the accuracy of human perception defines the domain of light transport simulation in computer graphics and remains the single aspect that sets it apart from transport simulation in other domains, such as nuclear science or astrophysics. With increased conformity towards traditional physics in terms of models, terminology and methodology, the opportunity increased to draw inspiration from other fields, their large body of ideas and methods. In fact, an extensive amount of research in light transport simulation for computer graphics is concerned with the translation, reframing, and enhancement of those techniques by exploiting limited requirements in accuracy.

In other domains, such as nuclear sciences, astrophysics or medical physics research, the spherical harmonics based angular discretization of the underlying light transport equations led to a large variety of methods. One of those methods, the diffusion approximation, has been introduced to graphics with great success in the past. A reason for the lack of application of other methods in graphics lies in the complexity of the underlying theory, which further results in large and unwieldy systems of equations. Nevertheless, it is important to study them thoroughly and understand their characteristics with regard to applications in computer graphics, where deterministic methods play a significant role and have led to major advances in the past.

This thesis gives an extensive derivation of the underlying theory of spherical harmonic methods and presents it in the context of light transport simulation for computer graphics. A deep and holistic view of the spherical harmonics methods is taken and their application to computer graphics is investigated. The results are two novel deterministic methods for solving light transport problems in graphics.

First, the thesis introduces the P_N-Method to the field of computer graphics. This method allows for spherical harmonics expansion up to an arbitrary level. To handle the complex equations, a fully automated discretization framework is devised as part of a solver that is driven by a computer algebra representation of the underlying transport equations. This novel strategy may inspire new methods for unsolved numerical problems in computer graphics. The chapter further presents a rendering framework and shows how to integrate the solution from any method based on spherical harmonics representation by separating direct from indirect illumination. The results show solutions for a typical volume rendering problem and comparisons against groundtruth solutions for canonical problems.

The P_N-method is based on the generalized spherical harmonics expansion up to arbitrary level. While expanding to higher order produces more accurate results, the method also becomes more and more unpractical due a system of equations which increases in size with higher order. In the fourth chapter, this thesis therefore examines flux-limited diffusion and introduces it to the field of computer graphics. It is based on the idea of truncating the spherical harmonics expansion at a very low order and reducing the error by making assumptions about the higher order contribution of light. The result is a novel method, which allows for the interactive computation of multiple scattering in highly scattering, heterogeneous participating media – a challenging problem for standard Monte-Carlo methods.

Zusammenfassung (German Abstract)

Diese Dissertation führt neue numerische Methoden zur Lichttransportsimulation in der Computergraphik ein, welche auf der Diskretisierung der Winkelvariablen durch Kugeloberflächenfunktionen in den zugrunde liegenden Gleichungen basieren.

Lichttransportsimulation ist in den digitalen Medien allgegenwärtig und notwendig, um realistische Bilder virtueller Szenen zu generieren. Auch wird sie zunehmend in der wissenschaftlichen Visualisierung eingesetzt. Dies erklärt sich dadurch, dass die visuelle Wahrnehmung des Menschen auf das Lesen realer Umgebungen trainiert ist, wie zum Beispiel die Glattheit einer glänzenden Oberfläche, oder die Form, welche durch indirekte Beleuchtung entsteht. Die frühen Anfänge der Generierung von realistischen Bildern in der Computergraphik stützten sich aufgrund der limitierten Rechenleistung auf ad-hoc Techniken und Heuristiken. Mit schneller werdenden Computern kam ein Wachstum an Modellgenauigkeit und ein Anspruch an physikalische Korrektheit einher.

Gegenwärtig wird das Feld der Lichttransportsimulation in der Computergraphik durch die Grenzen der menschlichen Wahrnehmung definiert. Dies ist ein Aspekt, welcher eine Abgrenzung zur Simulation von Lichttransport in anderen Fachbereichen wie Astrophysik oder Kernphysik darstellt. Durch die Angleichung an die traditionelle Physik in Bezug auf Modelle, Terminologie und Methodik, ergibt sich zunehmend die Gelegenheit, Ideen und Methoden anderer Fachbereiche in der Computergraphik anzuwenden. Tatsächlich befasst sich ein großer Teil der Forschung zur Lichttransportsimulation in der Computergraphik mit der Übersetzung, Neuausrichtung und Verbesserung der Methoden unter Berücksichtigung der schwächeren Anforderungen an die Genauigkeit.

In anderen Fachbereichen wie der Kernphysik, Astrophysik oder Medizinforschung haben Winkeldiskretisierungen mit Kugeloberflächenfunktionen zu einer großen Zahl an Methoden geführt. Eine dieser Methoden, die Diffusionsapproximation, wurde in der Vergangenheit mit großem Erfolg in der Computergraphik eingeführt. Der Grund, warum andere Methode nicht in der Computergraphik eingesetzt werden, liegt unter anderem daran, dass die zugrunde liegende Theorie sehr komplex ist und in große, sperrige Gleichungssysteme mündet. Denoch ist es wichtig, diese Methoden gründlich zu untersuchen und ihre Eigenschaften in Hinblick auf die Anwendung auf bestehende Probleme in der Computergraphik zu verstehen, wo deterministische Methoden bisher von großer Bedeutung waren.

Diese Dissertation gibt eine ausführliche Herleitung der Theorie der Lichttransportsimulation auf Basis von Diskretisierung mit Kugeloberflächenfunktionen und präsentiert sie im Kontext der Lichttransportsimulation für Computergraphik. Sie wirft einen tiefgehenden und ganzheitlichen Blick auf die gesamte Klasse an Methoden, welche diese Diskretisierung nutzen und untersucht ihre Anwendung in der Computergraphik. Das Ergebnis sind zwei neue deterministische Methoden zur Lösung von Lichttransportproblemen in der Computergraphik.

Zuerst führt diese Dissertation die P_N-Methode in die Computergraphik ein. Diese Methode erlaubt eine Entwicklung in Kugeloberflächenfunktionen bis zu beliebig hoher Ordnung. Um die komplexen Gleichungen zu verarbeiten, ist ein vollautomatisches Diskretisierungsframwork als Teil eines Lösers entwickelt worden, welcher von einer Computer-Algebra-Repräsentation der zugrunde liegendenden Gleichungen gespeist wird. Diese neue Strategie inspiriert möglicherweise neue Methoden für ungelöste numerische Probleme in der Computergraphik. Das Kapitel präsentiert zudem ein Rendering-framework und zeigt, wie Lösungen von Methoden, welche auf der Entwicklung in Kugeloberflächenfunktionen basieren, für das Rendering verwendet werden können. Es werden Ergebnisse für typische Volume-Rendering-Probleme gezeigt, als auch Vergleiche mit Referenzlösungen kanonischer Probleme.

Die P_N-Methode basiert auf der Entwicklung in Kugeloberflächenfunktionen. Während die Entwicklung in höhere Ordnungen bessere Ergebnisse produziert, wird die Methode gleichzeit unpraktischer wegen eines immer größeren Gleichungssystems, das zu lösen ist. Im vierten Kapitel wird daher die flux-limitierte Diffusion untersucht und in die Computergraphik eingeführt. Sie basiert auf der Idee, die Entwicklung in Kugeloberflächenfunktionen auf eine niedrige Ordnung zu beschränken und den Approximationsfehler mit Hilfe von Annahmen über die Lichtverteilung in der höheren Ordnung zu reduzieren. Das Ergebnis ist eine neue Methode, welche erlaubt, die Mehrfachstreuung in einem hochstreuenden partizipierenden Medium zu berechnen – eine Herausforderung für gewöhnliche Monte-Carlo Methoden.

Chapter 1

Introduction

The need and desire to faithfully reproduce the surrounding world using images is rooted deep in human nature. Images of our world have always played a fundamental role in human culture for telling stories, passing on knowledge and building visual maps of reality. Realistic images of the world are in essence a tool for humans to reflect upon it, learn from it and build ambitious new visions for it. There is a consistent line of development from cave and rock carvings dating back to 30,000 BC encompassing wall paintings and burial chambers in Egypt up to well reknown painters such as Da Vinci or Rembrandt. These works of art integrate elements of human expression that is the result of creative processes. However, there has also been the element of technique and understanding of nature, which was required to create realistic imagery. Painters such as Caspar David Friedrich would extensively study cloud shapes and cloud formations to be able to produce images of unprecedented realism.

With the computer, a completely new medium emerged, and with it a new methodology and way of thinking about realistic image creation. Crude rudimental rules, such as on aerial perspective by Leonardo Da Vinci [18], evolved into highly accurate physical models and instead of a paint brush, numerical methods are employed to compute pixel colors which create an image. This is the domain of computer graphics, a branch in computer sciences, which aims to develop methods for the generation of photorealistic images from digital content and constantly tries to push the boundaries of what is technically possible.

Computer generated imagery (CGI) is used widely across many domains and industries such as architecture, entertainment, advertising, automotive, manufacturing and scientific visualization, where each brings its own challenges with it. This high demand and wide adoption is why rendering in computer graphics is a heavily researched sub-

Figure 1.1: *Real world examples of participating media. Reproducing these optical phenomena with computers requires resource intensive light transport simulations for which deterministic methods are developed in this thesis*[1].

ject in industry and academia alike. The main challenge for research in rendering is trying to address the high amount of computation current algorithms require in order to generate realistic images. For example a single frame in an animation movie by Disney or Pixar takes multiple hours on average and much more in extreme cases. To produce all images for the whole movie, multiple computing centers with thousands of CPU cores are used, which requires substantial economical investments. One of the main efforts in rendering research therefore is to reduce the computational cost for realistic image generation.

The advent of graphics computing hardware has paved the way for realtime applications. Here, together with some additional compromises in physical accuracy, incredible performance gains enable the generation of images in fractions of seconds and are the basis for new powerful interactive applications in the industries mentioned above. Finding methods that can generate results at interactive rates while allowing control, or at least understanding of accuracy losses, is another important research direction in graphics.

Computer hardware is ever-evolving – while for a long time most of the increase in computing power came from higher density semiconductor fabrication – this trend has almost come to a halt. Today, the growth in compute power is driven by parallelization on multi- and manycore systems. These challenges impact current rendering algorithms which mostly do not play well to the intricacies of the underlying hardware. While some first steps are being made in this direction (see for example Eisenacher et al. [25]), designing rendering algorithms that fully exploit the potential of next

[1]Photos provided under Creative Commons CC0 license by Magda Ehlers (clouds), pixabay (iceberg), Brett Sayles (candle).

generation multi- or manycore hardware has been proclaimed as one of the future challenges for rendering by industry leaders (see Fascione [26]).

This thesis seeks to address these problems by following a trend which has been spurred by the so called physically based rendering revolution in graphics (see Keller et al. [46]). In the early days of computer graphics, limitations in computing power required the use of cheap ad-hoc methods and other tricks to facilitate efficient image generation algorithms. With the growth in computational power, techniques and methods based on accurate physical models have become increasingly feasible and started replacing older heuristic approaches. Rendering became concerned with energy conservation, the use of correct physical quantities for describing the virtual scene and quantifiable error to physically correct ground truth results. Consistently adopting the framework provided by optical physics allowed rendering researchers to much better investigate and compare with work in other fields attacking very similar problems. While graphics research has always drawn inspiration and methods from other scientific domains, the emergence of physically based rendering caused a significant increase in research work which is based on findings and theories from other domains. Physics provided a common language allowing much easier to talk across disciplines.

In this spirit, this thesis investigates theories and methods from the astrophysics domain and nuclear physics and adopts them for application in rendering. The resulting work presented in this thesis addresses all the three challenges mentioned above. Specifically, methods are presented which allow to reduce the computational cost for the demanding problem of rendering participating media such as clouds, smoke, fire, etc. and even allows – with some compromise in accuracy – applications at interactive rates. The methods devised in this thesis turn the rendering problem into the problem of solving a linear system of equations, exactly what high performance computing clusters (large scale many-core systems) have been designed for (see Koerner et al. [50]). This thesis constitutes a significant effort in putting forward an alternative approach to light transport simulation which has not seen a lot of attention lately but might have the potential to answer some of the research challenges in rendering today and tomorrow.

This thesis is concerned with rendering volumetric media such as clouds, fire, milk, and fog which interacts with light. These participating media elements are ubiquitous in the real world and therefore also common in rendering. Today's standard methods in rendering operate by tracing particles of light (photons) through a given scene. These particles interact with the scene and form complex trajectories from light sources to the sensor of the virtual camera which captures the image. Participating media such as clouds or milk cause very long trajectories for light particles and therefore are computationally very demanding.

The problem of tracing particles through a participating medium which causes many particle interactions and long trajectories is not exclusive to simulation of light in computer graphics. Very similar problems exist in nuclear physics for example, when computing the dose of radiation leaving the protected core of a nuclear power plant. Here, high-energy neutrons are traced from the reaction through the surrounding wall, which acts as a participating medium (e.g. Haghighat et al. [37]). In astrophysics, for example, the study of the formation of planets (which apparently to this day is not fully understood) has models that take the radiation of stars into account, and the associated simulation, therefore requires tracing particles that are emitted from fusion through clouds of debris and dust (see Armitage et al. [3]).

Common to all the problems mentioned above is the migration and interaction of particles or energy within a dense host medium. This is being described by linear transport theory, a field in mathematical physics, which is at the intersection between astrophysics, nuclear sciences and computer graphics.

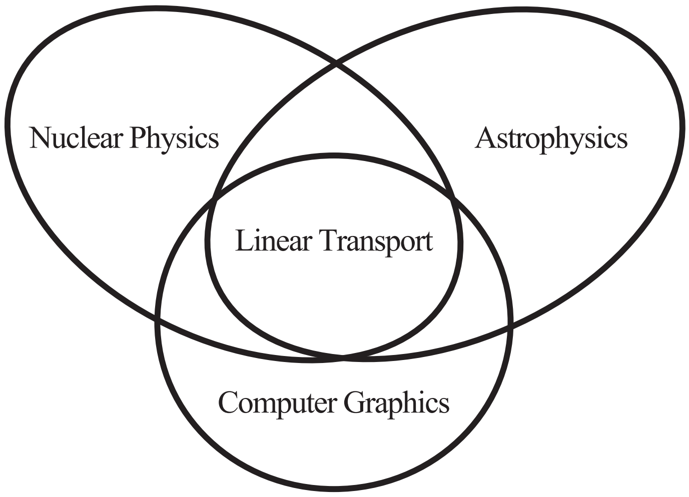


Figure 1.2: *Light transport simulation for rendering in graphics shares problems with other domains such as astrophysics and nuclear sciences. This thesis draws ideas from the long history of research done in those fields and inspires new methods for application in graphics.*

Now astrophysics and nuclear sciences have a much longer standing history than computer graphics and a lot of research was done before modern computers were widely available. This led to a large volume of work in those fields dealing with very similar problems in graphics from a very different perspective. As a result there are many interesting ideas, methods and techniques that can be adapted for applications in computer graphics.

D'Eon [21] in his work explains vividly the challenges this type of research comes with. Starting with notation and terminology which makes it difficult to read and understand papers from these fields. Further, many seminal texts from those other domains are out of print, expensive to aquire, lack offical digital distribution formats and generally are hard to find. Since much of academic research was published before modern computers became a commodity, most publications focus exclusively on theoretical treatments and simple one-dimensional or two-dimensional problems. Practical computational methods are lacking and simply were not focus of research back then. New methods will have to be devised after going through the literature and putting the puzzle pieces together. However, doing this type of translational research can lead to powerful new tools in graphics.

Significant recent advances in computer graphics are based to a large degree on methods which were invented in other scientific domains. For example methods for fast subsurface scattering like the one from D'Eon et al. [22], are based on a half-space approximation introduced by Brinkworth [7]. Vorba et al. [88] derived their adjoint-driven russian roulette from the weight window method introduced by Booth [5]. And zero-variance sampling by Krivanek et al. [52] came out of an investigation in sampling techniques introduced by Dwivedi [24]. All these inspirations came from other fields and were applied to great effect to rendering in computer graphics.

Like the work mentioned above, this thesis follows the same spirit of adapting an existing method from another field for the application to image generation in computer graphics. The P_N-method (see Brunner [8]) is a very popular method in nuclear sciences. It allows to compute a global solution without using the computational demanding standard method of following particle trajectories. The classical diffusion approximation (see Ishimaru [40]) is a method which is used in the astrophysics domain and actually has been introduced to graphics by Stam [82]. However an extension to flux-limited diffusion was introduced in the astrophysics domain by Levermore et al. [56] and has not been introduced to graphics yet. What all these different methods have in common is the fact that they are based on the discretization in angular domain using spherical harmonics.

Research Questions

This thesis explores whether advanced methods, based on the spherical harmonics discretization in angular domain, have merit for applications in computer graphics. More specifically, we seek to learn whether the P_N-method and flux-limited diffusion can be adopted for use in rendering. This endeavor can be formulate as a series of research questions:

Research question 1

How can the theory behind the P_N-method and behind flux-limited diffusion be integrated into the theoretical framwork used in rendering. For this question, the theory needs to be revisited and presented in the context of computer graphics. How is the theory derived from first order principles? Is it useful for applications in rendering? What are the theoretical limitations and how do they affect the scope of applications?

Research question 2

Following the theoretical treatment, we seek to find practical and efficient methods for integration into common rendering frameworks. How can the theoretical problem be solved for a typical application in rendering? How could a practical and efficient method look like? What shortcomings does it have and how could those be overcome?

Research question 3

Finally, we seek to understand how the P_N-method and flux-limited diffusion compare against classical diffusion and other families of rendering techniques. What are the benefits and what are the shortcomings? Which light transport effects are well suited for the new techniques and are there any effects for which the new techniques do not work well or even break down?

1.1 Summary of Original Contributions

The research which was carried out as part of this thesis resulted in two major contributions, where one is based on the P_N-method popular in nuclear sciences and the other uses flux-limited diffusion from the field of astrophysics. In both cases practical methods for full three-dimensional problems did not exist and had to be devised, resulting in original published contributions in its own right.

P_N-Method for Multiple Scattering in Participating Media [51] The first contribution is a novel deterministic method for solving the P_N-equations on a finite difference grid for applications in rendering. The P_N-theory is derived and presented for a computer graphics audience. An original side product is a very concise and compact form of the real-valued P_N-equation which is new to the whole literature on P_N-based methods. The devised method introduced in this thesis is based on the idea of automated discretization. This approach works not only for the P_N-equations, but also for arbitrary potentially coupled PDE's and therefore has the potential to be of interest for other applications and problems in graphics. The P_N-method produces large systems of partial differencial equations which are well suited for being solved on a high performance computing cluster. An initial investigation into this direction was presented in **High-Performance Computing for Artistic Content Creation** [50].

Flux-limited Diffusion for Multiple Scattering in Participating Media [48] The second contribution is a new and efficient method for computing the contribution of multiple scattered light in participating media. The method is based on flux-limited diffusion theory which is being derived and presented for a computer graphics audience. The moment closure problem and variable Eddington factors are introduced and a non-linear Gauss-Seidel method is developed to solve it. Furthermore, the results and performance traits are compared against the P_N-method and classical diffusion approximation to give a clear understanding about the merits of each method.

Finally, an additional contribution of this thesis is the clear and detailed discussion of the theory behind deterministic methods based on the spherical harmonics discretization of the linear transport equation. Furthermore, the classical diffusion approximation is derived and a multigrid method for solving it is presented, making this thesis a comprehensive treatment on the subject.

The author also pursued another set of topics during his studies – while these resulted in a peer-reviewed publication **Subdivision Next-Event Estimation for Path-Traced Subsurface Scattering** [49] – they are only peripheral to the body of this thesis and therefore not discussed.

1.2 Organization of the Dissertation

At the core of this thesis are chapters 3 and 4. Each chapter is related to one of the methods which can be derived from the spherical harmonics discretization of the underlying transport equation and contains a detailed derivation of the theory behind the method the chapter is dedicated to. Then in each chapter, a method for solving the respective equation is devised. These core chapters are complemented with a chapter on the foundations on light transport simulation (chapter 2). This section gives the radiative transfer equation and outlines the main approaches for solving it. Stochastic methods are introduced briefly and contrasted against deterministic methods to which spherical harmonics based discretizations covered in this thesis belong to. Finally, the thesis closes with chapter 5 that revisits the major conclusions from the three core chapters and identifies future directions of research. Figure 1.3 gives a visual overview of the structure of this thesis.

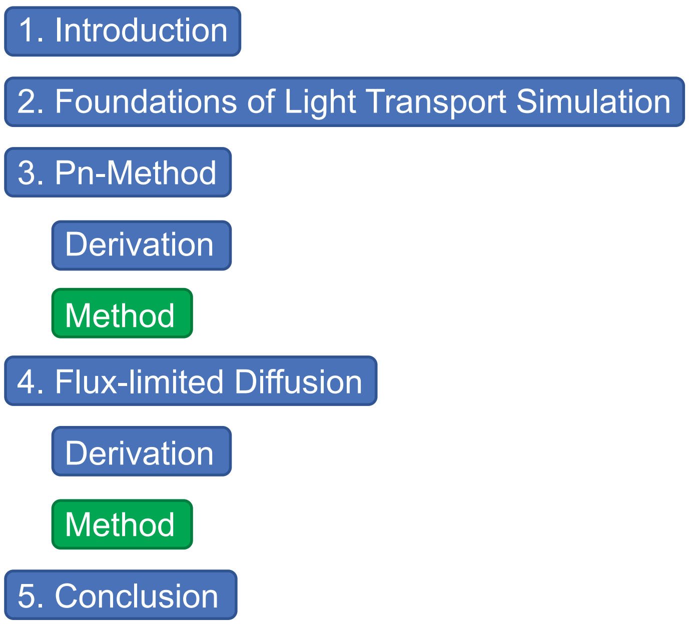


Figure 1.3: *Organigram of this thesis showing original contributions in green.*

Chapter 2

Foundations of Light Transport Simulation

This chapter gives an overview of light transport simulation in the context of rendering in computer graphics and specifically reviews the family of deterministic spherical harmonics based methods. After the introduction of the physical background and important terminology, a brief overview over the landscape of methods for solving light transport problems is given. This is divided into three sections as per the following categorization: analytical methods (section 2.2), stochastic methods (section 2.3) and deterministic methods (section 2.4).

The field of computer graphics is primarily concerned with the generation of images from virtual scenes as seen by a virtual camera. This requires to reproduce the emission of light and its interaction with the scene and the sensor. The study of the behavior of light is subject to optics, a separate branch of physics. Due to the intricate nature of light, a series of increasingly complex models exist, which can account for various properties. Fundamental to light is its particle-wave duality, which states that light can be described both, as individual particles (photons) or waves. Challenging to the subject is, that not all properties of light can be described by solely the particle nor the wave perspective. To account for all properties, both have to be considered and require quantum mechanics. The field of wave optics is based on a simpler model that can account for optical effects, caused by the wave property, such as interference and diffraction. The simplest model for light transport is provided by geometric optics, which models light as particle trajectories or rays and is referred to as ray optics. This model works well when the wavelength is small compared to the scene in which the light propagates.

As the virtual (and real) camera is supposed to capture the image as closely as possible to human perception, the simulated wavelength of light is limited to the visible range of a human being (380 to 740 nanometers). Because this is very small compared to distances encountered in real world scenes, wave effects are of less importance and the model of geometric optics is sufficient most of the time. It is the basis for light transport simulation in graphics. This thesis is exclusively concerned with light transport in participating media i.a. clouds, smoke, fire, and water.

2.1 The Radiative Transfer Equation

Geometric optics describes light as particles, that travel through the scene along straight lines between single interactions. At each interaction, the particle may change its course and continue to travel along another straight segment. When interacting with a medium, such as a cloud, each particle undergoes numerous interactions and creates long trajectories based on medium properties.

To derive a model that can explain this complex system of many particles interacting with the scene, a statistical approach is taken. Since photons carry energy, a collection of photons can be described by their collective energy. To express the propagation of photons, time is required. This is captured by the quantity flux ψ: it gives the amount of energy from all photons going through a surface within a time interval. It is given in Watts, which is specified in Joule (Energy) per second and depends on position $\mathbf{x}$ and direction $\boldsymbol{\omega}$:

$$\psi(\mathbf{x},\boldsymbol{\omega}) \quad \left[\mathrm{W} = \frac{\mathrm{J}}{\mathrm{s}}\right]$$

To find the flux for arbitrary surfaces (e.g. the camera sensor) and directions (e.g. the viewing direction of an observer), the radiance field L is required. It captures the amount of light going through an infinitesimally small area and arriving from an infinitesimally small opening angle around a given direction $\boldsymbol{\omega}$. More precisely, it specifies the change of flux according to a change in direction $\boldsymbol{\omega}$ and a change in surface area that is projected onto a plane perpendicular to direction $\boldsymbol{\omega}$:

$$L(\mathbf{x},\boldsymbol{\omega}) = \frac{\partial\partial\psi(\mathbf{x},\boldsymbol{\omega})}{\partial\boldsymbol{\omega}\partial A^{\perp}(\mathbf{x})} \quad \left[\frac{\mathrm{W}}{\mathrm{srm}^2}\right]$$

The radiance field allows finding the amount of energy going through arbitrary surfaces and arbitrary angles by integration. This yields additional derivative quantities such as the fluence ϕ, which integrates the radiance field over solid angle of the unit sphere at position $\mathbf{x}$:

$$\phi(\mathbf{x}) = \int_{\Omega} L(\mathbf{x},\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \quad \left[\frac{\mathrm{W}}{\mathrm{m}^2}\right]$$

It gives the total power of radiant energy arriving at position $\mathbf{x}$ gathered from all directions. Integrating the radiance field after projecting it onto the planes of the cartesian coordinate system, gives rise to the flux vector $\mathbf{E}$:

$$\mathbf{E}(\mathbf{x}) = \int_{\Omega} L(\mathbf{x},\boldsymbol{\omega})\boldsymbol{\omega}\mathrm{d}\boldsymbol{\omega}$$

The direction of the vector specifies the direction of bulk or dominant energy flow and the magnitude the net flux through an infinitesimally small surface, which is aligned to the direction.

The radiance field L describes the distribution of photon densities in the scene and can easily be used to generate images by integrating it over the sensor surface of a virtual camera sensor and over all directions, from which photons can arrive on the sensor. This results in the energy that the sensor receives over a certain time (flux ψ).

However, the model needs to describe changes in the radiance field L through changes in position. For this purpose, the directional derivative $(\boldsymbol{\omega}\cdot\nabla)L$ is used. It describes the rate of change of the radiance field L, when changing the position an infinitesimal step into direction $\boldsymbol{\omega}$.

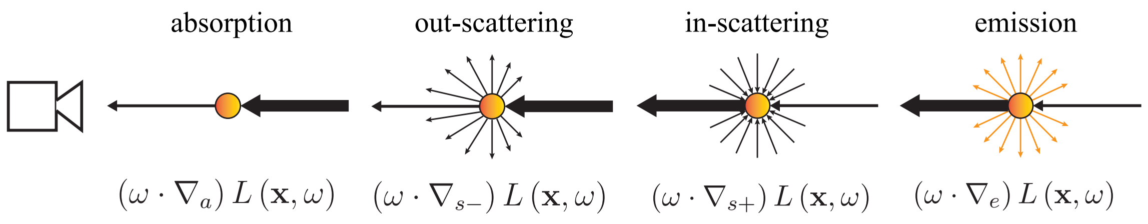

absorption
out-scattering
in-scattering
emission
$(\omega \cdot \nabla_a) L(\mathbf{x},\omega)$
$(\omega \cdot \nabla_{s-}) L(\mathbf{x},\omega)$
$(\omega \cdot \nabla_{s+}) L(\mathbf{x},\omega)$
$(\omega \cdot \nabla_e) L(\mathbf{x},\omega)$

Figure 2.1: *Directional derivatives of the radiance L along direction $\boldsymbol{\omega}$ model the interaction of light with participating media and constitue the radiative transfer equation.*

The radiance field L changes along direction $\boldsymbol{\omega}$ due to different optical phenomena, such as absorption, scattering and emission. Absorption models the effect of radiative energy being transformed into heat or kinetic energy. The amount of absorption depends on the medium and is controlled by the absorption coefficient σ_a. This value combines the number of absorbing particles in the medium as well as the collective surface area of their intersection with a hypothetical plane, which is oriented into the direction of change. The particles are assumed to be spherical or to have a random distribution of orientation. The coefficient controls the loss of energy due to absorption, when making an infinitesimal step along direction $\boldsymbol{\omega}$:

$$\left(\boldsymbol{\omega}\cdot\nabla_a\right)L = -\sigma_a L(\mathbf{x},\boldsymbol{\omega}) \tag{2.1}$$

Scattering describes collision of medium particles. Rather than being absorbed, they change their direction. This is controlled similarly to absorption by the scattering coefficient σ_s. Scattering has an effect on L in two ways. Firstly, most of the scattering photons will change their direction away from the direction along which the rate of change of L, namely outscattering is measured:

$$\left(\boldsymbol{\omega}\cdot\nabla_{s-}\right)L = -\sigma_s L(\mathbf{x},\boldsymbol{\omega}) \tag{2.2}$$

Secondly, scattering affects the way the radiance field changes along $\boldsymbol{\omega}$ in that photons arriving from all directions will collide with the volume and scatter into the direction $\boldsymbol{\omega}$. This is called inscattering:

$$\left(\boldsymbol{\omega}\cdot\nabla_{s+}\right)L = \sigma_s \int_{\Omega'} \rho\left(\mathbf{x},\boldsymbol{\omega}'\cdot\boldsymbol{\omega}\right)L(\mathbf{x},\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}' \tag{2.3}$$

The quantity p is the phase function, a medium parameter, which determines how light is redistributed from an incident direction $\boldsymbol{\omega}'$ to the outgoing direction $\boldsymbol{\omega}$. Since medium particles are assumed to either be spherical or randomly oriented, this function does not change as both vectors $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}$ are rotated, and therefore, the cosine of the angle between the two vectors suffices as a parameter.

Finally, the radiance field changes along direction $\boldsymbol{\omega}$, due to the medium emitting photons itsself into direction $\boldsymbol{\omega}$. This is modelled by a source term Q_e:

$$\left(\boldsymbol{\omega}\cdot\nabla_e\right)L = Q_e(\mathbf{x},\boldsymbol{\omega}) \tag{2.4}$$

All terms combined produce the radiative transfer equation (RTE), which expresses the change of the radiance field L, with respect to an infinitesimal change of position in direction $\boldsymbol{\omega}$ at point $\mathbf{x}$ due to absorption, scattering and emission:

$$\begin{aligned}(\nabla\cdot\boldsymbol{\omega})L(\mathbf{x},\boldsymbol{\omega}) =& -\sigma_t(\mathbf{x})L(\mathbf{x},\boldsymbol{\omega}) \\ & +\sigma_s(\mathbf{x})\int_{\Omega} \rho\left(\boldsymbol{\omega}'\cdot\boldsymbol{\omega}\right)L\left(\mathbf{x},\boldsymbol{\omega}'\right)\mathrm{d}\boldsymbol{\omega}' \\ & +Q_e(\mathbf{x},\boldsymbol{\omega})\ .\end{aligned} \tag{2.5}$$

where $\sigma_t = \sigma_a + \sigma_s$ is the extinction coefficient, which combines absorption and outscattering.

The radiative transfer equation, as presented, is based on important key assumptions about the modelled light transport (D'Eon [21]):

- Linear transport: levels of heat, created by the absorption events don't measurably change the properties of the medium itself. Further photons do not collide with each other and therefore, the transport equation remains linear.
- Steady-state: It is not of relevance to know how light propagates through the scene over time. The typical exposure times of cameras or the human visual system is in a regime, where it is safe to assume that sinks and sources are constant in time and are balanced out in an equilibrium state, such that the radiance field L does not change over time.
- Monoenergetic: often referred to as gray problems. Light is assumed to have a single frequency and color is introduced by solving the single frequency problem multiple times for color space primaries.
- Energy-conserving media: the medium will never scatter more energy at any point, than it receives.
- Scalar radiative transfer: polarization of light is not considered.
- Uncorrelated interaction events: particles have no memory and the probability distribution of an interaction event does not depend on the previous interaction in any way (Moon et al. [68]).
- Isotropic medium: the medium is assumed isotropic, which means that absorption and scattering coefficients are not direction dependent and the phase function only depends on the cosine of the angle between incident and outgoing direction (Jakob et al. [41]).

For notational convenience and for developing the theory for deterministic methods, the radiative transfer equation is expressed in operator notation where transport, collision, and scattering are operators $\mathcal{T}$, $\mathcal{C}$ and $\mathcal{S}$, being applied to the radiance field L:

$$\begin{aligned}\mathcal{T}(L) &= -\mathcal{C}(L)+\mathcal{S}(L)+Q_e \\ \mathcal{C}(L) &= \sigma_t(\mathbf{x})L(\mathbf{x},\boldsymbol{\omega}) \\ \mathcal{S}(L) &= \sigma_s\int_{\Omega'}\rho\left(\mathbf{x},\boldsymbol{\omega}'\cdot\boldsymbol{\omega}\right)L(\mathbf{x},\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}' \\ Q_e &= Q_e(\mathbf{x},\boldsymbol{\omega})\end{aligned} \tag{2.6}$$

Light transport simulation is the process for finding the solutions to L, either for specific values of $\mathbf{x}$ and $\boldsymbol{\omega}$ or globally. The following section gives an overview of the three main categories of methods for solving the radiative transfer equation, of which the last covers deterministic methods and leads into the subject of this thesis. The next section briefly covers analytical solutions which are popular and have been used to great effect. Section 2.3 covers stochastic methods, such as the Monte Carlo method. These methods will be introduced to motivate the need for deterministic methods as a potential means that boosts convergence of stochastic techniques. The last section introduces deterministic methods and covers how discretization of spatial and angular variable leads to a linear systems of equations.

2.2 Analytical Solutions

Analytical solutions to the radiative transfer equation exist for some very basic canonical problems, of which the point source problem is the most popular. It assumes a homogeneous medium (spatially constant σ_t) with an infinite extent. For this problem, a correct analytical solution exists (D'Eon et al. [22]). D'Eon also introduced a very accurate approximation by Grosjean [33], which is simpler to evaluate and convergent. This analytical solution will be used to verify the various methods developed as part of this thesis.

Pegoraro et al. [74] introduced a method based on the analytical solution to the point source problem for application in realtime rendering of single scattered light in participating media (termed airlight integral).

Jensen et al. [42] introduced the diffusion theory and in particular the analytical solution to the diffusion approximation for the point source problem (theoretical treatment in chapter 4.4). Besides, the authors introduced the method of images, which yielded an analytical solution to the half space problem. This problem consists of two different homogeneous media, which are separated by a plane of infinite extend (creating a half space). A solution to the half space problem allowed to locally approximate light transport at the surface of a bounded participating media, using a dipole configuration(termed subsurface scattering). Dipole based analytical models gained popularity in academia and industry and represented a rich and heavily researched branch within rendering. Further seminal work on this subject after Jensen was provided by D'Eon et al. [22] and Habel et al. [35].

The main drawback of analytical models is their restriction to simple problems and homogeneous media in particular. Also, they introduce in some cases significant error, when the canonical problem is used on geometry that violates the plane-parallel assumption. Further dipole methods are based on the diffusion theory, which has poor directional resolution and complications with energy conservation (see section 4.5).

2.3 Stochastic Methods

For most practical applications, analytical solutions do not exist, and numerical integration is required. The key challenge with solving the radiative transfer equation using numerical integration is the global nature of L. Changing a scene parameter in one place will affect all other parts in the scene. This global dependency is introduced by the integral over the radiance field in the scattering term.

Naively integrating the scattering term by using common quadrature rules that are based on interpolation functions on a regular grid (e.g. Riemann integration) becomes prohibitively expensive with one or two recursion levels already as the function evaluations grow exponentially with the number of recursions, known as the *curse of dimensionality*. An alternative to interpolation based-quadrature is Monte Carlo integration based on random samples. This is the core principle behind all stochastic methods.

The following sections provide a very brief overview of standard Monte Carlo methods for various reasons: First, it will help to differentiate deterministic methods, which are introduced later. Second, it explains how the ground truth images used in the result sections of chapter 3, chapter 4.4, and chapter 4 were generated. Furthermore, the introduced methods in this thesis require computation of the single scattered light contribution, for which stochastic methods are used. Last, this section intends to inspire research in combining stochastic methods with deterministic methods into hybrid approaches, combining the benefits of both worlds and by this motivates research into deterministic methods in general.

Distributed Ray Tracing

The invention of Monte-Carlo integration is attributed to Stanislaw Ulam who devised the idea in 1946 during his work on the Manhatten project in the Los Alamos National Lab (Metropolis [63, 64]). Consider a function $f(x)$ for which the following integral over domain $\mathbb{R}$ needs to be found:

$$\int f(x)\mathrm{d}x \tag{2.7}$$

The key idea is to turn the argument x of a function $f(x)$ into a random variable. A stochastic process for randomly choosing instances of x augments each individual value of x with a probability density $p(x)$ and turns the set of all possible x into a measure $\mu(x)$.

With x being a random variable, $f(x)$ becomes a random variable itsself for which statistical moments can be computed. In particular the expected value can be computed by integrating over the measure $\mu(x)$ and weighting the results of f with the probabilty for sampling x:

$$E[f] = \int_{\mu(x)} f(x)p(x)\mathrm{d}\mu(x) \tag{2.8}$$

The probability of sampling x is implicitly given by the function p which in turn is driven by the sampling method chosen. Since sample x represents an infinitely small point in the continuous space $\mu(x)$ a probability can not be assigned directly the same way mass can not be assigned to an infinitely small particle (the particle can always be seen to be smaller than any mass value assigned to it). This is why $p(x)$ returns a probability *density* (probability mass per probability volume) for a specific sample x. Probabilities can be found by integrating probability density p over elements in subsets of $\mu(x)$.

The key idea behind Monte Carlo integration is to turn the computation of the expected value into the computation of the integral in equation 2.7. This is done by introducing a new function g which cancels out the term $p(x)$:

$$E[g] = E\left[\frac{f(x)}{p(x)}\right] = \int_{\mu(x)} \frac{f(x)}{p(x)} p(x)\mathrm{d}\mu(x) = \int f(x)\mathrm{d}x \tag{2.9}$$

The power of Monte-Carlo integration as a numerical algorithm comes with that the expected value – and therefore the sought integral – can be estimated by the arithmetic mean over a set of N random samples of x. This estimate is denoted by the operator $\langle\cdot\rangle_N$:

$$\langle E[g]\rangle_N = \frac{1}{N}\sum_{i=0}^{N} g(X_i) \tag{2.10}$$

This relation is derived from the law of large numbers which states that the average of the results obtained from N number of trials of an experiment approaches its expected value as N increases (see standard textbooks such as Karlos et al. [44]):

$$\lim_{N\to\infty} \langle E[g]\rangle_N = E[g] \tag{2.11}$$

For numerical procedures a finite number of samples is used which introduces some error in the estimate. The estimation error shows up as noise when using independent random samples to compute the radiance field estimates on the camera sensor for each pixel.

Naively using the Monte-Carlo integration technique with the inscattering integral (see equation 2.3) gives rise to distributed raytracing and was developed by Cook

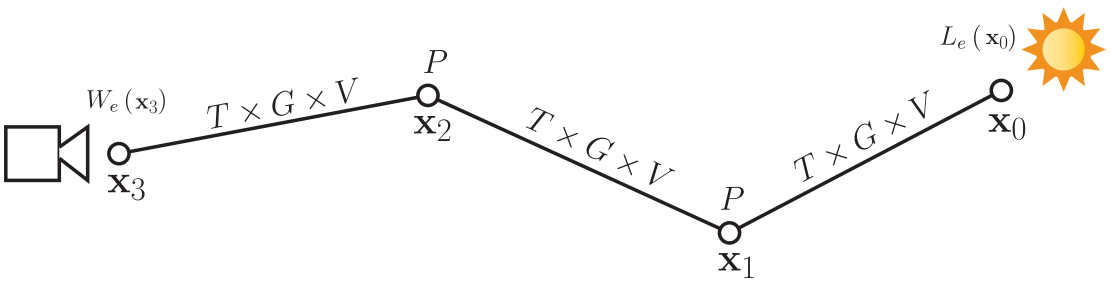


Figure 2.2: *Path sample and its contributing factors to energy loss of radiant energy from the light source to the sensor.*

et al. [16]. It allowed unbiased rendering of complex effects such as motion blur and depth of field but did not address the curse of dimensionality. Recursing the radiative transfer equation would still cause exponential growth of evaluations with each interaction, depending on the number of samples N.

Path Tracing

The rendering equation is an integral equation which describes the propagation of light in scenes without any participating medium. Instead of applying Monte-Carlo directly to the in-scattering integral of the rendering equation and integrating over solid angle, Veach [85] proposed to reformulate the equation as an integral over all possible paths of photons which start from the place of emission (light sources) and arrive at position $\mathbf{x}$ from direction $\boldsymbol{\omega}$ to contribute to $L(\mathbf{x},\boldsymbol{\omega})$. This laid the foundation of the path integral formulation which is the underlying framework used by all Monte-Carlo techniques in rendering today. Pauly et al. [72] then later applied this to the radiative transfer equation which opened up the use of Monte Carlo techniques for light transport simulation in participating media.

The path integral framework introduces the concept of a path $\bar{\mathrm{x}}$ which represents the trajectory a photon can make by traveling from where it was emitted to some position $\mathbf{x}$ with some incident direction $\boldsymbol{\omega}$. The path $\bar{\mathrm{x}}$ consists of N vertices $\mathbf{x}_i$ at which interaction events occur. The function $f(\bar{\mathrm{x}})$ takes such a single path as input and returns the contribution which a photon makes to $L(\mathbf{x},\boldsymbol{\omega})$ after being emitted with initial energy L_e from the light source and traveling along the path trajectory represented by $\bar{\mathrm{x}}$ accounting for all the terms in the radiative transfer equation 2.5 which reduce energy along the way.

Energy is reduced for each segment $\mathbf{x}_i\mathbf{x}_{i+1}$ by the geometry term G which results from the change in the projected area, the transmittance term T which accounts for absorption and outscattering and the visibility term V which accounts for geometry obstructing the path alltogether. At each interior path vertex $\mathbf{x}_i$, scattering is accounted for by multiplying with phase function term $P(\mathbf{x}_{i-1},\mathbf{x}_i,\mathbf{x}_{i+1})$. Finally an importance weight W_e accounts for sensor sensitivity.

$$f(\bar{\mathrm{x}}) = L_e\big(\mathbf{x}_0\big)W_e\big(\mathbf{x}_N\big)\prod_{i=0}^{i=N-1} G\big(\mathbf{x}_i,\mathbf{x}_{i+1}\big)T\big(\mathbf{x}_i,\mathbf{x}_{i+1}\big)V\big(\mathbf{x}_i,\mathbf{x}_{i+1}\big)\prod_{i=1}^{i=N-1} P\big(\mathbf{x}_{i-1},\mathbf{x}_i,\mathbf{x}_{i+1}\big) \quad (2.12)$$

The transmittance is defined as

$$T\big(\mathbf{x}_i,\mathbf{x}_{i+1}\big) = \exp^{-\int_{\mathbf{x}_i}^{\mathbf{x}_{i+1}}\sigma_t(\mathrm{x})\mathrm{dx}} \quad (2.13)$$

and the geometry term is defined as

$$G\big(\mathbf{x}_i,\mathbf{x}_{i+1}\big) = \frac{D\big(\mathbf{x}_i,\mathbf{x}_{i+1}\big)D\big(\mathbf{x}_{i+1},\mathbf{x}_i\big)}{\big\|\mathbf{x}_{i+1}-\mathbf{x}_i\big\|^2}, \quad (2.14)$$

where

$$D(\mathbf{a},\mathbf{b}) = \begin{cases} \big|\mathbf{n}_a\cdot\boldsymbol{\omega}_{ab}\big|, & \text{if } \mathbf{a} \text{ is a surface vertex} \\ 1, & \text{if } \mathbf{a} \text{ is a volume vertex} \end{cases} \quad (2.15)$$

Here $\mathbf{n}_a$ is the normal at the surface vertex $\mathbf{a}$ and $\boldsymbol{\omega}_{ab}$ is the normalized direction vector pointing from vertex $\mathbf{a}$ to vertex $\mathbf{b}$.

The last term is defined in terms of the volume phase function ρ and the surface scattering function ρ_s:

$$P\big(\mathbf{x}_{i-1},\mathbf{x}_i,\mathbf{x}_{i+1}\big) = \begin{cases} \rho_s\Big(\boldsymbol{\omega}_{x_ix_{i-1}}\cdot\boldsymbol{\omega}_{x_ix_{i+1}},\mathbf{n}_i\Big), & \text{if } \mathbf{x}_i \text{ is a surface vertex} \\ \rho\Big(\boldsymbol{\omega}_{x_ix_{i-1}}\cdot\boldsymbol{\omega}_{x_ix_{i+1}}\Big), & \text{if } \mathbf{x}_i \text{ is a volume vertex} \end{cases} \quad (2.16)$$

This thesis exclusively deals with participating media and therefore surface scattering is of no concern.

The set $\mu(\bar{\mathrm{x}})$ is the set of all possible photon trajectories which arrive at $\mathbf{x}$ from direction $\boldsymbol{\omega}$ and start at any light source in the scene. Each sample is assigned a probability density $p(\bar{\mathrm{x}})$ according to a sampling method. Light transport simulation aims to find a solution for $L(\mathbf{x},\boldsymbol{\omega})$. With the path integral formulation this can be expressed as a Monte-Carlo integral over path space:

$$L(\mathbf{x},\boldsymbol{\omega}) = E[g] = \int_{\mu(\bar{\mathrm{x}})} \frac{f(\bar{\mathrm{x}})}{p(\bar{\mathrm{x}})} p(\bar{\mathrm{x}})\mathrm{d}\mu(\bar{\mathrm{x}}) \quad (2.17)$$

This can be estimated using a finite number of samples X_i:

$$\langle E[g]\rangle_N = \frac{1}{N}\sum_{i=0}^{N} g\big(X_i\big) \approx L(\mathbf{x},\boldsymbol{\omega}) \quad (2.18)$$

As mentioned above, the estimation error shows up as image noise (if random samples are independent). It can be shown formally that to reduce the amount of noise in half, four times the number of samples are needed (see Veach [85]). This means that for noise-free results a huge number of samples are needed – a significant drawback of Monte-Carlo methods.

Variance Reduction

Reducing the variance of Monte-Carlo estimators and therefore improving the convergence behavior is the goal of most of the research on stochastic rendering methods. The primary strategy for variance reduction is to reduce the deflection or range of values under the integral sign. This is because the variance is driven by the amount the integrated function value deviates from its mean over the integrated range.

There are two ways to reduce the amount of variation of the integrand around its mean value. First we adjust the sampling strategy such that the probability density function $p(x)$ correlates with $f(x)$ as closely as possible. Ideally, the probability density function is identical to f up to a normalization factor c. This approach is called importance sampling:

$$L = E[g] = \int_{\mu(\bar{x})} \frac{f(\bar{x})}{p(\bar{x})} p(\bar{x}) \mathrm{d}\mu(\bar{x}) \overset{p(x)=cf(x)}{=} \frac{1}{c} \int_{\mu(\bar{x})} p(\bar{x}) \mathrm{d}\mu(\bar{x}) = \frac{1}{c} \tag{2.19}$$

Clearly, such a sampling strategy would have to know the solution L in the first place and therefore is only of theoretical interest. However, if the sampling strategies can account for some factors within f, a significant improvement in variance can be made. Finding good sampling strategies is a vast research branch in its own right within Monte-Carlo based rendering methods. The simplest form is incremental path construction which starts with an initial vertex either on the light source (light path sampling) or the sensor (eye path sampling) and incrementally samples additional points from which additional path segments are constructed. The random decisions are taken according to the terms in f (see equation 2.12). The probability density function is constructed by chaining the conditional probabilities for each decision made along the path. This causes the probability density function to respect individual terms and better correlate with the final contribution.

However, sampling according to individual terms does not respect the product of those terms and therefore is not optimal. This is a continuing subject of research. The basic incremental path sampling was extended by Lafortune et al. [53] and Veach et al. [87] to bi-directional path-tracing. Here two paths are constructed incrementally. One starts at the light source and the second at the camera sensor. From these two paths, multiple individual full paths were constructed and their contribution combined in order to reduce variance. This concept was later generalized by Veach et al. [86] into multiple importance sampling. It allowed combining different Monte-Carlo estimators optimized for different light transport aspect. This spurred an explosion of research into sampling techniques and strategies.

Instead of incrementally sampling local terms such as phase function and geometry term to construct the next path vertex, globally informed sampling techniques appear to be a promising avenue. These techniques work by using an approximation to the radiance field function for importance sampling and therefore produce a better variance reduction than just using local information at each path vertex. Methods differentiate in their caching datastructure (Gaussian-Mixture Models are used by Vorba et al. [89] and an adaptive spatial-directional tree datastructure by Müller et al. [69]) and the way these caches are bootstrapped (update from Monte-Carlo samples during rendering is done by Vorba et al. [89] and reinforcement learning is used by Müller et al. [69]). These methods are a strong motivation for the use of fast approximative solutions from deterministic methods which are the topic of this thesis. Instead of using expensive unbiased Monte-Carlo samples which are then discretized into a cache datastructure, an approximate solution from a deterministic method could be used to quickly learn the skeleton of light transport in the scene and bootstrap the cache.

The second popular variance reduction technique for Monte-Carlo path tracing is based on control variates. Here the idea is to reduce the deflection of the integrand by using a closed form solution C of an approximation $c(x)$ to the function $f(x)$:

$$L = E[g] = \int_{\mu(\bar{x})} \frac{f(\bar{x}) - c(\bar{x})}{p(\bar{x})} p(\bar{x}) d\mu(\bar{x}) + C(\mathbf{x}, \boldsymbol{\omega}) \tag{2.20}$$

$$C(\mathbf{x}, \boldsymbol{\omega}) = \int_{\mu(\bar{x})} c(\bar{x}) d\mu(\bar{x}) \tag{2.21}$$

In the ideal case of $c = f$ the integral becomes zero and the equation reduces to $L = C$. Again this would require a closed-form solution of the original problem which does not exist. However, if c correlates well with f, variance can be reduced. If no closed-form solution for C exists, it can also be integrated numerically using Monte-Carlo integration. Applying the control variates to the integration of C in such a nested fashion yields multilevel Monte-Carlo methods which have not been applied in rendering yet.

Control variates have been used for surface rendering by Mehta et al. [62] and Clarberg et al. [13]. For rendering of participating media, control variates have been applied successfully by Pegoraro et al. [73] where individual samples are used to create and update an adaptive cache of the discretized five-dimensional radiance field. This is very similar to the path-guiding techniques mentioned earlier with the difference that here the guiding cache is used as a control variate instead of a sampling function in the estimator. This work is likewise a good motivation for the deterministic methods developed as part of this thesis. Instead of building the cache from unbiased Monte-Carlo samples and introducing error by discretization into a cache datastructure, an approximation from deterministic methods could be used to bootstrap the cache.

More advanced variance reduction techniques work by better exploring the sample space $\mu(x)$. Markov-Chain-Monte-Carlo based methods start with a sample that has been generated using standard sampling and from there explore the sampling space by perturbation of that single sample. This concept has been further extended with Hamiltonian Monte-Carlo, where newton dynamics are used to generate trajectories which efficiently explore the important regions in high dimensional sample space. This requires sample space gradients which is a challenge due to the discontinuous visibility function. The coverage of these techniques is far beyond the scope of this thesis. The reader is referred to the work by Veach [85] and Cline et al. [14, 15] as a starting point into Markov-Chain-Monte-Carlo and the work by Li et al. [58] as an entry point into the use Hamiltonian Monte-Carlo for rendering.

The growth in computing power and the never ending need for more realistic imagery played into the fact that Monte-Carlo methods much better deal with higher dimensions and can produce an unbiased result. As in other fields, Monte-Carlo methods are now the gold standard for light transport simulation in computer graphics and improving their convergence is the primary research challenge.

The variance reduction techniques have been presented in this section because they show that having an approximation of the radiance function could be used with importance sampling and control variates to reduce the variance of Monte-Carlo estimators. This is a strong motivation for the use of fast approximative solutions from deterministic methods which will be discussed in the next section and are the topic of this thesis.

2.4 Deterministic Methods

Deterministic methods are based on the discretization of functions which depend on the continuous variables $\mathbf{x}$ and $\boldsymbol{\omega}$ into a discrete set of scalar values, that are flattened into a column vector $\mathbf{u}$. The discretization operator in the angular domain is denoted $\mathcal{P}_{\boldsymbol{\omega}}$ and the spatial discretization operator is expressed as $\mathcal{P}_{\mathbf{x}}$

$$\begin{tikzcd} L(\mathbf{x},\boldsymbol{\omega}) \arrow[r, "\mathcal{P}_{\boldsymbol{\omega}}"'] & {} \arrow[r, "\mathcal{P}_{\mathbf{x}}"'] & \mathbf{u} \end{tikzcd}$$

The discretization operators are also applied to the individual operators $\mathcal{T}$, $\mathcal{C}$ and $\mathcal{S}$, which constitute the radiative transfer equation, producing operator matrices T, C

and S respectively:

$$\begin{tikzcd}
\mathcal{T} \arrow[r, "\mathcal{P}_\omega"'] & {} \arrow[r, "\mathcal{P}_\mathbf{x}"'] & T \\
\mathcal{C} \arrow[r, "\mathcal{P}_\omega"'] & {} \arrow[r, "\mathcal{P}_\mathbf{x}"'] & C \\
\mathcal{S} \arrow[r, "\mathcal{P}_\omega"'] & {} \arrow[r, "\mathcal{P}_\mathbf{x}"'] & S
\end{tikzcd}$$

If linear, these discretizations allow to express the application of the radiative transfer operators to the radiance field (see equation 2.6) to be expressed in terms of matrix vector products:

$$\begin{aligned}
T\mathbf{u} &= -C\mathbf{u}+S\mathbf{u}+Q \\
T\mathbf{u}+C\mathbf{u}-S\mathbf{u} &= Q \\
(T+C-S)\mathbf{u} &= Q \\
A\mathbf{u} &= Q
\end{aligned}$$

Deterministic methods turn the problem of solving the radiative transfer equation into the problem of solving a system of linear equations with the solution vector being a discretized version of the radiance field L.

The particular choice of $\mathcal{P}_\mathbf{x}$ and $\mathcal{P}_\omega$ determines the concrete method used. For the spatial discretization $\mathcal{P}_\mathbf{x}$ a common choice in nuclear sciences are finite elements and hexagonal meshes in particular because it allows to best approximate solid geometry (e.g. from reactors) for boundary conditions. However, finite element discretization is unwieldy and difficult to setup. Throughout this thesis, we use a finite difference discretization which is very common in graphics and also easy to work within a practical context.

Spherical Harmonics Methods

Using the spherical harmonics expansion for the discretization of the radiative transfer equation is a very popular approach in other fields, such as nuclear physics or astrophysics. Functions, that depend on the angular variable are turned into a discrete set of spherical harmonics coefficients. Likewise, the radiative transfer equation is turned into a coupled set of partial differential equations through spherical harmonics projection yielding the general P_N-equations. The underlying theory has its origins in astrophysics (see Brunner [8]) and is also popular in nuclear sciences (see Seibold et al. [81]) and medical science (see Frank et al.[28]). In computer graphics P_N-theory was discussed briefly in a paper by Kajia [43] without any details on how to solve the equations. Max [59] pointed out, it is not clear if Kajiya succeeded at all at implementing a method for solving the P_N-equations, as all of the results in his paper were produced with a simpler approach[1]. In the following chapter of this thesis, we derive the real-valued P_N-equations and present a new method for solving them on a finite difference grid.

The P_N-theory has spurred the development of a rich variety of methods of which only the diffusion approximation has been introduced into the field of computer graphics (see Stam [82]). It is derived as a degenerated case of the P_N-equations and gained a lot of popularity in graphics. Wang et al. [90] applied diffusion to the problem of rendering subsurface scattering with heterogeneous participating media using finite elements. This approach was further extended to anisotropic media by Jakob et al. [41] who also used a more complex finite-element discretization in spatial domain. Arbree et al. [2] later in a similar approach addressed various problems in Wang's work and simplified discretization by using a tetrahedral mesh basis. Li et al. [57] used the same tetrahedral representation but a more efficient method for constructing the coefficient matrix which allowed realtime application under changing topology.

Another method, that is derived from the spherical harmonics expansion and can be seen as an extension to the classical diffusion approximation, is flux-limited diffusion. The theory has been invented in the astrophysics domain and is primarily attributed to Levermore et al. [56]. Zhang et al. [93], similar to flux-limited diffusion, uses a combination of diffusion and advection to model light transport. In contrast to flux-limited diffusion, their approach was not derived formally from the radiative transfer equation. The scattering contribution is modelled with a diffusion term and a linear diffusion coefficient. This is not sufficient as we will see. It furthermore is restricted to time-dependent problems and uses a time stepping scheme involving alternating diffusion and advection steps using arbitrary time step and duration parameters. In chapter 4 of this thesis we introduce flux-limited diffusion to the problem of rendering in computer graphics and present a new and efficient method for solving the flux-limited diffusion equation for steady-state radiative transfer.

Although classical diffusion already has been introduced to graphics, it is revisited in chapter 4.4 because its theory is important for the introduction of flux-limited diffusion and the multigrid diffusion solver presented has not been discussed in detail in the literature before.

[1][59] p.4: *"[...] Kajiya attempted to solve these equations for the case of isotropic scattering, but it is unclear whether he succeeded, since all the pictures in [43] were produced by the simpler 'slab' method.[...]"*

Discrete Ordinate Method (S_N-Method)

Instead of using the continuous spherical harmonics basis functions, an alternative approach for the discretization $\mathcal{P}_{\omega}$ is to replace the continuous variable $\boldsymbol{\omega}$ by a discrete set of direction vectors $\boldsymbol{\omega}_i$ (e.g. using Gaussian quadrature points) which causes the inscattering integral (equation 2.3) to turn into a sum and in the end produces a system of linear equations. The variable s is used to identify the direction vector, and the discrete directions are denoted s_i. This is why the method is also referred to as S_N-method (where N is the number of directions). Development of the method is mostly attributed to Wick [92] and Chandrasekhar [12], who derived the S_N-method in one dimension. Later, Carlson et al. [11] extended the method to two and three dimensions. For the one-dimensional case, it has been shown, that S_{N+1}-theory is equivalent to P_N-theory, if Gaussian quadrature points are used (see [17]). In this case, the discrete S_N-directions correspond to the zero crossings of the Legendre polynomials used in P_N. In two or three dimensions however, this correspondence is broken and causes non-physical results, such as the ray effects for which the S_N-method is known for. These are discretization artifacts of the angular variable and appear as unphysical concentrated beams of light within the volume.

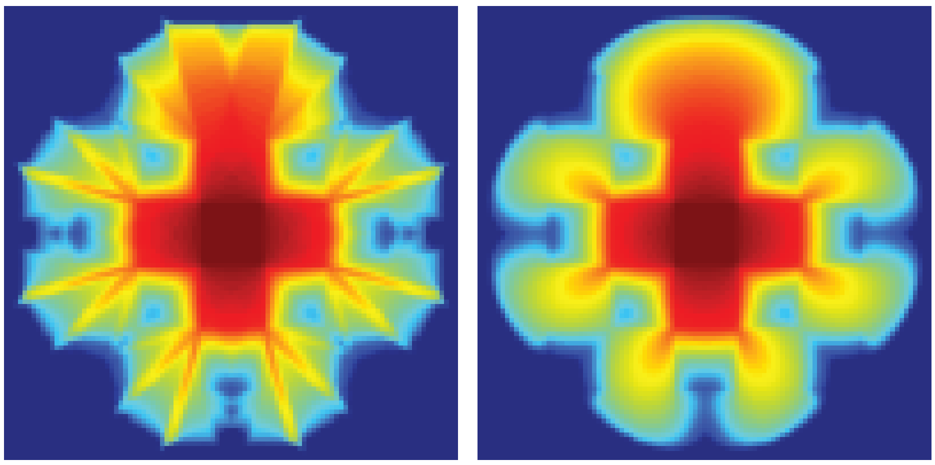


Figure 2.3: *This result from Camminady et al [10] demonstrates the ray effects produced by the discrete ordinate method (left) in comparison to the ground truth (right) for the checkerboard problem. We also simulate the checkerboard problem for our methods (see 3.7.2).*

Like the P_N-method, the S_N method also has not been fully explored in the context of rendering in computer graphics yet. The most extensive investigations thereto were done by by Max [59], who introduced a method for computing light propagation in participating media based on directional bins in a regular grid hence resembling the method of discrete ordinates.

Lattice-Boltzmann Method

The Lattice-Boltzmann method was introduced by Geist et al. [31] as an alternative to finite-element and finite difference methods for solving partial differential equations. The idea is to represent the particle distribution using a spatial grid. Transport is simulated by applying simple update rules to each grid cell in successive iterations. However, the method hardly gained traction, as there is no directional resolution but only photon densities per grid cell.

Heuristic Methods

The necessity of fast computation and the need for realtime performance in graphics gave rise to methods which have not been derived formally from radiative transfer and which take additional assumptions to allow further simplification.

Miller et al. [66] present a method for computing multiple scattering in clouds, that separates the cloud into directly illuminated boundary parts facing the lightsource and a remaining indirectly illuminated part. The method operates in two steps: It first computes the distance function to the directly illuminated part for the whole dataset. This requires solving the Eikonal equation using the fast marching method (see [84]). Then it computes the amount of light reaching each point within the medium as a function of distance. While being a rough simplification, the method allows for fine control through customizable radiance-distance profiles.

Light Propagation Volumes by Kaplanyan et al. [45] were originally developed for surface geometry only and have later been extended to participating media by Billeter et al. [4]. The scene is aligned with a regular grid where each voxel contains a truncated spherical harmonics expansion of the radiance field (similar to P_N). Light emission is computed from single scattered light using reflective shadow maps by Dachsbacher et al. [19]. Indirect light is computed by an iterative method. For each iteration, the light at every voxel is distributed to all voxels in a local neighborhood. The exchange of light is computed for each neighboring voxel taking the projected surface area of voxel faces into account. The process of updating all neighboring voxels within a certain radius, using the light at the center voxel, has some resemblance of wavefronts being emitted from the voxel. Although the technique is not derived directly from radiative transfer, it became very popular with realtime rendering for its convincing results.

The benefits of deterministic methods generally are that they provide a noise-free solution for the complete radiance field L throughout the whole domain. Their main drawbacks are their significant memory requirements for storing the representation

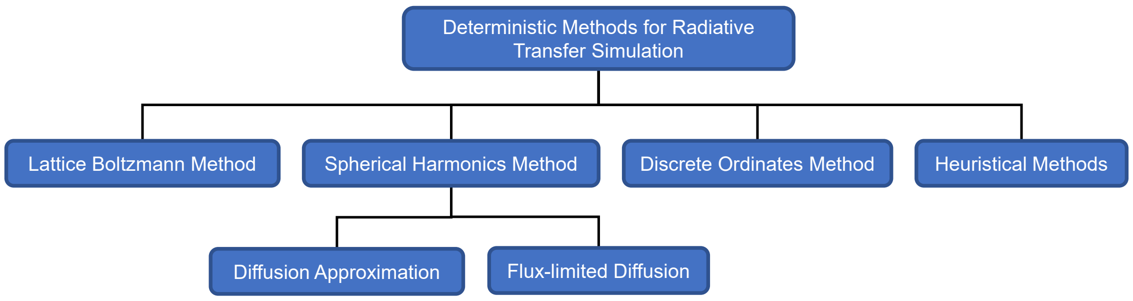

Deterministic Methods for Radiative Transfer Simulation
Lattice Boltzmann Method
Spherical Harmonics Method
Discrete Ordinates Method
Heuristical Methods
Diffusion Approximation
Flux-limited Diffusion

Figure 2.4: *Overview of deterministic methods for light transport simulation.*

of the five-dimensional radiance field function and the discretization error from spatial and angular discretization. Furthermore, deterministic methods tend to not be as general as Monte-Carlo methods and are limited to a subset of problems (e.g. isotropic phase functions). Additionally, some deterministic methods take a lot of time to converge to a solution in certain scenarios.

These drawbacks and the focus on accurate and unbiased Monte-Carlo methods have shifted attention away from deterministic methods lately. However, trends such as the path-guiding techniques discussed in section 2.3 motivate further research on deterministic methods, as they may be useful for bootstrapping guiding caches and therefore could lead to hybrid methods combining the best of both worlds. Consequently, this thesis investigates spherical harmonics methods, the branch of deterministic methods, which has seen most popularity in other domains, and tries to assess their merit for applications in rendering.

Chapter 3

P_N-Method for Multiple Scattering in Participating Media

This chapter is based on the publication P_N-Method for Multiple Scattering in Participating Media [51]

The P_N-method presented in this chapter is a deterministic method for solving the radiative transfer equation. It is based on the spherical harmonics discretization of the radiative transfer equation and its related quantities in the angular domain.

This chapter will give a thorough derivation of the P_N-theory and also devise a method for solving the underlying equations. The first section introduces the spherical harmonics expansion and its important properties. The complex-valued P_N-equations are derived in section 3.2. The fact, that the radiance field is real-valued can be used to cut the number of unknown variables in half. This is done by using the real-valued P_N-equations. In section 3.3, a compact form of the real-valued P_N-equations are derived, which, to the author's best knowledge, has not been given anywhere else in the literature. After a short note on two-dimensional problems in section 3.4, the chapter introduces a new method for solving the P_N-equations (section 3.5). The chapter closes by discussing its integration into a rendering framework, and highlights results (section 3.7).

3.1 Spherical Harmonics

Spherical harmonics (SH) play the role of Fourier series for the sphere and are an important tool for solving numerical problems involving the spherical domain. They are often obtained as eigensolutions to the surface Laplacian, which is the analog to developing the Fourier series as eigensolutions of the operator $(\mathrm{d}/\mathrm{dx})^2$ on a finite line with the boundary conditions that y and dy/dx match at the two ends (see Riley et al. [79]). The result of this derivation are complex-valued functions $Y_{\mathbb{C}}$, which are defined as:

$$Y_{\mathbb{C}}^{l,m}(\theta,\phi) = \begin{cases} C^{l,m} e^{im\phi} P^{l,m}(\cos\theta), & \text{for } m \geq 0 \\ (-1)^m \overline{Y_{\mathbb{C}}^{l|m|}}(\theta,\phi), & \text{for } m < 0 \end{cases} , \tag{3.1}$$

where $P^{l,m}$ are the associated Legendre polynomials. While those can be defined in many different ways, the most numerically robust way to evaluate them is by using the following set of recurrence relations (Press et al. [76]):

$$\begin{aligned} P^{0,0}(\cos\theta) &= 1 , \\ P^{m,m}(\cos\theta) &= (2m-1)!!\left(1-\cos^2\theta\right)^{\frac{m}{2}} , \\ P^{m+1,m}(\cos\theta) &= \cos\theta(2m+1)P^{m,m}(\cos\theta) \\ P^{l,m}(\cos\theta) &= \frac{\cos\theta(2l-1)}{l-m}P^{l-1,m}(\cos\theta) - \frac{l+m-1}{l-m}P^{l-2,m}(\cos\theta) . \end{aligned} \tag{3.2}$$

The factor $C^{l,m}$ in equation 3.1 is defined as

$$C^{l,m} = (-1)^m \sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}} \tag{3.3}$$

Since spherical harmonics are widely used in science, their definition can vary and one has to be careful, when comparing them across literature. This concerns the $(-1)^m$ factor in particular, which is called the Condon-Shortley phase. Sometimes, this factor is part of the definition of the associated Legendre polynomial $P^{l,m}$ and therefore does not appear in $C^{l,m}$. More importantly, the definition of $C^{l,m}$ depends on how the spherical harmonics are expected to be normalized.

Normalization and Orthogonality

The coefficient $C^{l,m}$ has been defined as such that the spherical harmonics function $Y_{\mathbb{C}}$ scales to the unit norm

$$\left\|Y_{\mathbb{C}}^{l,m}\right\| = \sqrt{\left\langle Y_{\mathbb{C}}^{l,m}, Y_{\mathbb{C}}^{l,m}\right\rangle} = 1 . \tag{3.4}$$

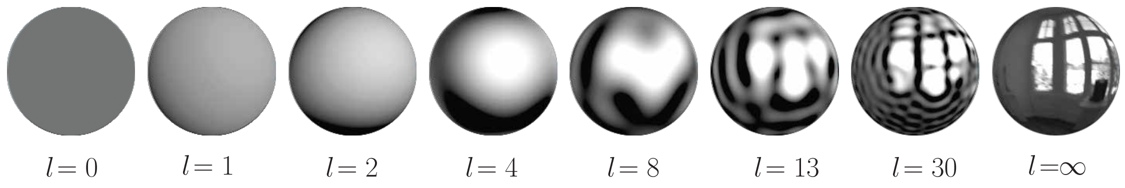


Figure 3.1: *Approximating a spherical function using spherical harmonics with increasing truncation order (from left to right). Like with the Fourier-transform, higher truncation order allows capturing higher frequencies.*

Note that the normalization constant is sometimes chosen differently in the literature (e.g. 4π), producing a different $C^{l,m}$. The unit constant we use here is one of the most common. The spherical harmonics functions $Y_{\mathbb{C}}$ are complex-valued functions on the unit sphere, for which the inner product is defined as:

$$\langle f,g \rangle = \int_{\Omega} f(\boldsymbol{\omega})\overline{g}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{3.5}$$

with $\overline{g}$ being defined as the complex conjugate of g (Folland [27]).

The spherical harmonics functions form an orthogonal family and thus result in:

$$\left\langle Y_{\mathbb{C}}^{l_1,m_1}, Y_{\mathbb{C}}^{l_2,m_2} \right\rangle = 0 \ , \quad \text{for } l_1 \neq l_2 \text{ and } m_1 \neq m_2 \tag{3.6}$$

The properties of normalization (equation 3.4) and orthogonality (equation 3.7) give

$$\left\langle Y_{\mathbb{C}}^{l_1,m_1}, Y_{\mathbb{C}}^{l_2,m_2} \right\rangle = \int_{\Omega} Y_{\mathbb{C}}^{l_1 m_1}(\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l_2 m_2}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} = \delta_{l_1 m_1}\delta_{l_2 m_2} \ . \tag{3.7}$$

Projection and Reconstruction

Because the spherical harmonics are orthonormal, a spherical functional $f(\boldsymbol{\omega})$ can be projected into scalar spherical harmonics basis function coefficients $f^{l,m}$, using the inner product from equation 3.5. For convenience, we define the projection operator $\mathcal{P}$. This operator takes an arbitrary spherical functional and returns its spherical harmonics projection for a given pair of spherical harmonics coefficients l,m:

$$\mathcal{P}^{l,m}(f) = \left\langle f, Y_{\mathbb{C}}^{l,m} \right\rangle = \int_{\Omega} f(\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l,m}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \ . \tag{3.8}$$

Note, the use of the complex conjugate $\overline{Y_{\mathbb{C}}}$ for computing the coefficients $f^{l,m}$. This is due to the definition of the inner product (equation 3.5).

Given the coefficients $f^{l,m} = \mathcal{P}^{l,m}(f)$, the function f can be fully reconstructed by summing up the weighted contributions from the spherical harmonics basis functions

$$f(\boldsymbol{\omega}) \approx \sum_{l=0}^{N} \sum_{m=-l}^{l} f^{l,m}(\mathbf{x}) Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) \,. \tag{3.9}$$

The function can be reconstructed if N approaches infinity. However, as with the Fourier-expansion, it is useful to truncate the expansion at a specific N (giving P_N-method its name). This limits the number of coefficients at the expense of introducing an approximation error by cutting off higher frequencies. The number of sperical harmonics coefficients for truncation order N is $(N+1)\times(N+1)$.

Frequency-invariant Rotation and Rotational Symmetry

Applying a rotation R to the argument of a spherical harmonics basis function of band l results in a linear combination of spherical harmonics basis functions of the same order:

$$Y_{\mathbb{C}}^{l,m}(R\boldsymbol{\omega}) = \sum_{j=-l}^{l} r^{l,j} Y_{\mathbb{C}}^{l,j}(\boldsymbol{\omega}) \,. \tag{3.10}$$

This is true since the spherical harmonics basis functions of order l form an irreducible basis for the group of 3D-rotations (see Corollary 17.17 in [38]). Equation 3.10 implies that a rotation of a function represented in spherical harmonics, does not introduce new or looses any frequencies from or to other spherical harmonic bands and therefore is not subject to aliasing.

The spherical harmonics projection of a rotationally symmetric function f simplifies to Zonal Spherical Harmonics, which are characterized by the fact that only coefficients with $f^{l,0}$ are needed for reconstruction. This property is needed for the derivation of the P_N-equations. In particular, the phase function depends only on the angle between incident direction $\boldsymbol{\omega}_i$ and outgoing direction $\boldsymbol{\omega}_o$, which means that the phase function will be rotationally symmetric around the outgoing direction, if it is fixed.

A rotation $R(\alpha)$ of angle α around the pole axis is expressed in spherical harmonics as:

$$\rho_{R(\alpha)}(Y_{\mathbb{C}}^{l,m}) = e^{-im\alpha} Y_{\mathbb{C}}^{l,m} \,.$$

If a function f is rotationally symmetric around the pole axis, then the following applies:

$$\rho_{R(\alpha)}(f) = f$$

and in spherical harmonics this would be:

$$\sum_{l,m} e^{-im\alpha} f^{l,m} Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) = \sum_{l,m} \rho^{l,m} Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega})$$

By equating coefficients this yields:

$$f^{l,m} = f^{l,m} e^{-im\alpha}$$

Since $e^{-im\alpha} = 1$ for all α only when $m = 0$, the conclusion is that $f^{l,m} = 0$ for all $m \neq 0$. For a function, which is rotationally symmetric around the pole axis, only the $m = 0$ coefficients will be valid.

As mentioned earlier, this will be useful during the derivation of the P_N-equation and its scattering term in particular. If the outgoing direction $\boldsymbol{\omega}_o$ of the phase function at the north pole ($\boldsymbol{\omega}_o = \mathbf{e}_3$) is fixed, then the reconstruction requires exclusively the spherical harmonics coefficients with $m = 0$:

$$\rho(\boldsymbol{\omega}_i) = \sum_{l} \rho^{l0} Y_{\mathbb{C}}^{l0}(\boldsymbol{\omega}_i) \tag{3.11}$$

Recursive Relation

Another property, which will be important for the derivation of the P_N-equations (and its derivative term in particular) is the following recursive relation

$$\boldsymbol{\omega}\overline{Y_{\mathbb{C}}^{l,m}} = \frac{1}{2}\begin{pmatrix} c^{l-1,m-1}\overline{Y_{\mathbb{C}}^{l-1,m-1}} - d^{l+1,m-1}\overline{Y_{\mathbb{C}}^{l+1,m-1}} - e^{l-1,m+1}\overline{Y_{\mathbb{C}}^{l-1,m+1}} + f^{l+1,m+1}\overline{Y_{\mathbb{C}}^{l+1,m+1}} \\ i\left(-c^{l-1,m-1}\overline{Y_{\mathbb{C}}^{l-1,m-1}} + d^{l+1,m-1}\overline{Y_{\mathbb{C}}^{l+1,m-1}} - e^{l-1,m+1}\overline{Y_{\mathbb{C}}^{l-1,m+1}} + f^{l+1,m+1}\overline{Y_{\mathbb{C}}^{l+1,m+1}}\right) \\ 2\left(a^{l-1,m}\overline{Y_{\mathbb{C}}^{l-1,m}} + b^{l+1,m}\overline{Y_{\mathbb{C}}^{l+1,m}}\right) \end{pmatrix}, \tag{3.12}$$

where

$$a^{l,m} = \sqrt{\frac{(l-m+1)(l+m+1)}{(2l+1)(2l-1)}} \qquad b^{l,m} = \sqrt{\frac{(l-m)(l+m)}{(2l+1)(2l-1)}} \qquad c^{l,m} = \sqrt{\frac{(l+m+1)(l+m+2)}{(2l+3)(2l+1)}}$$

$$d^{l,m} = \sqrt{\frac{(l-m)(l-m-1)}{(2l+1)(2l-1)}} \qquad e^{l,m} = \sqrt{\frac{(l-m+1)(l-m+2)}{(2l+3)(2l+1)}} \qquad f^{l,m} = \sqrt{\frac{(l+m)(l+m-1)}{(2l+1)(2l-1)}}$$

This relation implies that a direction vector $\boldsymbol{\omega}$ scaled by a complex SH basis function $Y_{\mathbb{C}}$ of order l,m can be expressed as a vector of complex values basis functions of higher and lower order. Such recursion relations seem to be hard to find in the standard literature and are preserved through citations in relevant articles[1], such as Seibold et al. [81] or Brunner et al. [9]. The signs for the x- and y- component depend on the handedness of the coordinate system, in which the spherical harmonics basis functions are defined.

3.2 Complex-valued P_N-equations

Equipped with key properties of the spherical harmonics expansion, the complex-valued P_N-equations are derived in this section. The derivation follows two basic steps, which are executed separately on each term of the radiative transfer equation. The first step is to substitute each angular dependent radiative transfer quantity, such as the radiance field L, by its (truncated) spherical harmonics projection (equation 3.21). The second step is to apply the spherical harmonics projection to each term of the radiative transfer equation. This results in terms, which will depend on the spherical harmonics indices l,m and therefore form a system of equations. The size of this system is driven by the truncation order N. The higher the value N, the higher frequencies in angular domain are taken into account for an approximation of the radiative transport.

Transport Term

The transport term $\mathcal{T}$ of the RTE is given as

$$(\boldsymbol{\omega}\cdot\nabla)L(\mathbf{x},\boldsymbol{\omega})$$

Replacing L with its spherical harmonics expansion results in:

$$(\boldsymbol{\omega}\cdot\nabla)\left(\sum_{l,m}L^{l,m}(\mathbf{x})Y^{l,m}_{\mathbb{C}}(\boldsymbol{\omega})\right)$$

Next, the whole term is projected into spherical harmonics, which means a multiplication with $\overline{Y^{l'm'}_{\mathbb{C}}}$ and integration over solid angle:

$$P_\Omega(\mathcal{T}) = \int_\Omega \overline{Y^{l'm'}}(\boldsymbol{\omega}\cdot\nabla)\sum_{l,m}L^{l,m}(\mathbf{x})Y^{l,m}_{\mathbb{C}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}$$

The spatial derivative can be extracted to get:

$$P_\Omega(\mathcal{T}) = \nabla\cdot\int_\Omega \boldsymbol{\omega}\overline{Y^{l'm'}_{\mathbb{C}}}\sum_{l,m}L^{l,m}(\mathbf{x})Y^{l,m}_{\mathbb{C}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}$$

[1]according to email exchange with experts from nuclear sciences

Applying the recursion relation (equation 3.12) will result in:

$$\begin{pmatrix} \frac{1}{2}\partial_x \\ \frac{i}{2}\partial_y \\ \partial_z \end{pmatrix} \cdot \int_\Omega \begin{pmatrix} c^{l'-1,m'-1}\overline{Y_{\mathbb{C}}^{l'-1,m'-1}} - d^{l'+1,m'-1}\overline{Y_{\mathbb{C}}^{l'+1,m'-1}} - e^{l'-1,m'+1}\overline{Y_{\mathbb{C}}^{l'-1,m'+1}} + f^{l'+1,m'+1}\overline{Y_{\mathbb{C}}^{l'+1,m'+1}} \\ -c^{l'-1,m'-1}\overline{Y_{\mathbb{C}}^{l'-1,m'-1}} + d^{l'+1,m'-1}\overline{Y_{\mathbb{C}}^{l'+1,m'-1}} - e^{l'-1,m'+1}\overline{Y_{\mathbb{C}}^{l'-1,m'+1}} + f^{l'+1,m'+1}\overline{Y_{\mathbb{C}}^{l'+1,m'+1}} \\ a^{l'-1,m'}\overline{Y_{\mathbb{C}}^{l'-1,m'}} + b^{l'+1,m'}\overline{Y_{\mathbb{C}}^{l'+1,m'}} \end{pmatrix} \sum_{l,m} L^{l,m}(\mathbf{x}) Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega}$$

Integrating the vector term over solid angle, can be expressed as seperate solid angle integrals over each component. The integrals are split into separate terms:

$$\begin{pmatrix} \frac{1}{2}\partial_x \\ \frac{i}{2}\partial_y \\ \partial_z \end{pmatrix} \cdot \begin{pmatrix} c^{l'-1,m'-1}\sum_{l,m} L^{l,m}(\mathbf{x}) \int_\Omega \overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega}) Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} \quad - \quad \dots \\ -c^{l'-1,m'-1}\sum_{l,m} L^{l,m}(\mathbf{x}) \int_\Omega \overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega}) Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} \quad + \quad \dots \\ a^{l'-1,m'}\sum_{l,m} L^{l,m}(\mathbf{x}) \int_\Omega \overline{Y_{\mathbb{C}}^{l'-1,m'}}(\boldsymbol{\omega}) Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} \quad + \quad \dots \end{pmatrix}$$

Applying the orthogonality property to the solid angle integrals will select specific l,m in each term:

$$\begin{pmatrix} \frac{1}{2}\partial_x \\ \frac{i}{2}\partial_y \\ \partial_z \end{pmatrix} \cdot \begin{pmatrix} c^{l-1,m-1}L^{l-1,m-1} - d^{l+1,m-1}L^{l+1,m-1} - e^{l-1,m+1}L^{l-1,m+1} + f^{l+1,m+1}L^{l+1,m+1} \\ -c^{l-1,m-1}L^{l-1,m-1} + d^{l+1,m-1}L^{l+1,m-1} - e^{l-1,m+1}L^{l-1,m+1} + f^{l+1,m+1}L^{l+1,m+1} \\ a^{l-1,m}L^{l-1,m} + b^{l+1,m}L^{l+1,m} \end{pmatrix}$$

Which gives the projected transport term:

$$\begin{aligned} \mathcal{P}_\omega(\mathcal{T}) = & \frac{1}{2}c^{l-1,m-1}\partial_x L^{l-1,m-1} - \frac{1}{2}d^{l+1,m-1}\partial_x L^{l+1,m-1} - \frac{1}{2}e^{l-1,m+1}\partial_x L^{l-1,m+1} \\ & + \frac{1}{2}f^{l+1,m+1}\partial_x L^{l+1,m+1} - \frac{i}{2}c^{l-1,m-1}\partial_y L^{l-1,m-1} + \frac{i}{2}d^{l+1,m-1}\partial_y L^{l+1,m-1} \\ & - \frac{i}{2}e^{l-1,m+1}\partial_y L^{l-1,m+1} + \frac{i}{2}f^{l+1,m+1}\partial_y L^{l+1,m+1} + a^{l-1,m}\partial_z L^{l-1,m} + b^{l+1,m}\partial_z L^{l+1,m} \end{aligned} \tag{3.13}$$

Scattering Term

The scattering term $\mathcal{S}$ in the RTE is given as:

$$\sigma_s(\mathbf{x}) \int_\Omega \rho(\mathbf{x}, \boldsymbol{\omega}' \cdot \boldsymbol{\omega}) L(\mathbf{x}, \boldsymbol{\omega}') \mathrm{d}\boldsymbol{\omega}'$$

The phase function, used in isotropic scattering medium, only depends on the angle between incident and outgoing direction, and therefore is rotationally symmetric around the pole defining axis. This property allows defining a rotation $R(\boldsymbol{\omega})$, which rotates the phase function, such that the pole axis aligns with the outgoing direction vector $\boldsymbol{\omega}$. This rotation is expressed as:

$$\rho_{R(\boldsymbol{\omega})}(\rho) \, ,$$

which can be implemented by applying the inverse rotation $R(\boldsymbol{\omega})^{-1}$ to the arguments of ρ. With this rotated phase function, the integral of the scattering operator can be expressed as a convolution:

$$\begin{aligned}\int_{\Omega'} \rho(\mathbf{x},\boldsymbol{\omega}'\cdot\boldsymbol{\omega})L(\mathbf{x},\boldsymbol{\omega}')\mathrm{d}\boldsymbol{\omega}' &= \int_{\Omega'} \rho_{R(\boldsymbol{\omega})}(\rho)(\cos\theta')L(\mathbf{x},\boldsymbol{\omega}')\mathrm{d}\boldsymbol{\omega}' \\ &= \langle L,\rho_{R(\boldsymbol{\omega})}(\rho)\rangle \end{aligned} \tag{3.14}$$

As the inner product integral of the convolution is evaluated, the phase function rotates along with the argument $\boldsymbol{\omega}$.

Now, the spherical harmonics expansions of L and ρ (equation 3.11) in the definition for the inner product of the convolution (equation 3.14) are used:

$$\langle L,\rho_{R(\boldsymbol{\omega})}(\rho)\rangle = \left\langle \sum_{l,m} L^{l,m}(\mathbf{x})Y_{\mathbb{C}}^{l,m}, \rho_{R(\boldsymbol{\omega})}\left(\sum_{l}\rho^{l0}Y_{\mathbb{C}}^{l0}\right)\right\rangle$$

Due to linearity of the inner product operator, the non-angular dependent parts of the expansions can be pulled out:

$$\langle L,\rho_{R(\boldsymbol{\omega})}(\rho)\rangle = \sum_{l,m} L^{l,m}(\mathbf{x})\left\langle Y_{\mathbb{C}}^{l,m}, \rho_{R(\boldsymbol{\omega})}\left(\sum_{l}\rho^{l0}Y_{\mathbb{C}}^{l0}\right)\right\rangle$$

and further:

$$\langle L,\rho_{R(\boldsymbol{\omega})}(\rho)\rangle = \sum_{l'}\sum_{l,m}\rho^{l'0}L^{l,m}(\mathbf{x})\left\langle Y_{\mathbb{C}}^{l,m}, \rho_{R(\boldsymbol{\omega})}\left(Y_{\mathbb{C}}^{l'0}\right)\right\rangle$$

The rotation $\rho_{R(\boldsymbol{\omega})}$ of a function with frequency l results in a function of frequency l (equation 3.10). In addition the spherical harmonics basis functions $Y_{\mathbb{C}}^{l,m}$ are orthogonal. Therefore the following applies:

$$\left\langle Y_{\mathbb{C}}^{l,m}, \rho_{R(\boldsymbol{\omega})}\left(Y_{\mathbb{C}}^{l'm'}\right)\right\rangle = 0 \qquad \text{for all } \; l \neq l'$$

which further simplifies the inner product integral to:

$$\langle L,\rho_{R(\boldsymbol{\omega})}(\rho)\rangle = \sum_{l,m}\rho^{l0}L^{l,m}(\mathbf{x})\left\langle Y_{\mathbb{C}}^{l,m}, \rho_{R(\boldsymbol{\omega})}\left(Y_{\mathbb{C}}^{l0}\right)\right\rangle$$

What remains to be resolved is the inner product. The spherical harmonics basis functions $Y_{\mathbb{C}}^{l,m}$ are eigenfunctions of the inner product integral operator in the equation above (Dai [20]):

$$\left\langle Y_{\mathbb{C}}^{l,m}, \rho_{R(\boldsymbol{\omega})}\left(Y_{\mathbb{C}}^{l0}\right)\right\rangle = \lambda_l Y_{\mathbb{C}}^{l,m}$$

with

$$\lambda_l = \sqrt{\frac{4\pi}{2l+1}}$$

Replacing the inner product leads to:

$$\langle L, \rho_{R(\boldsymbol{\omega})}(p)\rangle = \sum_{l,m} \lambda_l \rho^{l0} L^{l,m}(\mathbf{x}) Y_{\mathbb{C}}^{l,m}$$

This allows to express the scattering term using SH expansions of phase function p and radiance field L:

$$\begin{aligned} \sigma_s(\mathbf{x}) \int_\Omega \rho(\mathbf{x}, \boldsymbol{\omega}' \cdot \boldsymbol{\omega}) L(\mathbf{x}, \boldsymbol{\omega}') \mathrm{d}\boldsymbol{\omega}' &= \sigma_s(\mathbf{x}) \langle L, \rho_{R(\boldsymbol{\omega})}(p)\rangle \\ &= \sigma_s(\mathbf{x}) \sum_{l,m} \lambda_l \rho^{l0} L^{l,m}(\mathbf{x}) Y_{\mathbb{C}}^{l,m} \end{aligned}$$

However, a spherical harmonics expansion of the term itself has not been done yet. It is still a scalar function that depends on direction $\boldsymbol{\omega}$. The scattering term is projected into spherical harmonics, by multiplying with $\overline{Y_{\mathbb{C}}^{l'm'}}$ and integrating over solid angle $\boldsymbol{\omega}$. All factors that do not depend on $\boldsymbol{\omega}$ will be pulled out, and then the SH orthogonality property is applied:

$$\begin{aligned} &\int_\Omega \overline{Y_{\mathbb{C}}^{l'm'}}(\boldsymbol{\omega}) \sigma_s(\mathbf{x}) \sum_{l,m} \lambda_l \rho^{l0} L^{l,m}(\mathbf{x}) Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} \\ =& \lambda_l \sigma_s(\mathbf{x}) \rho^{l0} L^{l,m}(\mathbf{x}) \sum_{l,m} \int_\Omega \overline{Y_{\mathbb{C}}^{l'm'}}(\boldsymbol{\omega}) Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} \\ =& \lambda_l \sigma_s(\mathbf{x}) \rho^{l0} L^{l,m}(\mathbf{x}) \sum_{l,m} \delta_{ll'} \delta_{mm'} \\ =& \lambda_l \sigma_s(\mathbf{x}) \rho^{l0} L^{l,m}(\mathbf{x}) \end{aligned}$$

Hence we optain the scattering term of the complex-valued P_N-equations:

$$\mathcal{P}_{\boldsymbol{\omega}}(\mathcal{S}) = \lambda_l \sigma_s(\mathbf{x}) \rho^{l0} L^{l,m}(\mathbf{x}) \tag{3.15}$$

Collision and Emission Term

The collision term $\mathcal{C}$ of the RTE is given as:

$$-\sigma_t(\mathbf{x}) L(\mathbf{x}, \boldsymbol{\omega})$$

The radiance field L is replaced with its spherical harmonics expansion:

$$-\sigma_t(\mathbf{x})\sum_{l,m} L^{l,m}(\mathbf{x})Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega})$$

Multiplying with $\overline{Y_{\mathbb{C}}^{l'm'}}$ and integrating over solid angle, after pulling some factors out of the integral and applying the orthogonality identity (equation 3.7) results in:

$$\begin{aligned}&-\sigma_t(\mathbf{x})\sum_{l,m} L^{l,m}(\mathbf{x})\int_\Omega \overline{Y_{\mathbb{C}}^{l'm'}}(\boldsymbol{\omega})Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\ &=-\sigma_t(\mathbf{x})\sum_{l,m} L^{l,m}(\mathbf{x})\delta_{ll'}\delta_{mm'}\\ &=-\sigma_t(\mathbf{x})L^{l,m}(\mathbf{x}) \qquad (3.16)\end{aligned}$$

The derivation of the projected emission term Q is identical to the derivation of the projected collision term. After replacing the emission field Q with its SH projection and multiplying the term with the conjugate complex of $Y_{\mathbb{C}}$ results (after application of the orthogonality property) in:

$$\mathcal{P}_{\boldsymbol{\omega}}(\mathcal{Q})=Q^{l,m}(\mathbf{x},\boldsymbol{\omega}) \qquad (3.17)$$

Final Equation

After putting the projected terms together (equation 3.13, 3.16, 3.15 and 3.17) according to the radiative transfer equation:

$$\mathcal{P}_{\boldsymbol{\omega}}(\mathcal{T})=\mathcal{P}_{\boldsymbol{\omega}}(\mathcal{C})+\mathcal{P}_{\boldsymbol{\omega}}(\mathcal{S})+\mathcal{P}_{\boldsymbol{\omega}}(\mathcal{Q}) \qquad (3.18)$$

we get, as the result, the complex-valued P_N-equations:

$$\begin{aligned}&\frac{1}{2}c^{l-1,m-1}\partial_x L^{l-1,m-1}-\frac{1}{2}d^{l+1,m-1}\partial_x L^{l+1,m-1}-\frac{1}{2}e^{l-1,m+1}\partial_x L^{l-1,m+1}\\ &+\frac{1}{2}f^{l+1,m+1}\partial_x L^{l+1,m+1}-\frac{i}{2}c^{l-1,m-1}\partial_y L^{l-1,m-1}+\frac{i}{2}d^{l+1,m-1}\partial_y L^{l+1,m-1}\\ &-\frac{i}{2}e^{l-1,m+1}\partial_y L^{l-1,m+1}+\frac{i}{2}f^{l+1,m+1}\partial_y L^{l+1,m+1}+a^{l-1,m}\partial_z L^{l-1,m}+b^{l+1,m}\partial_z L^{l+1,m}\\ &=-\sigma_t(\mathbf{x})L^{l,m}(\mathbf{x})+\lambda_l\sigma_s(\mathbf{x})\rho^{l0}L^{l,m}(\mathbf{x})+Q^{l,m}(\mathbf{x},\boldsymbol{\omega}) \qquad (3.19)\end{aligned}$$

3.3 Real-valued P_N-equations

Depending on the truncation value N, the complex-valued P_N-equations will induce a number of spherical harmonics coefficients for every discretized position within the domain. These will be complex and therefore produce two unknowns per coefficient. Moreover, the pipeline from solver to rendering integration will have to deal with complex numbers. This is an unnecessary burden, as there exist only real-valued radiance fields and volumetric quantities in rendering.

If the spherical harmonics projected function or operator is real-valued, then the resulting projection will have a redundant structure, which allows cutting the number of coefficients in half. The real-valued spherical harmonics basis function (denoted $Y_{\mathbb{R}}$) can be defined by projecting the remaining coefficients onto their real and imaginary parts:

$$Y_{\mathbb{R}}^{l,m} = \begin{cases} \frac{i}{\sqrt{2}}\left(Y_{\mathbb{C}}^{l,m} - (-1)^m Y_{\mathbb{C}}^{l,-m}\right), & \text{for } m < 0 \\ Y_{\mathbb{C}}^{l,m}, & \text{for } m = 0 \\ \frac{1}{\sqrt{2}}\left(Y_{\mathbb{C}}^{l,-m} + (-1)^m Y_{\mathbb{C}}^{l,m}\right), & \text{for } m > 0 \end{cases} \tag{3.20}$$

Therefore, the number of spherical harmonics coefficients remains the same, but since those are now real, the number of unknowns is cut in half when compared to complex-valued spherical harmonics.

In this section, the real-valued P_N-equations are derived by using $Y_{\mathbb{R}}$ instead of $Y_{\mathbb{C}}$ for the derivation. The real-valued P_N-equations as such are not new. However, performing the derivation steps manually produces very large equations, which are hard to work with and make it very difficult to see structures and opportunities for simplification. Therefore, in the literature, the real-valued P_N-equations are usually specified as complex-valued P_N-equations in matrix form with an additional transformation matrix, representing the projection to real-valued spherical harmonics (Seibold et al. [81]). An explicit form of the real-valued P_N-equation cannot be found in the literature. An attempt to derive such a form has been made by Frank et al. [29] (appendix A), but their result still requires matrix notation with unwieldy coefficient matrices.

In this thesis a new explicit form of the real-valued P_N-equations is derived, which is very concise and compact. This derivation was made possible by using the computer algebra representation framework which has been developed as part of the solver (see section 3.5.1). The derivation steps for the P_N-equations were carried out using this representation and simplification opportunities that could be discovered much easier, leading to the compact form presented at the end of this section.

Being real-valued implies that no complex-conjugate is needed for the inner product. The projection into real-valued spherical harmonics therefore is

$$f^{l,m} = \left\langle f, Y_{\mathbb{R}}^{l,m} \right\rangle = \int_{\Omega} f(\boldsymbol{\omega}) Y_{\mathbb{R}}^{l,m}(\boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} \ . \tag{3.21}$$

Similarly, the reconstruction is done by using the real-valued spherical harmonics basis function

$$f(\boldsymbol{\omega}) \approx \sum_{l=0}^{N} \sum_{m=-l}^{l} f^{l,m}(\mathbf{x}) Y_{\mathbb{R}}^{l,m}(\boldsymbol{\omega}) \,. \tag{3.22}$$

All properties of the complex-valued spherical harmonics basis functions, except the recursion relation in equation 3.12, carry through to the real-valued basis functions. In particular, the orthogonality property still holds:

$$\left\langle Y_{\mathbb{R}}^{l_1,m_1}, Y_{\mathbb{R}}^{l_2,m_2} \right\rangle = \int_{\Omega} Y_{\mathbb{R}}^{l_1 m_1}(\boldsymbol{\omega}) Y_{\mathbb{R}}^{l_2 m_2}(\boldsymbol{\omega}) \mathbf{d}\boldsymbol{\omega} = \delta_{l_1 m_1} \delta_{l_2 m_2} \,. \tag{3.23}$$

Transport Term

Carrying out the derivation for the transport term $\mathcal{T}$ of the radiative transfer equation produces very different terms compared to the complex-valued counterpart. The derivation is extensive and can be found in appendix A. As the spherical harmonics basis functions are different depending on the sign of m (equation A.16), the projected transport term differs accordingly. After carrying out the derivation steps, the result for $m < 0$ is:

$$\begin{aligned}
&-\frac{1}{2}c^{l-1,m-1}\partial_y L^{l-1,-m+1} + \frac{1}{2}d^{l+1,m-1}\partial_y L^{l+1,-m+1} - \frac{1}{2}\beta^m e^{l-1,m+1}\partial_y L^{l-1,-m-1} \\
&+\frac{1}{2}\beta^m f^{l+1,m+1}\partial_y L^{l+1,-m-1} + \frac{1}{2}c^{l-1,m-1}\partial_x L^{l-1,m-1} \\
&-\frac{1}{2}\delta_{m\neq -1} e^{l-1,m+1}\partial_x L^{l-1,m+1} + \frac{1}{2}\delta_{m\neq -1} f^{l+1,m+1}\partial_x L^{l+1,m+1} - \frac{1}{2}d^{l+1,m-1}\partial_x L^{l+1,m-1} \\
&+a^{l-1,m}\partial_z L^{l-1,m} + b^{l+1,m}\partial_z L^{l+1,m}
\end{aligned}$$

with

$$\beta^x = \begin{cases} \frac{2}{\sqrt{2}}, & \text{for } |x| = 1 \\ 1, & \text{for } |x| \neq 1 \end{cases} \tag{3.24}$$

Following the derivation through for $m > 0$ gives:

$$\begin{aligned}
&\frac{1}{2}c^{l-1,-m-1}\partial_x L^{l-1,m+1} - \frac{1}{2}d^{l+1,-m-1}\partial_x L^{l+1,m+1} - \frac{1}{2}\beta^m e^{l-1,m-1}\partial_x L^{l-1,m-1} \\
&\frac{1}{2}\beta^m f^{l+1,-m+1}\partial_x L^{l+1,m-1} \frac{1}{2}c^{l-1,-m-1}\partial_y L^{l-1,-m-1} - \frac{1}{2}d^{l+1,-m-1}\partial_y L^{l+1,-m-1} \\
&\delta_{m\neq 1}\frac{1}{2}e^{l-1,-m+1}\partial_y L^{l-1,-m+1} - \delta_{m\neq 1}\frac{1}{2}f^{l+1,-m+1}\partial_y L^{l+1,-m+1} a^{l-1,-m}\partial_z L^{l-1,m} \\
&b^{l+1,-m}\partial_z L^{l+1,m}
\end{aligned}$$

Similar simplifications apply to the remaining terms of the spherical harmonics expansion of the transport term for $m = 0$, resulting in the expression:

$$\begin{aligned}&\frac{1}{\sqrt{2}}c^{l-1,-1}\partial_x L^{l-1,1} - \frac{1}{\sqrt{2}}d^{l+1,-1}\partial_x L^{l+1,1} + \frac{1}{\sqrt{2}}c^{l-1,-1}\partial_y L^{l-1,-1}\\ &-\frac{1}{\sqrt{2}}d^{l+1,-1}\partial_y L^{l+1,-1} a^{l-1,0}\partial_z L^{l-1,0} + b^{l+1,0}\partial_z L^{l+1,0}\end{aligned} \tag{3.25}$$

Collision, Scattering and Emission

The remaining terms of the radiative transfer equation are derived in the same way, as their complex-valued counterpart. The collision term for $m < 0$, $m = 0$ and $m > 0$ is

$$\mathcal{P}_{\boldsymbol{\omega}}(\mathcal{C}) = -\sigma_t L^{l,m} \ . \tag{3.26}$$

For the scattering term the equation is

$$\mathcal{P}_{\boldsymbol{\omega}}(\mathcal{S}) = \sigma_s \lambda_l \rho^{l,0}(\mathbf{x}) L^{l,m} \tag{3.27}$$

and finally for the emission term

$$\mathcal{P}_{\boldsymbol{\omega}}(\mathcal{Q}) = Q^{l,m} \ . \tag{3.28}$$

Final Equation

By putting equations 3.25, 3.26, 3.27 and 3.28 together, the result for $m = 0$ is:

$$\begin{aligned}&\frac{1}{\sqrt{2}}c^{l-1,-1}\partial_x L^{l-1,1} - \frac{1}{\sqrt{2}}d^{l+1,-1}\partial_x L^{l+1,1} \frac{1}{\sqrt{2}}c^{l-1,-1}\partial_y L^{l-1,-1}\\ &-\frac{1}{\sqrt{2}}d^{l+1,-1}\partial_y L^{l+1,-1} + a^{l-1,0}\partial_z L^{l-1,0} + b^{l+1,0}\partial_z L^{l+1,0}\\ &= -\sigma_t L^{l,m} + \sigma_s \lambda_l \rho^{l,0} L^{l,m} + Q^{l,m}\end{aligned} \tag{3.29}$$

The equations for $m < 0$ and $m > 0$ can be compressed into a single expression by differentiating the signs using $\pm$. The top sign is associated with $m < 0$ and for $m > 0$ the bottom sign is used. This results in the real-valued P_N-equation for $m < 0$ and

$m > 0$:

$$\begin{aligned}
&\frac{1}{2}c^{l-1,\pm m-1}\partial_x L^{l-1,m\mp 1}-\frac{1}{2}d^{l+1,\pm m-1}\partial_x L^{l+1,m\mp 1}-\frac{1}{2}\beta_x^m e^{l-1,m\pm 1}\partial_x L^{l-1,m\pm 1}\\
&+\frac{1}{2}\beta_x^m f^{l+1,\pm m+1}\partial_x L^{l+1,m\pm 1}\mp\frac{1}{2}c^{l-1,\pm m-1}\partial_y L^{l-1,-m\pm 1}\\
&\pm\frac{1}{2}d^{l+1,\pm m-1}\partial_y L^{l+1,-m\pm 1}\mp\beta_y^m\frac{1}{2}e^{l-1,\pm m+1}\partial_y L^{l-1,-m\mp 1}\\
&\pm\beta_y^m\frac{1}{2}f^{l+1,\pm m+1}\partial_y L^{l+1,-m\mp 1}+a^{l-1,\pm m}\partial_z L^{l-1,\mp m}+b^{l+1,\pm m}\partial_z L^{l+1,\mp m}\\
&=-\sigma_t L^{l,m}+\sigma_s\lambda_l\rho^{l,0}L^{l,m}+Q^{l,m}\ ,
\end{aligned} \tag{3.30}$$

where

$$\beta_x^m=\begin{cases}0, & \text{for } m=-1\\ \frac{2}{\sqrt{2}}, & \text{for } m\neq 1\\ 1, & \text{otherwise}\end{cases}\ ,\quad \beta_y^m=\begin{cases}\frac{2}{\sqrt{2}}, & \text{for } m=-1\\ 0, & \text{for } m\neq 1\\ 1, & \text{otherwise}\end{cases}\ ,$$

and

$$a^{l,m}=\sqrt{\frac{(l-m+1)(l+m+1)}{(2l+1)(2l-1)}}\qquad b^{l,m}=\sqrt{\frac{(l-m)(l+m)}{(2l+1)(2l-1)}}$$

$$c^{l,m}=\sqrt{\frac{(l+m+1)(l+m+2)}{(2l+3)(2l+1)}}\qquad d^{l,m}=\sqrt{\frac{(l-m)(l-m-1)}{(2l+1)(2l-1)}}$$

$$e^{l,m}=\sqrt{\frac{(l-m+1)(l-m+2)}{(2l+3)(2l+1)}}\qquad f^{l,m}=\sqrt{\frac{(l+m)(l+m-1)}{(2l+1)(2l-1)}}$$

$$\lambda_l=\sqrt{\frac{4\pi}{2l+1}}$$

These real-valued results were validated to match the project of the complex version using concrete problem instances.

Throughout this thesis and in this chapter in particular, only these real-valued P_N-equations are used. The next section will cover two-dimensional problems, followed by the introduction of a numerical method for solving these coupled partial differential equations.

3.4 Two-dimensional problems

Working with three-dimensional problems is computationally demanding and makes debugging and visualization difficult. It is common practice to study and validate ideas on two-dimensional problems. The derivation of the two-dimensional P_N-equation is straightforward and will be presented in this section.

A domain in the two-dimensional space $\mathbb{R}^2$ is considered, which is extruded infinitely along the z-axis in both directions into $\mathbb{R}^3$. It shall be further assumed that input fields, such as σ_t, and boundary conditions, also do not change along the z-axis.

Since the domain has infinite extent in z-direction and radiative quantities do not vary along that axis, the radiance field L will also not change along the z-axis at any position within the domain. Further, the upper hemisphere of the spherical radiance function will be a mirrored version of the lower hemisphere, with the equator being the mirror plane. In mathematical terms this implies that L is an even function in the polar angle θ.

From the definition of the associated Legendre polynomials (see equation 3.2) it can be concluded that they are either even or odd according to

$$P^{l,m}(-\cos\theta) = (-1)^{l+m} P^{l,m}(\cos\theta)\ . \tag{3.31}$$

If $l+m$ is odd, then $P^{l,m}$ is odd. This implies that P^{lm} integrates to zero over a symmetric interval around the origin, such as the polar angle range. Since the spherical harmonics basis functions scales P^{lm} only uniformly and along the azimuthal angle, it can be inferred that the spherical harmonics basis functions are odd in the polar angle θ if $l+m$ is odd.

The spherical harmonics projection of the radiance field multiplies the radiance field with the spherical harmonics basis function and integrates it over solid angle. This is done for each spherical harmonics coefficient (l,m-pair) that is required. It has been established that if $l+m$ is odd, the basis function will be odd in θ. Since the radiance field is even in θ, this means that their product will be odd. This implies, that the integral of their product over the polar angle range will be zero.

This in turn leads to the spherical harmonics coefficients of L to be zero for all coefficients where $l+m$ is odd. Further, all occurances of ∂_z in the P_N-equations can be set to zero.

Another way of optaining the two-dimensional equations would be to define the logical equivalent of the RTE for a two-dimensional domain and then transform that with a lower-dimensional Fourier transform. It is not necessarily clear, that this would give the same answer.

3.5 P_N-Solver with Automated Stencil Generation

In this section, a new method for solving the P_N-equations is being given. This method is based on the idea of automatically generating the linear system of equations directly from a computer algeba representation of the P_N-equations. This section will go through the manual steps required to build such a system from general partial differential equations, followed by the motivation for the automated approach taken in this thesis. In subsections 3.5.1 to 3.5.6, the solver and the automation of the individual system-building steps are explained in detail, including the developed software framework and important numerical aspects, such as handling boundary conditions and staggered grids.

There are three reasons why the solver has been designed around the idea of automatic system building. The first reason is that the number of equations of the system depends on the user parameter N (the spherical harmonics truncation order) and is therefore not known in advance during solver implementation. Secondly, the number of equations increases quadratically with truncation order N, which makes manual construction of the system increasingly laborious, time consuming, and error-prone.

The third reason is that there exists a variety of variations of the P_N-method in other fields, which deal with different aspects of P_N-theory, such as reduction of ringing artefacts and improvement of convergence. Such variations include Simplified-P_N (SP_N) [61], Filtered-P_N (FP_N) [78], Diffusion-Correction-P_N (DP_N) [80] and Least-Squares-P_N (LSP_N) [39]. Common to all these variations is that they reduce to a set of coupled partial differential equations. The idea behind the design of the solver in this thesis is to be able to generate the coefficient matrix and right hand side vector directly from the specification of such a coupled system of equations. This would allow easy implementation and cross-comparisons of these alternate theories from the literature.

The P_N-equations have been derived in previous sections by discretizing the continuous angular variable in the radiative transfer equation. This led to a discrete set of coupled partial differential equations, which still depend on the continuous spatial variable $\mathbf{x}$. Therefore, the next step is to discretize this spatial variable on a particular domain with input fields for the various radiative quantities, such as extinction or the phase function. Finally, using this discretization, a system of linear equations can be built by assembling a coefficient matrix A and a right hand side vector $\mathbf{b}$.

We will now summarize the steps for building a linear system from a given partial difference equation (using the two-dimensional Laplace equation as example). The next subsections then will revisit each step and elaborate on how it has been automated as part of the solver.

Spatial Discretization

As mentioned in section 2.4, a discrete representation of the geometry that covers the problem domain is required in order to discretize the spatial variable. Throughout this thesis, a regular Cartesian grid is used, which will be referred to as a finite difference grid. This type of grid is simple and commonly used in graphics. In future it might be worth exploring other complex meshes, such as tetrahedral irregular grids, which are often used in the finite element analysis.

The finite difference grid defines the locations at which radiative transfer quantities and the solution will be specified. It is parameterized by a worldspace bounding box and a resolution $res = (I, J, K)$ or gridspacing $h = (h_x, h_y, h_z)$ in each dimension. The local space of the grid is referred to as a voxelspace.

Using finite difference grids, the discretization consists of replacing quantities, which depend on the continuous variable $\mathbf{x}$ with its discrete counterpart x_{ijk}, where ijk is a coordinate in voxelspace. This coordinate is still continuous and $x_{i+\frac{1}{2}jk}$ is a valid coordinate. Quantities around grid point locations are found by interpolation. In this thesis a trilinear interpolation for three-dimensional grids and bilinear interpolation for two-dimensional grids will be used. Higher order interpolation would be possible and an avenue for future exploration.

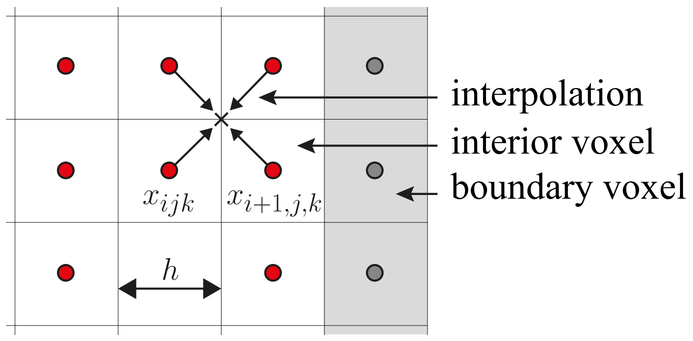


Figure 3.2: *Image of finite difference grid showing domain (white cells), boundary (gray cells), gridpoints (red), voxelsize h and interpolation.*

The first step during a discretization is the replacement of $\mathbf{x}$ by its discrete counterpart x_{ijk}. Using the two-dimensional Laplace equation $\nabla^2\phi(\mathbf{x}) = q(\mathbf{x})$ as an example, this results in:

$$\nabla^2\phi(\mathbf{x}) = q(\mathbf{x}) \xrightarrow{\mathbf{x}=x_{ijk}} \partial_x^2\phi_{ij} + \partial_y^2\phi_{ij} = q_{ij}$$

For notational convenience $\phi_{ijk} = \phi(x_{ijk})$ is defined. The spatial derivatives are approximated using central differences:

$$\partial_x \phi_{ijk} = \frac{\phi_{i+\frac{1}{2}jk} - \phi_{i-\frac{1}{2}jk}}{h_x} \tag{3.32}$$

Continuing with the Laplace example, the substitutions of spatial derivatives are carried out next. Nested derivatives affect the indices accordingly:

$$\begin{aligned} &\xrightarrow{\text{central difference}} \left(\frac{\partial_x \phi_{i+\frac{1}{2}j} - \partial_x \phi_{i-\frac{1}{2}j}}{h_x}\right) - \left(\frac{\partial_y \phi_{ij+\frac{1}{2}} - \partial_y \phi_{ij-\frac{1}{2}}}{h_y}\right) = q_{ij} \\ &\xrightarrow{\text{central difference}} \frac{\frac{\phi_{i+1j} - \phi_{ij}}{h_x} - \frac{\phi_{ij} - \phi_{i-1j}}{h_x}}{h_x} - \frac{\frac{\phi_{ij+1} - \phi_{ij}}{h_y} - \frac{\phi_{ij} - \phi_{ij-1}}{h_y}}{h_y} = q_{ij} \end{aligned} \tag{3.33}$$

Canonical Form

After application of the discretization, equation 3.33 is further brought into canonical form, which means that it is factorized according to the unknowns ϕ_{ij}. This results in the discretized Laplace equation:

$$\xrightarrow{\text{factorization}} \frac{1}{h_x^2}\phi_{i+1j} + \frac{1}{h_x^2}\phi_{i-1j} + \frac{1}{h_y^2}\phi_{ij+1} + \frac{1}{h_y^2}\phi_{ij-1} - 2\left(\frac{1}{h_x^2} + \frac{1}{h_y^2}\right)\phi_{ij} = q_{ij} \tag{3.34}$$

Stencil Code Crafting

Consequently, the canonical form is used to derive the stencil code. This is done by interpreting the equation as a product between a row in the coefficient matrix A and the vector of unknowns (the solution vector $\mathbf{u}$), resulting in the right hand side q_{ij}. The regular structure of the discretization reveals a pattern, which is the same for every row and is called a stencil code.

The purpose of the stencil code is to be executed repeatedly to populate the system matrix A and right hand side $\mathbf{b}$. This requires a mapping of multi-dimensional indices ijk to linear indices into rows of $\mathbf{u}$ (and columns of A). This mapping is referred to as the *index* function. The components of the solution vector $\mathbf{u}$ are associated with unknowns at specific grid locations ijk. In case of the Laplace example this would be written as:

$$u_{\text{index}(ijk)} = \phi_{ijk} \quad \text{with} \quad \text{index} : \mathbb{Z}^3 \mapsto \mathbb{Z} \tag{3.35}$$

```
function applyStencil( A, b, i, j)
    row = index(i, j)
    A[row, index(i+1, j)] = 1/(h*h)
    A[row, index(i, j)] = -2/(h*h)
    A[row, index(i-1, j)] = 1/(h*h)
    ...
    b[row] = q[i, j]

```

Figure 3.3: *Spatial discretization of partial differential equations will lead to stencil code which is executed to populate the system matrix A and right hand side vector b.*

Also functions for accessing discretized input fields (such as q in the Laplace example) and parameters, such as the voxel size h, will be called by the stencil code to retrieve required values.

System Matrix Population and Solve

The stencil code is compiled once and then executed for each row in A to populate the system. Different standard methods for solving the linear system $A\mathbf{u} = \mathbf{b}$ can be applied to find the solution $\mathbf{u}$, the discretized radiance field. This solution can then be used in a rendering application.

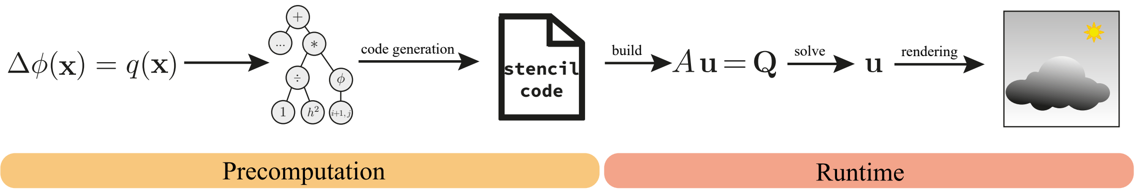


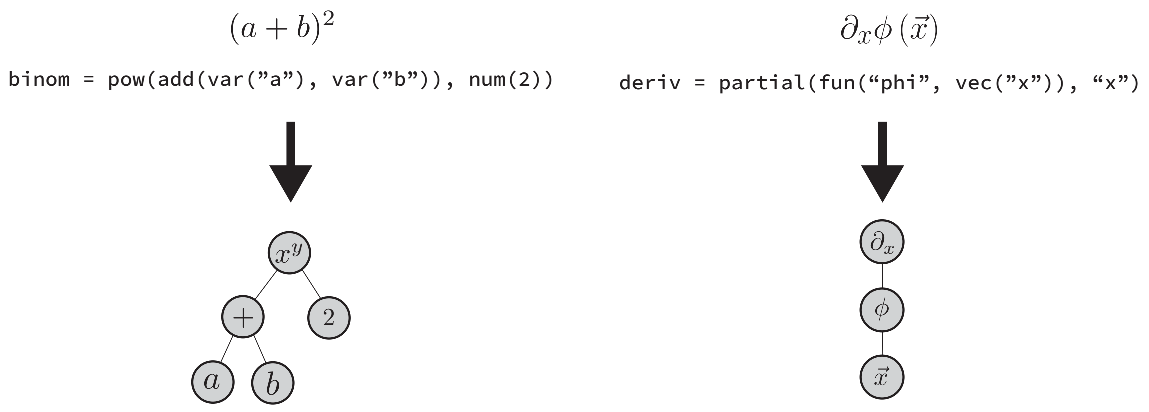
Figure 3.4: *Overview of the method for solving the P_N-equations which is introduced in this chapter. The core idea is to automate the stencil code generation using a computer algebra representation of the equations.*

In the following subsections, the new solver will be introduced as as a general framwork for automization of the steps for system building. The first subsection discusses how a computer algebra representation is used to represent the equation to be solved. Subsection 3.5.2 shows how the spatial discretization and stencil code generation are modeled as operations on that representation. Subsection 3.5.4 explains the aspects of the system that are related to boundary conditions, followed by subsection 3.5.5 which discusses the surrounding framework and how the generated stencil is used within that framework. Subsection 3.5.6 elaborates on the extension of the solver to staggered grids, which is necessary for the solver to produce useful results. Finally, subsection 3.5.7 is dedicated to properties of the coefficient matrix A and methods for solving the linear system.

3.5.1 Computer Algebra Representation

At the core of the solver is a computer algebra representation of the equations, which the system is trying to solve on a given domain. This representation is a mathematical expression tree (Preiss [75]). Each node within that tree can have any number of children and represents a constant, mathematical symbol or a mathematical operation, such as integration, derivation, power, addition or multiplication. Representations of complex equations are constructed by nesting nodes into a larger tree, where child nodes are considered as arguments of the operation, represented by the parent node.

A number of computer algebra systems exist, such as SymPy [65], which allow to generate and manipulate computer algebra representations. However, for this thesis a small lightweight computer algebra representation containing the required subset has been developed. Figure 3.5 demonstrates how this system is used to build the expression trees for the examples above.



$(a+b)^2$

```
binom = pow(add(var(”a”), var(”b”)), num(2))
```

$\partial_x \phi(\vec{x})$

```
deriv = partial(fun(“phi”, vec(”x”)), “x”)
```

Figure 3.5: *An Python-API is used to create the expression tree representation for mathematical expressions.*

Manipulations of mathematical expressions, such as expansions, the application of identities, substitution, factorization or rearranging of terms can be expressed as operations on the expression tree (see figure 3.6). Section 3.5.2 will assess how the spatial discretization is carried out as such an operation.

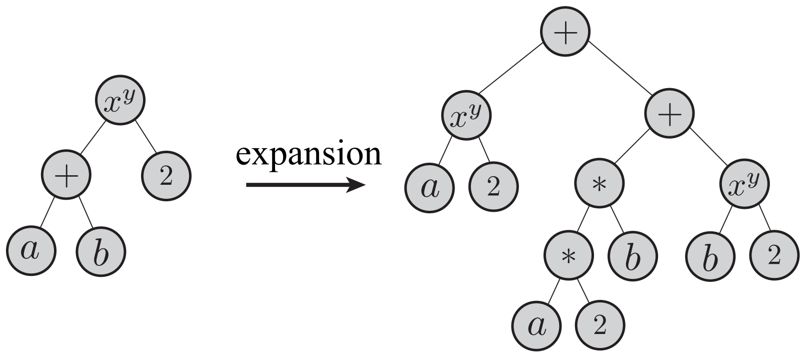


Figure 3.6: *Algebraic manipulations of equations are expressed as operations on the expression tree given by the computer algebra representation.*

3.5.2 Discretization

After expressing the P_N-equations in the computer algebra representation, the spatial discretization is carried out as a manipulation step on the mathematical expression tree. This is done by parsing the tree from the root. The equation is supposed to be discretized at position $\mathbf{x}$, which initially coincides with the location x_{ijk} in the discretized domain. This location is kept on a stack by the parser.

When a differential operator is encountered during tree parsing, the subtree representing the expression to be derived is instantiated twice according to the finite difference expression in equation 3.32 with appropriate weighting factors. The current position on the stack is adjusted and pushed on the stack again when descending down in the tree on each side of the central difference expression. This will produce higher order stencils for nested differential operators.

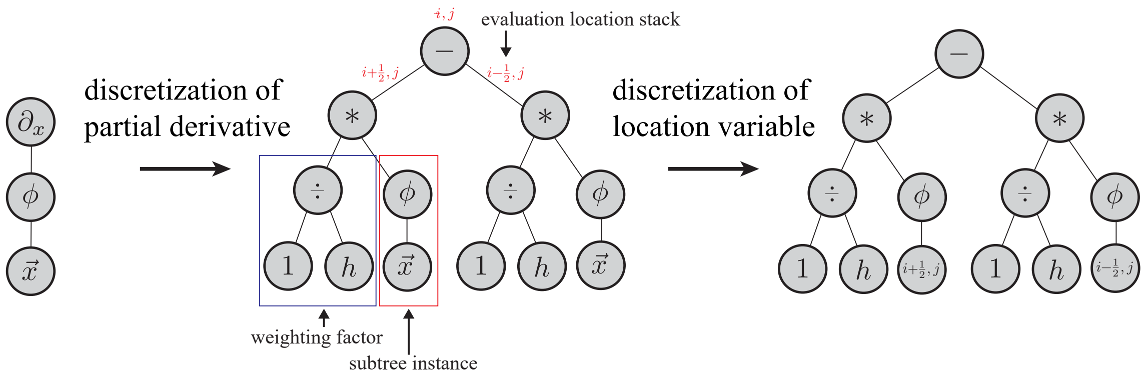

discretization of partial derivative
discretization of location variable
evaluation location stack
weighting factor
subtree instance

Figure 3.7: *Discretization is carried out as an operation on the expression tree where partial derivatives are replaced by finite difference subtrees. Continous variables are replaced by discrete counterparts, for which offsets are retrieved from a stack that is maintained during tree parsing.*

Once the parser arrives at the symbol **x** in the tree, it is replaced by its discrete counterpart. The top of the discrete position stack indicates where the unknown is expected to be evaluated relative to position ijk, at which the whole equation is being evaluated. If the position on top of the stack is the same, then the unknown at ijk is used to replace **x** in the expression. If the unknown is expected to be evaluated at a different location than ijk (due to central differences), then **x** is being substituted by an expression which expresses the interpolation of the unknowns from the gridpoints at which they are defined for the position on top of the stack (figure 3.8). This ensures, that the final expression only contains unknowns, for which an element in the solution vector exists. This will automatically handle the introduction of staggered grids later in section 3.5.6.

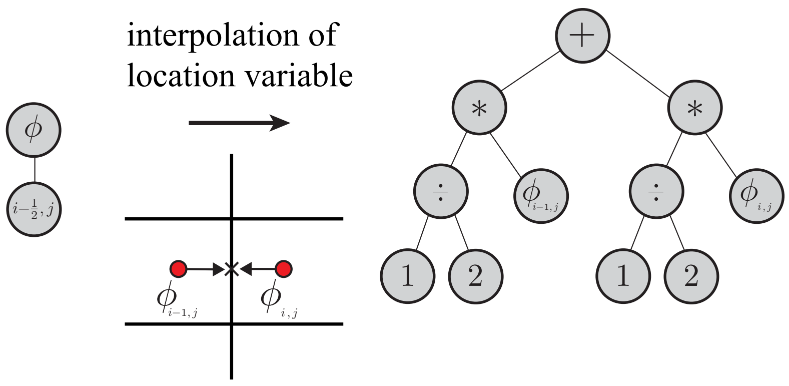
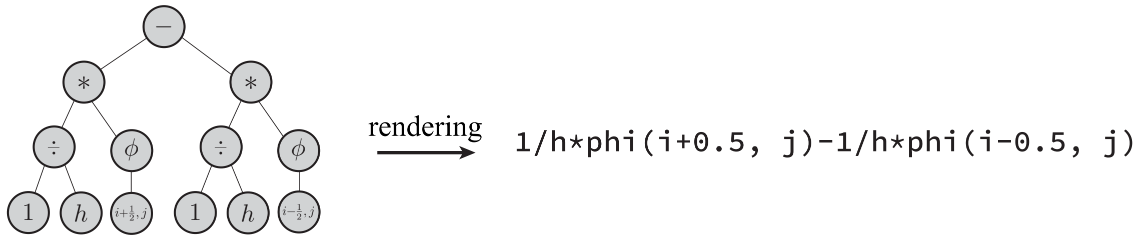


Figure 3.8: *Continuous field variables are turned into discrete counterparts by replacing them with an expression subtree which realizes an interpolation.*

After applying the discretization to the expression tree, the equation is being factorized according to the unknowns (ϕ in the Laplace example). The factorization is applied as a sequence of manipulation operations to the expression tree, including application of the distributive law. A seperate step iterates over all resulting terms and merges coefficients from multiple instances of the same unknown, so that each unknown appears only once (canonical form).

3.5.3 Stencil Code Generation

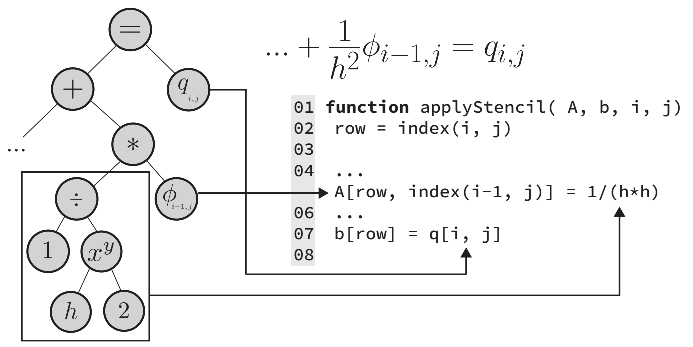
The expression tree of the canoncial form is analyzed and rendered into stencil code. This is facilitated by different rendering frontends which allow for rendering the expression tree to latex or C++-code (see figure 3.9).



```
1/h*phi(i+0.5, j)-1/h*phi(i-0.5, j)
```

Figure 3.9: *Rendering frontends allow turning expression trees into source code.*

If a term contains an unknown, then its factor expression subtree is rendered into a source code string. The symbols for voxel size and other radiative quantities are replaced by function calls into an API, which is being provided by the framework. Finally, the voxel-space coordinate associated with the unknown is used to find a discrete index into the finite difference grid, relative to the original coordinate ijk, at which the equation is being evaluated. This discrete index is being used to compute the column in A. Hence the assignment expression in the stencil code can be put together.



```
function applyStencil( A, b, i, j)
 row = index(i, j)

 ...
 A[row, index(i-1, j)] = 1/(h*h)
 ...
 b[row] = q[i, j]

```

Figure 3.10: *The discretized equation in canonical form allows straightforward generation of stencil code from the different parts of the expression tree.*

If the term does not contain an unknown, the whole term is rendered into a single expression, which is being assigned to the current row of the right hand side vector **b**.

The stencil code is generated with the assumption of certain function calls being available. These are integrated into a single API which is being provided by the solver framework (figure 3.11). This API allows the stencil to query the current row in the system for which it is being executed, the voxelsize as well as functions for accessing discrete radiative transfer properties.

```

V3i getVoxelCoord(); // returns voxel coordinate for which stencil is currently run
Domain& getDomain(); // returns domain information (bounding box, resolution, voxelsize)
Scene& getProblem(); // returns information on radiative quantity fields, emission fields etc.
V3d evalScattering( const V3i& offset ); // returns scattering coefficient at shifted voxellocation
V3d evalAbsorption( const V3i& offset ); // returns absorption coefficient at shifted voxellocation
V3d evalExtinction( const V3i& offset ); // returns extinction coefficient at shifted voxellocation
...
MatrixAccessHelper coeff_A( int coeff_i, V3i& voxel_j, int coeff_j ); // access matrix entry in A
MatrixAccessHelper coeff_b( int coeff_i ); // returns access to the right hand side entry in b
...
```

Figure 3.11: *Stencil code generation assumes an API for querying voxel fields of radiative quantities and accessing entries in coefficient matrix A and right hand side b*

3.5.4 Boundary Conditions

Central difference discretization will introduce references to spherical harmonics coefficients, which are outside the computational domain. Those references will be part of the stencil code, which has been generated without the notion of boundaries and boundary conditions and are handled transparently by the solver framework.

Boundary conditions define how the problem behaves directly on the boundary interface of the domain, and are an important aspect of establishing the system of linear equations. Boundary conditions can be specified without effort through the Dirichlet boundary conditions and Neumann boundary conditions, which are both supported by the system introduced in this chapter.

If $D \in \mathbb{R}^3$ defines the volume of the computational domain across which to find a solution to a partial differential equation, then $\partial D \in \mathbb{R}^3$ can be the set of points, which lie directly on the boundary of that domain. With Dirichlet boundary conditions, the value of the unknown is specified at the interface ∂D directly. With Neumann boundary conditions, the behavior of the solution at the boundary is specified by giving a normal derivative of the solution. Using the Laplace example, this would be:

$$\text{PDE:} \quad \nabla^2\phi(\mathbf{x}) = q(\mathbf{x})$$

$$\text{Dirichlet BC:} \quad \phi(\mathbf{x}) = f(\mathbf{x}) \qquad \forall \mathbf{x} \in \partial D$$

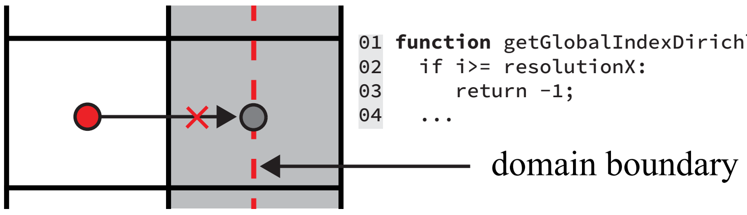
$$\text{Neumann BC:} \quad \frac{\partial \phi(\mathbf{x})}{\partial \mathbf{n}} = g(\mathbf{x}) \qquad \forall \mathbf{x} \in \partial D$$

In most applications for computer graphics, a value of zero for either Dirichlet or Neumann boundaries is adequate and is handled by the system through remapping the index: Considering the first term of the discretized Laplace equation in equation 3.34, the result is the following assignment within the stencil code:

$$A[\mathrm{index}(i,j),\mathrm{index}(i+1,j)] = 1.0/(h_x*h_x); \tag{3.36}$$

The function call $index(i+1,j)$ will not be able to produce a valid column index into the coefficient matrix A for voxels adjacent to the boundary, because the unknown ϕ has no discrete element in the solution vector voxels outside the domain. If the value of ϕ on the boundary is zero, then the unknown on the boundary voxel would be zero and its coefficient would not matter. Therefore, the simplest solution is to ignore the coefficient not writing anything into the matrix A. This equates to matrix A being zero outside the computation domain.

In the solver framework, this is implemented by overloading the assignment operator. If the target coefficient of the assignment belongs to a boundary voxel and Dirichlet boundary conditions are enabled, then the assignment is simply ignored and no change to the system matrix A is done.



```
function getGlobalIndexDirichlet(i, j)
  if i>= resolutionX:
     return -1;
  ...
```

Figure 3.12: *Dirichlet boundary conditions with zero value are modelled by ignoring coefficient assignments to boundary voxels.*

With Neumann boundary conditions, the derivative is specified at the boundary. Since the derivative is found using central differences between two adjacent voxels, the boundary condition is controlled by how the coefficient at the voxel near the boundary relates to its adjacent coefficient at the boundary voxel. In the simplest case, the derivative is set to zero, in which the boundary voxel coefficient needs to be identical with its adjacent voxel inside

$$\partial\phi = 0 \quad \forall\phi \in \partial D \implies \frac{\phi_{i+1,j}-\phi_{i,j}}{h_x} = 0 \implies \phi_{i+1,j} = \phi_{i,j}\,.$$

In the case of adjacent coefficients being equal across the boundary interface, a coefficient b assigned to the boundary $\phi_{i+1,j}$ can be added to the adjacent unknown inside the domain resulting in:

$$a\phi_{i,j}+b\phi_{i+1,j} = (a+b)\phi_{i,j} \quad \text{with} \quad \phi_{i+1,j} = \phi_{i,j} \tag{3.37}$$

In the solver framework, this is implemented by index remapping. If the target coefficient of the assignment belongs to a boundary voxel and Neumann boundary conditions are enabled, then the index function returns the index of the adjacent voxel inside the domain.

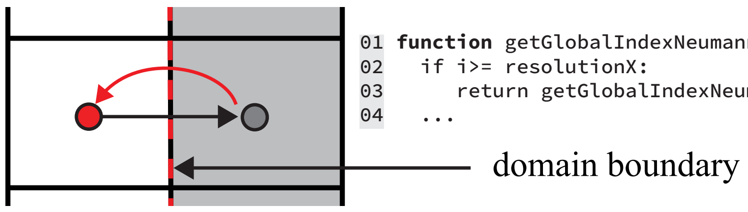

```
function getGlobalIndexNeumann(i, j)
  if i>= resolutionX:
     return getGlobalIndexNeumann(i-1, j);
  ...
```

Figure 3.13: *Neumann boundary conditions with zero value are modelled by remapping indices towards the adjacent inner voxel when assigning coefficients.*

Note how the choice and implementation of the boundary condition has an effect on where the effective boundary interface of the domain is located. With Dirichlet boundary conditions, the interface is directly located at the voxel centers of boundary voxels, while with Neumann boundary conditions, the interface is located at directly between the boundary voxels and inner voxels (figure 3.12 and figure 3.13).

3.5.5 Solver Framework

The stencil code is the result of the automated discretization step, which is carried out on the computer algebra representation and rendered into source code. The resulting stencil code is compiled and linked with a solver framework, which runs it to propagate the system matrix A and right hand side vector $\mathbf{b}$. By rendering the tree to a different representation and assuming a different API used by the stencil code, different solver frameworks can be supported. Therefore, the details about the surrounding framework are not relevant to the method in this thesis. In this section, however, an outline of the framework, which was implemented as part of this thesis will be given.

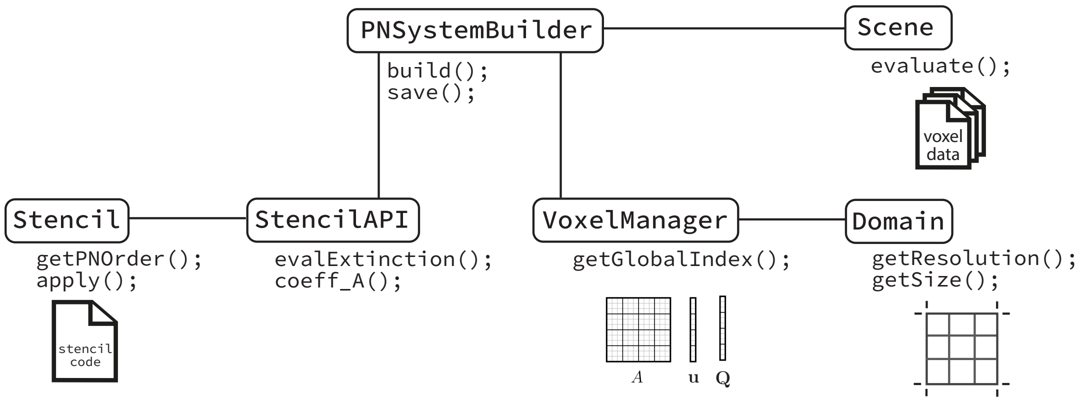


Figure 3.14: *High level overview over the P_N-solver architecture.*

The *Stencil* class is the key structure around which the system is designed, and which serves as a wrapper around the generated stencil code. It specifies the dimension (two- or threedimensional) and the truncation order N, as these parameters are being decided during stencil code generation. The *Stencil* class also indirectly specifies the number of spherical harmonics coefficients per voxel. It further specifies the placement of each spherical harmonics coefficient within the voxel. This will become important when staggered grids are introduced. The stencil specifies the highest order of any derivative operator, which was found during discretization. This affects the number of layers of boundary voxels around the computational domain. Finally, the *Stencil* class has an apply function, which takes the reference to a *StencilAPI* object as an input and runs the generated stencil code that uses that API to propagate the system matrix A and right hand side vector $\mathbf{b}$.

The *StencilAPI* class serves as the single interface between the generated stencil code and the surrounding parts of the framework. Figure 3.11 gives an overview of the different query functions offered by the API. Essential are functions for mapping between coordinate spaces, such as world-space, voxel-space, discrete grid coordinates ijk and linear indices into the solution vector. Further, those functions allow querying radiative transfer quantities, such as extinction coefficient or phase function spherical harmonics coefficients at world space positions. Both groups of functions will make use of a reference to a *VoxelManager* and a *Scene* class, respectively.

The *Scene* class contains all inputs required to specify the problem to be solved. A *Domain* object describes the extent of the computational domain in world-space, as well as the resolution of the finite difference grid. Further attached are references to abstract *Field* interfaces, which allow querying scalar or vector fields at arbitrary worldspace positions for extinction coefficient, albedo, and the spherical harmonics coefficients of phase function and emission field. This is the place where the problem description is provided as input to the system. Different implementations of these abstract interfaces can be used, such as *Constant* fields or *VoxelGrid* fields. This follows the idea of resolution independent volumes introduced by Tessendorf et al. [83].

The resolution of the finite difference grid provided by the *Domain* class and the number of spherical harmonics coefficients by the *Stencil* class, impose the layout of the voxel grid on the system. The *Stencil* class gives the highest order of the derivation operator found during discretization, and with that drives the number of layers of boundary voxels. The final system of linear equations has to be a single column solution vector with a coefficient matrix of compatible dimensions. Since the computational domain is discretized using grid coordinates ijk, a mapping from three-dimensional coordinates ijk and spherical harmonics coefficient index c to a linear global index needs to be provided. This task becomes more involved, when staggered grids are introduced. This bookkeeping of interior and boundary voxels and their contained spherical harmonics coefficients is performed by the *VoxelManager* class. This class is initialized with the information about the grid resolution by the *Domain*, number of boundary voxel layers, number of coefficients, and the position of each coefficient on the grid by the *Stencil* class. It manages the memory and mapping from multidimensional grid indices to linear indices within the global system.

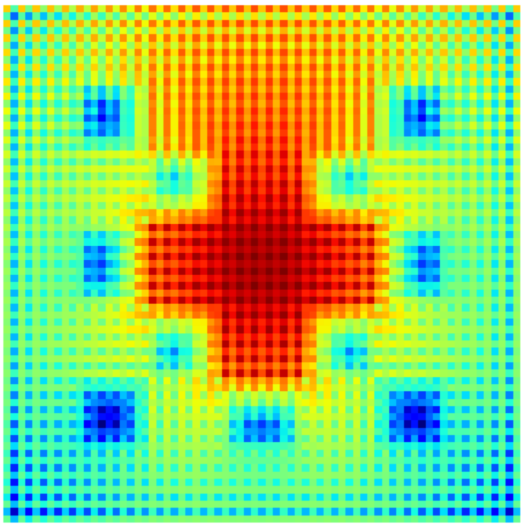
Finally, the *PNSystemBuilder* class is the structure, which brings the different parts together. It is initialized wit the *Scene* class and the *Stencil* class. Both are provided by the client code and are used during initialization to set up the *VoxelManager*, which provides the layout and setup for the solution vector $\mathbf{u}$, right hand side vector $\mathbf{b}$ and system matrix A. The purpose of the *PNSystemBuilder* class is to build the system matrix A and right hand side vector $\mathbf{b}$ and return both to client code. This is done by calling the *build* function on the object.

The *build* function iterates over all voxels. For each voxel, the *StencilAPI* object is set up with the correct indices and data pointers and passed as an argument to a call of the *apply* function of the *Stencil* class. As explained in section 3.5.2, the stencil code will make appropriate calls to the *StencilAPI* object, which will propagate the system matrix A and right hand side vector $\mathbf{b}$ with the correct values for the row, corresponding to the current voxel in the cycle of the *build* function. The system is fully propagated after the stencil has been executed for every voxel. Separate functions of the *PNSystemBuilder* class allow querying the system matrix A and right hand side vector $\mathbf{b}$ after those have been build.

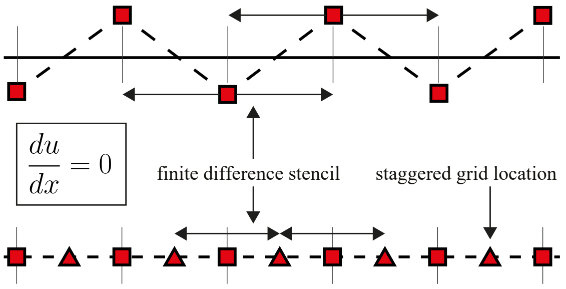
3.5.6 Staggered Grids

In figure 3.15a, preliminary results for the two-dimensional checkerboard problem are shown, a popular reference problem for deterministic methods (section 3.7). As seen in the figure, it suffers from severe artifacts, showing an oscillating structure. The cause for these artifacts is a known problem for partial differential equations, which have different quantities influencing each other at the same spatial grid locations. The problem is visualized in figure 3.15b, using a simple one-dimensional grid as an example. Shown is a simple setup with two variables, which are located at the same grid points, referred to as collocated grids. The variable β depends on the derivative of α. Due to central difference discretization, two different solutions that are decoupled from each other can emerge. These solutions are interleaved in an oscillating pattern throughout the grid as shown in the upper part of the figure.

The solution to this problem is to offset the coefficients α and β from each other spatially. This configuration is called staggered grids. As visualized in figure 3.15b bottom, setting up the coefficients in that way prevents decoupling and results in a



(a) *P_1-solution of the checkerboard problem using collocated discretization grid.*



(b) *Example showing the problem with finite differences producing a false solution (top) and its treatment with staggering (bottom).*

Figure 3.15: *Artefacts in the solution (left) due to shortcomings of collocated grid locations. These are addressed using staggered grids (exemplified right).*

single coherent solution throughout the grid. A similar staggering of the interdependent variables is required for the P_N-solver to produce correct results.

The staggered grid locations are found by subdividing the original grid into a finer grid of double resolution. The locations are at the grid points of that finer grid. In terms of the original grid, these are located at the voxel centers, the center of the voxel faces between adjacent voxels, the center of voxel edges and at voxel corners as shown in figure 3.16.

With a staggered grid configuration, the question is which unknowns (the spherical harmonics coefficients) are placed at which staggered grid location. Seibold et al. [81] analysed the structure of the transport term and found that its discretization matrix relates coefficients to their derivatives in a very particular structure after converting the matrix to real-valued spherical harmonics. If coefficients are located on the staggered grid according to this structure, hence no interpolation is required when evaluating central differences of the stored coefficients (see figure 3.17).

Assuming this structure, the unknown locations are found by the following algorithm: First the zero coefficient is placed at the voxel center. Then all coefficients are found, which depend on derivatives of the zero coefficient. These coefficients are placed according to the central difference offsets. Following this, the coefficients are found, which depend on derivatives of the recently placed coefficients. The central difference

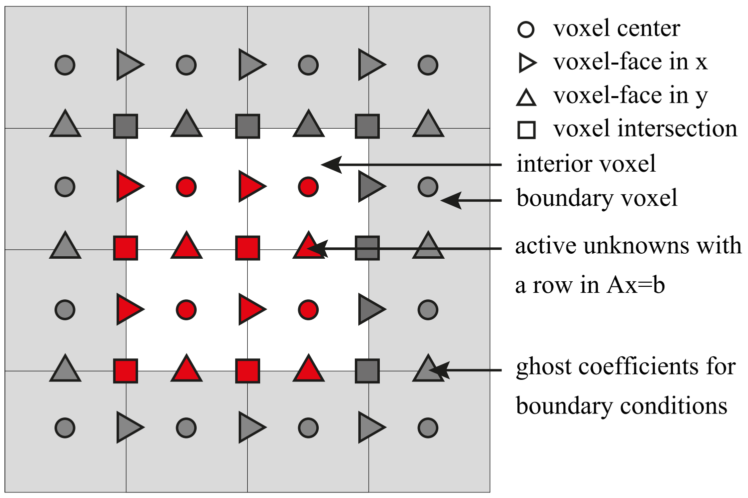


Figure 3.16: *Staggered grid configuration in 2d. Extending the concept to 3d is straightforward.*

offsets are used anew to find their new location. The particular structure of discretization matrix of the transport term in the real-valued P_N-equations is what allows placing the coefficients without any conflicts.

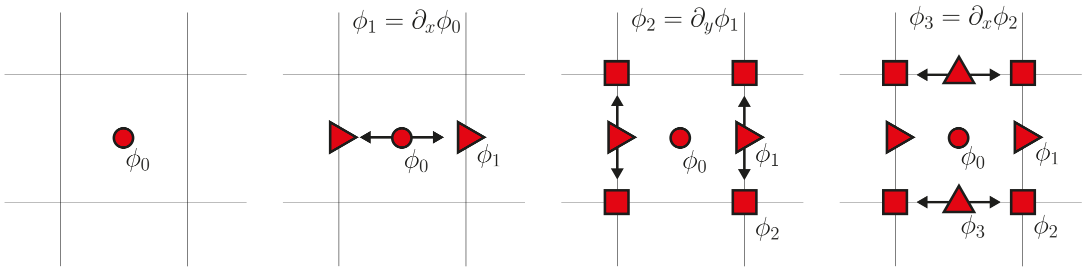

ϕ_0
$\phi_1 = \partial_x\phi_0$
ϕ_0
ϕ_1
$\phi_2 = \partial_y\phi_1$
ϕ_0
ϕ_1
ϕ_2
$\phi_3 = \partial_x\phi_2$
ϕ_0
ϕ_1
ϕ_3
ϕ_2

Figure 3.17: *The process of finding staggered grid locations for unknowns ϕ_i. The P_N-equations relate the unknowns through spatial derivatives, which imply an order on the finite difference grid. The first unknown is placed at the voxel center and the remaining unknown locations are found by following the spatial derivatives through the equations.*

It is important to realize that each equation in the system of linear equations relates a particular coefficient with spherical harmonics indices l,m in a particular voxel within the finite difference grid to other coefficients at the same or adjacent voxels (due to derivatives). Therefore, the equation associated with a particular spherical harmonics index l,m within the P_N-equations, is implicitly defined at the location of the spherical harmonics coefficient with the same l,m index. Due to staggering, the equations are consequently defined at different locations. This has a paramount consequence for the spatial discretization step during stencil code generation.

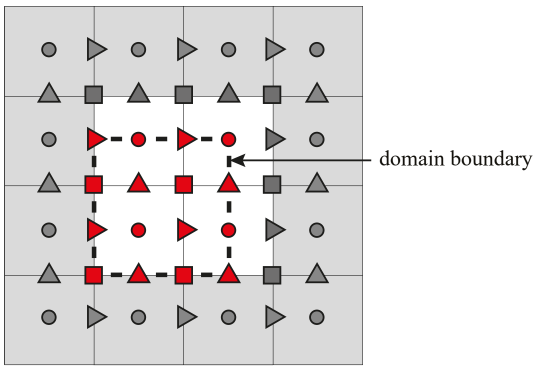
In section 3.5.2 the spatial discretization was carried out as a manipulation on the expression tree representing the P_N-equations. This manipulation step traversed the tree and replaced the continuous spatial variable $\mathbf{x}$ by its discrete counterpart x_{ijk}. During traversal, a stack was used to keep track of the current location which would change due to shifts from central difference discretizations of differential operators within the expression tree. This stack was initialized with the reference position x_{ijk}, which would refer to the voxel center. This was correct, as all coefficients and equations were collocated at this position. With staggered grids, the equations are to be evaluated at the grid location implied by the position of the respective spherical harmonics coefficient with index l,m. Therefore, to account for staggered grids, the position stack needs to be initialized with the staggered grid location of the l,m-equation, which is being discretized.

The discretization step will replace the evaluation of a coefficient at a particular location, with an interpolation of that coefficient from the surrounding grid locations at which it is defined (figure 3.8). This approach will handle the staggering naturally.

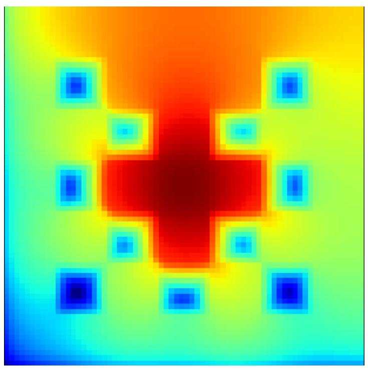
Staggering spherical harmonics coefficients also makes handling boundary conditions more involved. With collocated grids, it is straightforward to identify boundary voxels and voxels in the computational domain. Each voxel in the computational domain contributes the same number of unknowns to the system. With staggered grids, some boundary voxels will have a subset of their unknowns that lie directly on the boundary. Without changes to the system, this will imply a shifted boundary interface and introduce an asymmetry. Coefficients, which are located on the boundary, will be unknowns and part of the solution vector on one side and subject to boundary conditions on the other side (figure 3.18 a). This creates different boundary conditions on opposite sides of the problem domain and causes an asymmetry in the solution for symmetric problems (figure 3.18 b).

The solution is to introduce additional unknowns to the system for every coefficient which is located in a boundary voxel as shown in figure 3.19a. This will correct the location of the boundary interface and allows the same boundary conditions on opposite sides of the computational domain.

The handling of boundary conditions is done by the solver framework. In order to support staggered grids, the stencil code will be generated with an additional function, which specifies the grid location of each coefficient. This function is used by the *VoxelManager* to work out which additional unknowns are required. This results in complex management of indices of unknowns, as a simple indexing function is not sufficient anymore. Transparently handling this complexity is the purpose of the *VoxelManager* class.

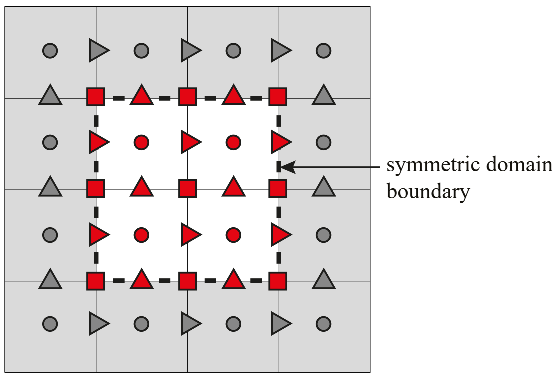


(a) *Per voxel staggered grid locations.*

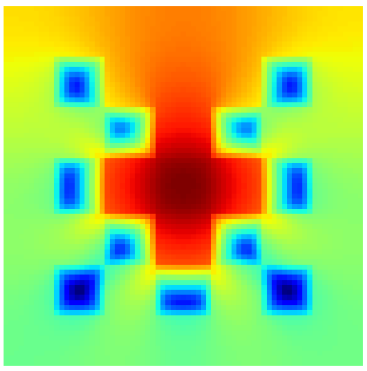


(b) P_1*-solution.*

Figure 3.18: *Dealing with staggered grid locations per voxel (left) implies a boundary interface which produces asymmetric results as seen for the solution of the checkerboard problem (right).*



(a) *Adjusted placement of staggered grid coefficients.*



(b) P_1*-solution of the checkerboard problem.*

Figure 3.19: *Placing staggered grid coefficients such that the defined boundary interface is symmetric (left) leads to a correct symmetric solution of the checkerboard problem (right).*

For client code applications, it is desirable to have all coefficients defined at the centers of voxels covering the problem domain. Therefore the solver framework offers an *unstagger* function for staggered grids, which interpolates all coefficients at the voxel centers of the problem domain from the surrounding coefficient locations. Further the additional unknowns, which have been introduced to get correct boundary conditions, will be removed. This function converts a staggered grid solution vector to a collocated grid solution vector.

As shown in figure 3.19b, using the collocated grid from converting a staggered grid solution of a staggered grid solve, will produce a result free of oscillation and asymmetry artefacts.

3.5.7 System Matrix and Solution

After initializing the *PNSystemBuilder* object matrix with the *Stencil* and *Scene*, the system matrix A and right hand side vector $\mathbf{b}$ are built and made available to the client code, which will use it to solve for the solution vector $\mathbf{u}$. The solver developed as part of this thesis uses Eigen [34], but any other linear algebra package with standard methods available is applicable too.

However, the system matrix A, when generated from the P_N-equations, has properties that restrict the application of methods for solving the system. The number of rows and columns of A is determined by the total number of voxels in the computational domain, multiplied by the number of spherical harmonics coefficients per voxel. The matrix is squared and sparse due to the local structure of the discretization and coupling. Unfortunately, the system matrix A is non-symmetric (due to the transport term) and not diagonal dominant, which together rules out many iterative methods for solving linear systems.

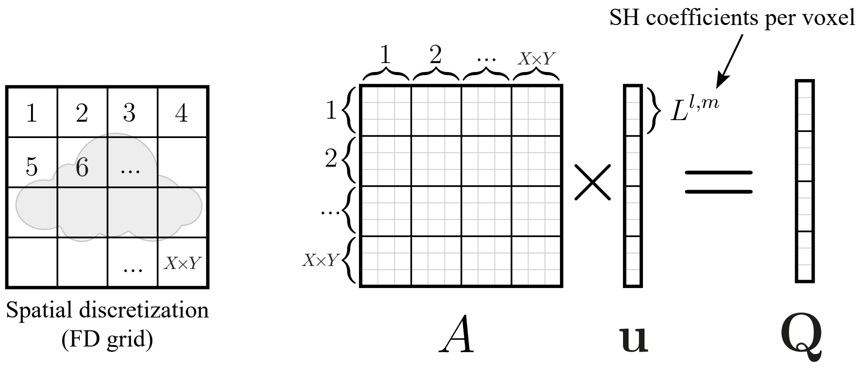


Figure 3.20: *Structure of coefficient matrix A and solution vector $\mathbf{u}$ after discretization of the P_N-equations on a finite difference grid.*

Some standard methods, such as LU-decomposition, QR-decomposition or Singular-value decomposition allow for solving systems of linear equations with non-symmetric coefficient matrices A and are applicable to solving the system generated by the P_N-solver.

Due to the asymmetry in A, applying iterative methods becomes a challenge. While iterative methods exist for non-symmetric matrices (based on Arnoldi iterations), these methods are more complex and tend to require more memory, that furthermore linearly grows with each operation. So only a limited number of iterations can be done before a restart becomes necessary.

Instead, iterative methods can be applied more easily by solving the normal form

$$A^T A\mathbf{u} = A^T\mathbf{b} \ . \tag{3.38}$$

This gives a symmetric and positive definite system matrix $A^T A$, albeit with a higher condition number. Investigation of other solution schemes (e.g. multigrid) would be an interesting avenue for future work.

However, more importantly, in the presence of vacuum regions, matrix A becomes singular and the system cannot be solved at all. The intuition behind this behavior is that in case of vacuum, the collision and scattering terms vanish completely, leaving the transport operator to exclusively define the solution. Since the transport operator only defines the relationship between unknowns in terms of their derivatives, it becomes clear that this is not enough information to constitute a unique solution. This is best exemplified by looking at a minimal version of that problem:

$$\frac{df(x)}{dx} = b(x)$$

This differential equation imposes a constraint on the derivative of the solution, which is unique up to a constant factor. Hence there are infinitely many solutions. Therefore, there is no unique solution to this problem and hence the matrix A after discretization will be singular.

A way out of this problem is to specify additional boundary conditions. However, for problems of simple complexity, this would already amount to finding a solution to the problem in the first place.

A simple alternative to which this thesis refers back is minimum thresholding. A minimum threshold value σ_{min} is specified by the user. That threshold value is returned for any extinction value that falls below that threshold. This amounts to adding an ambient extinction medium by which vacuum regions are avoided completely and which causes the system to never become singular. Let σ_{ot} be the original extinction coefficient coming from the input dataset. Unless stated otherwise, throughout this thesis the extinction coefficient is defined to be

$$\sigma_t(\mathbf{x}) = \begin{cases} \sigma_{ot}(\mathbf{x}), & \text{for } \sigma_{ot} \geq \sigma_{min} \\ \sigma_{min}, & \text{otherwise} \end{cases} \tag{3.39}$$

This concludes the section on the P_N-solver and its system components.

3.6 Integration into Rendering Software

This section will explain in detail how the solver and its result can be integrated and used in a standard rendering framework.

The solver framework is encapsulated using the *PNSystemBuilder* class and therefore, can be easily migrated from a standalone application into any rendering framework. As outlined in section 3.5.5, references to a *Stencil* class and the *Scene* class are needed to create a *PNSystemBuilder* instance.

The truncation order N is a user parameter, which ideally is decided at runtime. However, the stencil code will always be the same for a given truncation order N, since it has been generated independently of grid resolution, boundary conditions, and radiative transfer parameters. It suggests itself to precompute and precompile the stencil code for all possible options for N. The rendering framework is compiled once with all possible stencil codes. As soon as the user decided, which truncation order N to use, a *Stencil* class is generated using the function pointers of the respective compiled stencil code functions.

The *Scene* class is also created by the rendering framework. It will be populated with references to *Field* classes for radiative transfer quantities, such as extinction, albedo, emission and phase function. The *Field* classes serve as adapter pattern according to Gamma et al. [30] and can be used to feed arbitrary representations of spatially varying scalar fields, as input to the solver framework.

An important aspect of providing inputs to the solver relates to the emission term, which describes the illumination within the problem domain. Light sources, that illuminate the volume from outside, will have to be expressed as emission on the boundary interface and light sources inside the problem domain, such as point- or area lights need to be rasterized into the emission field. Expressing outside illumination at the boundaries potentially conflicts with the boundary conditions, which cause the solution near the boundary to be less accurate. Rasterizing light sources inside the domain will be challenging for the solver, if strong directionality is present. The reason for this is the spherical harmonics approximation that has difficulties representing delta or strongly directional functions in angular domain due to the truncation of higher frequency moments.

To alleviate the challenges of light sources with strong directionality, the rendering system, developed as part of this thesis, implements an idea, which is inspired by Grosjean's analytical approximation to the exact solution for a point light source within a homogeneous medium (Grosjean [33]). Instead of providing an approximation to the full light transport caused by the point light, the approximation separates the unscattered light from the scattered light. The solution for the unscattered contribution is exact and an approximation is needed only for the scattered light. This approach is reasonable as the light contribution from scattered light has more energy in the lower frequencies due to the phase function, which acts as a convolution filter on the radiance function in angular domain.

Following this idea by Grosjean, the radiance field L is separated into unscattered light L_u and multiple scattered light L_m:

$$L(\mathbf{x},\boldsymbol{\omega}) = L_u(\mathbf{x},\boldsymbol{\omega})+L_m(\mathbf{x},\boldsymbol{\omega}) \tag{3.40}$$

The unscattered light is the contribution of the radiance field, that did not undergo any scattering events. Therefore, it satisfies the following restricted radiative transfer equation, where the scattering term vanishes:

$$(\nabla\cdot\boldsymbol{\omega})L_u(\mathbf{x},\boldsymbol{\omega}) = -\sigma_t(\mathbf{x})L_u(\mathbf{x},\boldsymbol{\omega}) \tag{3.41}$$

When replacing the radiance field L with the above decomposition in the radiative transfer equation (equation 2.5) the result is:

$$\begin{aligned}(\nabla\cdot\boldsymbol{\omega})L_u(\mathbf{x},\boldsymbol{\omega})+(\nabla\cdot\boldsymbol{\omega})L_m(\mathbf{x},\boldsymbol{\omega}) =& -\sigma_t(\mathbf{x})L_u(\mathbf{x},\boldsymbol{\omega})\\ & -\sigma_t(\mathbf{x})L_m(\mathbf{x},\boldsymbol{\omega})\\ & \underbrace{+\sigma_s(\mathbf{x})\int_\Omega \rho\left(\boldsymbol{\omega}'\cdot\boldsymbol{\omega}\right)L_u\left(\mathbf{x},\boldsymbol{\omega}'\right)\mathrm{d}\boldsymbol{\omega}'}_{L_s}\\ & +\sigma_s(\mathbf{x})\int_\Omega \rho\left(\boldsymbol{\omega}'\cdot\boldsymbol{\omega}\right)L_m\left(\mathbf{x},\boldsymbol{\omega}'\right)\mathrm{d}\boldsymbol{\omega}'\\ & +Q_e(\mathbf{x},\boldsymbol{\omega})\ .\end{aligned}$$

By using equation 3.41, the extinction terms for the unscattered light can be cancelled out, producing the medium radiative transfer equation:

$$\begin{aligned}(\nabla\cdot\boldsymbol{\omega})L_m(\mathbf{x},\boldsymbol{\omega}) =& -\sigma_t(\mathbf{x})L_m(\mathbf{x},\boldsymbol{\omega})\\ & +\sigma_s(\mathbf{x})\int_\Omega \rho\left(\boldsymbol{\omega}'\cdot\boldsymbol{\omega}\right)L_m\left(\mathbf{x},\boldsymbol{\omega}'\right)\mathrm{d}\boldsymbol{\omega}'\\ & \underbrace{+L_s(\mathbf{x},\boldsymbol{\omega})+Q_e(\mathbf{x},\boldsymbol{\omega})}_{Q}\ .\end{aligned}$$

The quantity L_s is the single scattered light, which is folded together with the volume emission Q_e into the general emission term Q.

Using the emission term Q, the medium radiative transfer equation is solved, using the P_N-method previously described. The solution vector $\mathbf{u}$ is brought into collocated form, using the *unstaggering* function described in section 3.5.6. This allows straightforward subsequent use in a rendering system for image generation.

The solution vector provided by the solver contains a set of spherical harmonics coefficients for every voxel of the computational grid. This set of coefficients describes the spherical radiance function at the voxelcenter. The developed framework provides a *PNSolution* class that is a voxelgrid of K-dimensional vectors. The number of vector elements K is the number of spherical harmonics coefficients, which is driven by the truncation order of the P_N-equations. The radiance field is reconstructed by first interpolating all coefficients at the given worldspace position from the coefficients at surrounding voxels, followed by computing the spherical harmonics reconstruction in equation 3.22.

The images presented here will be rendered from the solution directly. It would also be possible to use the solution in combination with one of the variance reduction techniques outlined in section 2.3.

Volume rendering frameworks are concerned with solving the radiative transfer equation 2.5. In order to integrate the solution of the medium radiative transfer equation, the inscattering term in the radiative transfer equation has to be split and the medium radiance L_m is replaced by its truncated spherical harmonics reconstruction $\widehat{L}_m$:

$$\begin{aligned}(\nabla\cdot\boldsymbol{\omega})L(\mathbf{x},\boldsymbol{\omega}) =& -\sigma_t(\mathbf{x})L(\mathbf{x},\boldsymbol{\omega})\\ &+\sigma_s(\mathbf{x})\int_\Omega \rho\left(\boldsymbol{\omega}'\cdot\boldsymbol{\omega}\right)\left(L_u\left(\mathbf{x},\boldsymbol{\omega}'\right)+\widehat{L}_m\left(\mathbf{x},\boldsymbol{\omega}'\right)\right)\mathrm{d}\boldsymbol{\omega}'\\ &+Q_e(\mathbf{x},\boldsymbol{\omega})\\ =& -\sigma_t(\mathbf{x})L(\mathbf{x},\boldsymbol{\omega})\\ &+\sigma_s(\mathbf{x})\int_\Omega \rho\left(\boldsymbol{\omega}'\cdot\boldsymbol{\omega}\right)L_u\left(\mathbf{x},\boldsymbol{\omega}'\right)\mathrm{d}\boldsymbol{\omega}'\\ &+\sigma_s(\mathbf{x})\int_\Omega \rho\left(\boldsymbol{\omega}'\cdot\boldsymbol{\omega}\right)\widehat{L}_m\left(\mathbf{x},\boldsymbol{\omega}'\right)\mathrm{d}\boldsymbol{\omega}'\\ &+Q_e(\mathbf{x},\boldsymbol{\omega})\end{aligned} \tag{3.42}$$

When seperating the radiance field and using the single scattered light as emission term for the P_N-method, this equation needs to be solved to fully account for the radiance field. Any Monte-Carlo method would need to integrate the single scattered light (e.g. by doing next event estimation). The indirect or multiple scattered light is accounted for by integrating over the angular dependent P_N-solution and weighting it by the phase function and scattering term to account for the inscattering of indirect illumination.

If the phase function is isotropic, then equation 3.42 further simplifies to

$$\begin{aligned}(\nabla\cdot\boldsymbol{\omega})L(\mathbf{x},\boldsymbol{\omega}) &= -\sigma_t(\mathbf{x})L(\mathbf{x},\boldsymbol{\omega})+\sigma_s(\mathbf{x})\int_\Omega \rho\left(\boldsymbol{\omega}'\cdot\boldsymbol{\omega}\right)L_u\left(\mathbf{x},\boldsymbol{\omega}'\right)\mathrm{d}\boldsymbol{\omega}' \\ &\quad +\frac{1}{4\pi}\sigma_s(\mathbf{x})\int_\Omega \widehat{L}_m\left(\mathbf{x},\boldsymbol{\omega}'\right)\mathrm{d}\boldsymbol{\omega}'+Q_e(\mathbf{x},\boldsymbol{\omega}) \\ &= -\sigma_t(\mathbf{x})L(\mathbf{x},\boldsymbol{\omega})+\sigma_s(\mathbf{x})\int_\Omega \rho\left(\boldsymbol{\omega}'\cdot\boldsymbol{\omega}\right)L_u\left(\mathbf{x},\boldsymbol{\omega}'\right)\mathrm{d}\boldsymbol{\omega}' \\ &\quad +\frac{1}{4\pi}\sigma_s(\mathbf{x})L_m^{0,0}(\mathbf{x})+Q_e(\mathbf{x},\boldsymbol{\omega})\ ,\end{aligned} \tag{3.43}$$

due to zero moment property of the radiance field $\int_\Omega \widehat{L}_m \mathrm{d}\boldsymbol{\omega}' = L_m^{0,0}$.

Figure 3.21 gives a schematic overview over how the method is integrated into a rendering framework. The single scattered light is precomputed and projected into the emission term for the P_N-solver. Together with the other discretized radiative transfer parameters, the P_N-solution is obtained. The final image is generated by path tracing to the first bounce into the volume, and by computing the inscattered uncollided light (either by using Monte Carlo integration or by using the precomputed single scattered light, which was used as emission term for the P_N-solver). Instead of continuing tracing the path to account for indirect illumination, a separate inscattering integral is computed over the multiple scattered light from the P_N-solution.

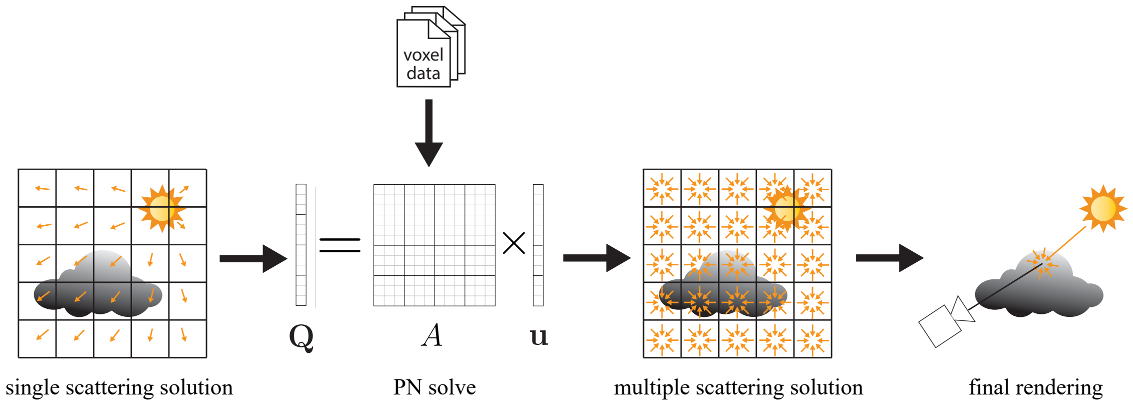


Figure 3.21: *Overview over the full rendering process. The single scattering solution is precomputed and projected into RHS Q. The coefficient matrix A is assembled from voxelized RTE quantities and used to compute solution u. That solution is then un-projected and used to evaluate the multiple scattering contribution during final rendering step.*

After having outlined how the P_N-Method and its solution can be integrated into a rendering application, concrete results will be shown and discussed in the following section.

3.7 Results

After outlining the P_N-solver in section 3.5 and its integration into a rendering framework in the previous section, this section will give and discuss results produced by the system, which was developed as part of this thesis.

3.7.1 Point Source Problem

The point source problem is a canonical problem, which is well-suited to validate the method and assess its accuracy. The problem is simple, and an analytical solution exists to allow comparison against the ground truth (section 2.2). The medium is homogeneous with infinite extent. This is approximated for the solver by a unit cube. The emission term is an ideal unit power point light with isotropic emission in angular domain and a delta function in spatial domain. This delta function is discretized into the finite difference grid by distributing its unit power over the volume of a single voxel:

$$Q_{ijk} = \frac{1}{h_x h_y h_z} \tag{3.44}$$

The index ijk refers to the voxel, in which the point light is located.

Note here, that the radiance field is not separated as outlined in section 3.6. The emission field represents only the discretized light source and therefore, the resulting discretized radiance field represents all light transport, including single scattered light.

The correct analytical solution for fluence is given by D'Eon et al. [22]. The authors also introduce a very accurate approximation, called the Grosjean solution, which is simpler to evaluate and always convergent. The fluence is identical to the zero coefficient of the spherical harmonics expansion of the radiance field ($L^{0,0}$), which can be compared directly with the Grosjean solution. Figure 3.22 compares the solution for P_1, P_2, P_3, P_4, P_5 against the Grosjean solution. The figure shows that with higher N, the P_N-solution increases accuracy. Also noticeable, with increasing truncation order N, the P_N-solution oscillates around the exact solution and approaches it with alternating under- and overestimation, especially near the point source. Even truncation order will overestimate and odd truncation order will produce an underestimate. The performance characteristics are given in table 4.2

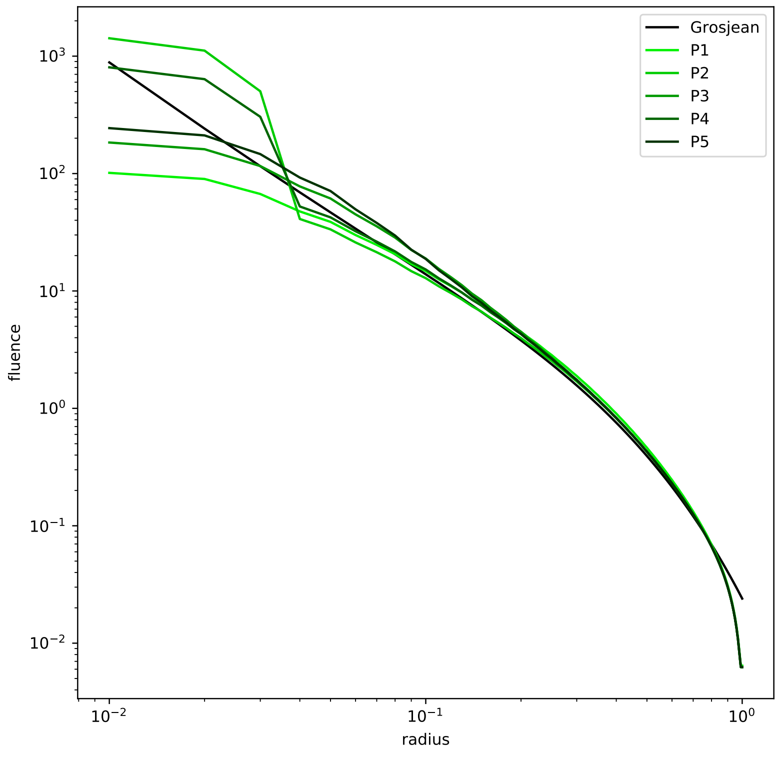


Figure 3.22: *Comparison of P_N-results for the pointsource problem with different truncation order N against groundtruth solution (Grosjean). The disagreement on the right comes from zero dirichlet boundary conditions.*

Table 3.1: *Performance characteristics of the P_N-method for the point source problem for a* 64×64×64 *sized grid.*

Truncation order **N**	1	3	5
Number of rows/columns in A	1.048.576	4.194.304	9.437.184
Size of linear system (in MB)	8.4	33.5	75.5
Solve time (in min)	10	21	45

3.7.2 Checkerboard Problem

The checkerboard problem is a two-dimensional setup solved by the two-dimensional P_N-equations, introduced in section 3.4. This problem is known to the nuclear science field and is used across literature to compare and validate implementations of deterministic methods. It consists of absorbing regions, which are embedded in a scattering medium and arranged in a checkerboard pattern. An emitting region is located in the center square.

Running the P_N-method on the checkerboard problem, allows to compare it against the StaRMAP solver by Seibold et al. [81]. Their solver solves the time-dependent P_N-equations and employs a time-stepping scheme, for which a step-size and a target time has to be specified. In contrast, the solver, which has been introduced in this thesis, solves for the steady-state solution of the real-valued P_N-equations using a global linear system. No step-size or duration parameters are required. The P_N-method will be used on the checkerboard problem for $N = 5$. The results from StaRMAP have been generated by setting a long duration and a small step-size to get an accurate and close-to steady state result.

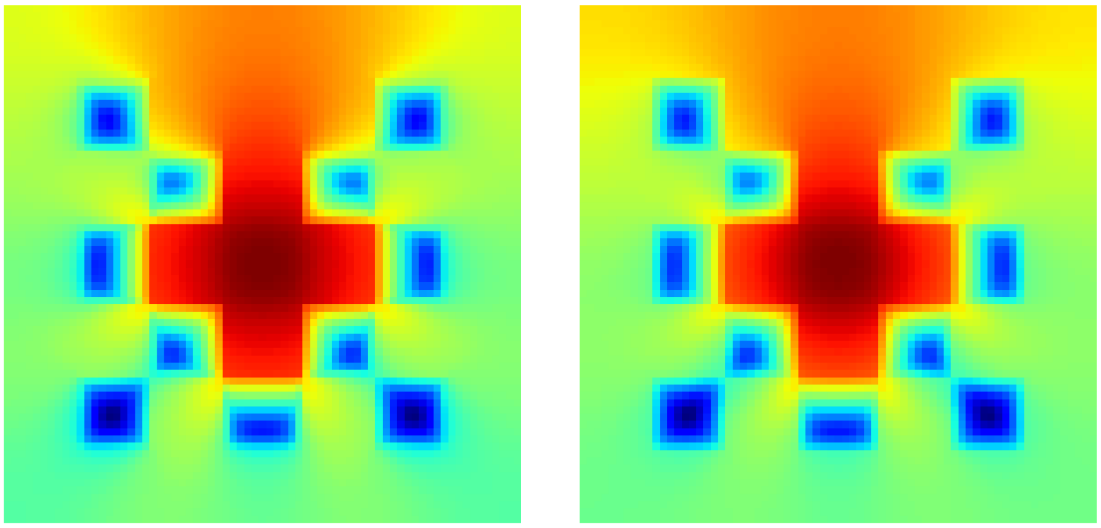


Figure 3.23: *Solver P_5-result for the checkerboard problem (left) in comparsison to StaRMAP by Seibold et al. [81] (right). The solution of the solver, developed as part of this thesis, is in very good agreement.*

3.7.3 Procedural Cloud

For a more practical example the P_N-solver is used to compute the multiple scattered light in a procedurally generated cloud dataset, which is being illuminated by a directional light. The radiance field is separated into single scattered and multiple scattered light as explained in section 3.6. A basic forward path tracer (section 2.3) is used to trace camera rays and integrate single scattered light according to equation 3.42.

Characteristic about the dataset is the presence of very dense media embedded in regions of very low density and vacuum (zero extinction), which exhibits very strong density gradients. The presence of vacuum has a signficiant effect on the convergence behavior of the P_N-method as explained in section 3.5.7. The convergence deteriorates in the presence of very low density regions. The coefficient matrix A becomes singular, when regions of pure vacuum exist in the dataset.

The P_N-method for $N = 1,2,3,4,5$ has been run using a minimum threshold of $\sigma_{min} = 1.0e^{-3}$. The resolution of the finite difference grid used by the P_N-method is 64×64×64. Separating the multiple scattered light from the single scattered light has the benefit that the multiple scattered light has lower frequencies, which allows using a coarser finite difference grid without a significant loss of accuracy.

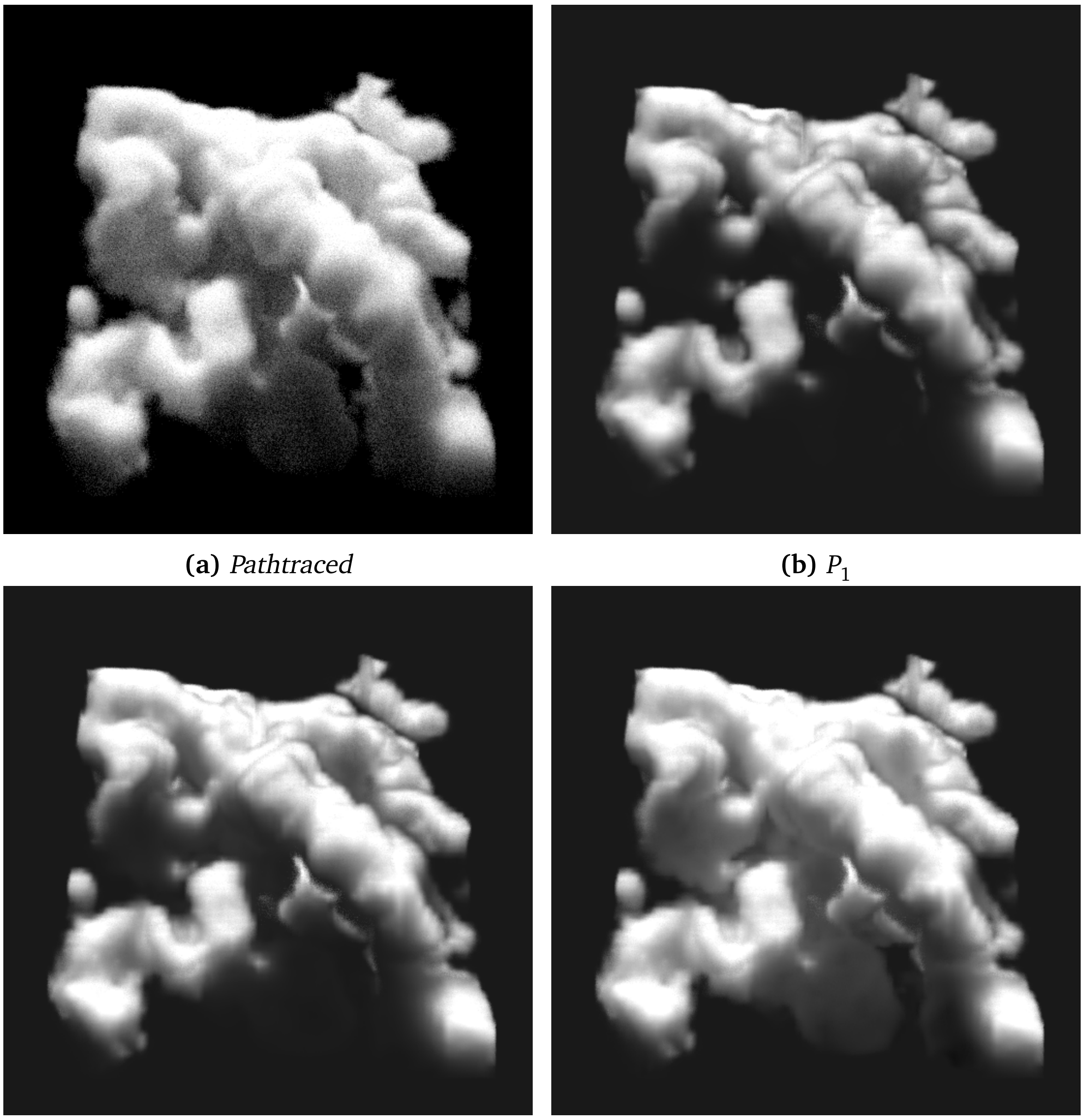
The results for the procedural cloud problem are shown in figure 3.24. As expected, the increase of the truncation order improves the accuracy of the solution. Especially the regions at the bottom of the cloud, which only receive indirect light are better resolved with higher N. However, with the increase of the truncation order, the size of the system increases and requires longer time to converge.

Table 3.2: *Performance characteristics of the P_N-method for the procedural cloud problem for a* 64×64×64 *sized grid with a minimum extinction coefficient threshold of* 4.

Truncation order **N**	1	3	5
Number of rows/columns in A	1.048.576	4.194.304	9.437.184
Size of linear system (in MB)	8.4	33.5	75.5
Solve time (in s)	9	14	20

The performance characteristics shown in table 3.2 are heavily driven by the presence of low density or vacuum regions. In case of vacuum regions specifically, the condition number is singular. In that case the residual flattens and the solver actually never really converges below a certain error value. Therefore application of a minimum threshold for the extinction coefficient σ_t becomes necessary. With a sufficiently high threshold (relative to the length scale of the problem), computation times can be reduced drastically while compromising on the discrepancy to the original problem containing vacuum regions. In figure 3.25 the convergence behavior for solving the normal form using the conjugate-gradient method is being compared for different choices of the threshold σ_{min}.

For rendering of clouds and other participating media in computer graphics, vacuum or low density regions are of great importance. In these scenarios, the P_N-method is not competitive when compared to other techniques.



(a) *Pathtraced*

(b) P_1

(c) P_3

(d) P_5

Figure 3.24: *P_N-results for the procedural cloud dataset (multiple scattered light) with different truncation order N.*

3.8 Discussion

The solver, which has been introduced in this chapter produces results that are in good agreement with radiative transfer ground truth solutions. This has been demonstrated with the point source problem in section 3.7.1. The results for the checkerboard problem (section 3.7.2) are in very good agreement with results from deterministic meth-

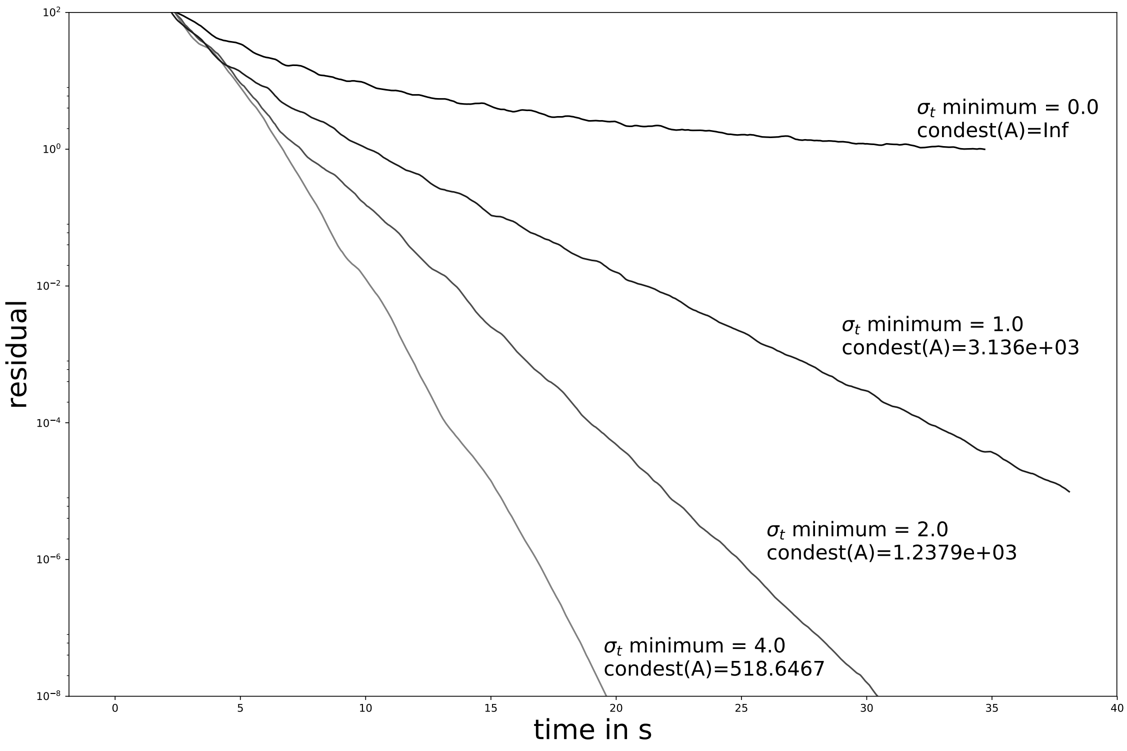


Figure 3.25: *Convergence of P_5 for different choices of minimum threshold σ_{min} and its effect on the conditional number for the coefficient matrix A.*

ods from nuclear physics. This further demonstrates the correct derivation of the real-valued P_N-equations and implementation of the solver, developed as part of this thesis.

The procedural cloud results in section 3.7.3 show, that in order to capture light in the visually important indirect regions of the dataset, truncation orders of $N = 5$ or higher are necessary. The computation time and memory requirements in this case are signficiant as seen in table 3.2. In the presence of vacuum, the solver never converges and the extinction coefficient needs to be thresholded. These are very heavy requirements, which will limit the use of the P_N-method in practical applications. The next chapter introduces Flux-limited Diffusion which is derived as a degenerated case of the P_1-equations and produces more accurate results at lower computational cost for $N = 1$.

Chapter 4

Flux-limited Diffusion for Multiple Scattering in Participating Media

This chapter is based on the publication Flux-limited Diffusion for Multiple Scattering in Participating Media [48]

In the previous chapter, the angular variable of the radiative transfer equation was discretized into a set of coupled partial differential equations, namely the P_N-equations. Furthermore, the chapter introduced a discretization of these equations into a system of linear equations, which could be solved using standard methods. Since the number of equations depends on the truncation order N, higher N will result in increasingly larger systems. The results showed that the amount of computational resources required makes this approach unattractive for practical rendering applications, especially for higher N (section 3.7).

In this chapter, flux-limited diffusion is introduced to solve the problem of rendering participating media in computer graphics. The theory has been invented in the astrophysics domain and is primarily attributed to Levermore et al. [56]. Their work introduces the theory and discusses its properties for certain canonical problems, such as a single point source within a homogeneous participating medium. In this chapter, a novel numerical method is devised, based on flux-limited diffusion theory. It allows solving for the fluence on a finite difference grid with higher accuracy at a significantly lower cost than P_1.

In order to derive the theory for flux-limited diffusion, we will switch in this chapter to a different way of expressing the spherical harmonics expansion. This alternative representation is called *moment expansion*. It is more efficient in regards to notation for lower order N and allows to present the theory more intuitively.

In section 4.1 and 4.2, we introduce the moment expansion for the radiance field and radiative transfer equation respectively. The section 4.3 follows by introducing the *moment closure problem* and its naive solution, which gives rise to the diffusion approximation. The diffusion approximation is discussed in section 4.4. It is a method for solving the P_N-equations for order $N = 1$ very efficiently and it is an important building block for the theory of flux-limited diffusion.

The chapter then continues by revisiting the moment closure problem from section 4.3 and introduces the flux-limit constraint, which is violated by diffusion and causes energy loss. The streaming transport regime is introduced, for which an advection style equation is derived in section 4.6. Flux-limited diffusion is derived as a means to blend between the two extreme modes of transport using local information. This blending is driven by the variable Eddington factor, which is introduced in section 4.7. Various concrete realizations of the Eddington factor are given and compared in section 4.8. Finally, an iterative solver is presented in section 4.9, for which results are presented in section 4.10. The chapter closes with a discussion in section 4.11.

4.1 Moment Expansion of the Radiance Field

In section 3.1 it has been established that spherical functions can be represented using a linear combination of spherical harmonics basis functions (equation 3.9). The coefficients of these basis functions are found by projection (equation 3.21). Another option for representing spherical functions is the coordinate free form given by tensor calculus. This form is derived by representing the spherical harmonics basis functions themself as linear combination of products of the components of the basis vectors, which establish the underlying coordinate system. Because this representation allows to change the coordinate system independently from the representation, it is referred to as being coordinate free. For a refresher on tensor calculus, an introductory standard such as Grinfeld [32] is recommended. As with the spherical harmonics expansion, the coordinate free representation can be found for the radiance field, as well as the individual terms of the radiative transfer equation. This is referred to as the moment expansion or multipole expansion of the radiance field and the radiative transfer equation respectively.

The moment expansion is a different approach to represent the spherical harmonics expansion of the radiance field and the radiative transfer equation. Its benefit lies in its notational efficiency, especially for lower order. However, for the truncation order of $N = 2$ and higher, it introduces additional redundancy which makes the standard spherical harmonics expansion much better suited when deriving computational methods from the theory, such as the P_N-method in chapter 3. Because the flux-limited diffusion is related to the first order truncation $N = 1$, the moment expansion is the standard way of representing the theory. No additional redundancy is introduced and the notational efficiency allows to express the underlying intuition easier.

The moment expansion is derived, starting from the spherical harmonics reconstruction of the radiance field:

$$L(\boldsymbol{\omega}) = \sum_{l=0}^{\infty}\sum_{m=-l}^{l} L^{l,m} Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) \tag{4.1}$$

with $L^{l,m}$ being the spherical harmonics basis function coefficients, which are found by spherical harmonics projection (equation 3.21).

The main step in the derivation of the moment expansion is the replacement of the spherical harmonics basis function $Y_{\mathbb{C}}^{l,m}$ by a sum of tensor contractions:

$$Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) = \sum_{j=0}^{\infty} y_j^{l,m} \odot N_j(\boldsymbol{\omega}) \tag{4.2}$$

The operator symbol $\odot$ denotes a tensor contraction. In this instance, it is a sum over products of the individual components of the tensors $y_j^{l,m}$ and N_j. Therefore, each contraction collapses into a scalar value. These values are then added up over all moments j. The tensor N_j is a tensor of rank j with $N_0 = 1$, and is constructed by a sequence of outer products of the direction vector $\boldsymbol{\omega}$:

$$N_j(\boldsymbol{\omega}) = N_{k_1,k_2,\dots,k_{j-1},k_j} = \boldsymbol{\omega}_{k_1}\boldsymbol{\omega}_{k_2}\dots\boldsymbol{\omega}_{k_{j-1}}\boldsymbol{\omega}_{k_j} \tag{4.3}$$

The index set $\{k_1,k_2,\dots,k_{j-1},k_j\}$ identifies specific tensor components, with each index running over all components of $\boldsymbol{\omega}$ ($k_i \in \{x,y,z\}$).

The tensors $y_j^{l,m}$ are found by expanding $Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega})$ into a sum of tensor components of N_j. The factors to these components can be easily extracted and constitute the components of $y_j^{l,m}$. An expansion is possible by using the following non-recursive definition of the Legendre-polynomial, which is derived with the use of the multiple-angle formula[1]:

$$P^{lm}(\theta,\phi) = \sin^m(\phi)\sum_{j}^{\lfloor\frac{l-m}{2}\rfloor} a^{lmj}\cos^{l-m-2j}(\theta) \qquad \text{for } m \geq 0 \tag{4.4}$$

[1] http://mathworld.wolfram.com/Multiple-AngleFormulas.html (date of access: 07/26/2018)

Inserting this into the definition of the spherical harmonics basis function $Y_{\mathbb{C}}^{l,m}$ (equation 3.3) results in:

$$Y_{\mathbb{C}}^{l,m}(\theta,\phi) = C^{l,m}\left(e^{i\phi}\sin(\phi)\right)^m \sum_{j}^{\lfloor\frac{l-m}{2}\rfloor} a^{lmj}\cos^{l-m-2j}(\theta) \qquad \text{for } m \geq 0 \tag{4.5}$$

As a next step, $Y_{\mathbb{C}}^{l,m}$ will be expressed in terms of unit direction vector $\boldsymbol{\omega}$, instead of spherical coordinates. This is done by using the identity $e^{i\phi}\sin(\phi) = \boldsymbol{\omega}_x + i\boldsymbol{\omega}_y$ and $\cos(\theta) = \boldsymbol{\omega}_z$:

$$Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) = C^{l,m}\left(\boldsymbol{\omega}_x + i\boldsymbol{\omega}_y\right)^m \sum_{j}^{\lfloor\frac{l-m}{2}\rfloor} a^{lmj}\boldsymbol{\omega}_z^{l-m-2j} \qquad \text{for } m \geq 0 \tag{4.6}$$

$$= (-1)^m \overline{Y_{\mathbb{C}}^{l,|m|}} \qquad \text{for } m < 0 \tag{4.7}$$

Expanding the expression for concrete values of l and m, will give a sequence of terms, which involve components of N_j, multiplied with specified factors. By seeing the expanded formula as a tensor contraction, the components of $y_j^{l,m}$ can be extracted. For $l = 0$ and $m = 0$ equation 4.6 results in:

$$Y_{\mathbb{C}}^{0,0}(\boldsymbol{\omega}) = C^{0,0}a^{0,0,0} = \frac{1}{\sqrt{4\pi}}N_0 \quad , \tag{4.8}$$

from wich the following can be inferred:

$$y_0^{0,0} = \frac{1}{\sqrt{4\pi}} \tag{4.9}$$

For $l = 1$ the result is:

$$Y_{\mathbb{C}}^{1,0}(\boldsymbol{\omega}) = C^{1,0}a^{1,0,0}\boldsymbol{\omega}_z = \sqrt{\frac{3}{4\pi}}\boldsymbol{\omega}_z \tag{4.10}$$

$$Y_{\mathbb{C}}^{1,1}(\boldsymbol{\omega}) = C^{1,1}a^{1,1,0}\boldsymbol{\omega}_x + C^{1,1}a^{1,1,0}\boldsymbol{\omega}_y i = -\sqrt{\frac{3}{8\pi}}\boldsymbol{\omega}_x - \sqrt{\frac{3}{8\pi}}\boldsymbol{\omega}_y i \tag{4.11}$$

$$Y_{\mathbb{C}}^{1,-1}(\boldsymbol{\omega}) = -\overline{Y_{\mathbb{C}}^{1,1}}(\boldsymbol{\omega}) = \sqrt{\frac{3}{8\pi}}\boldsymbol{\omega}_x - \sqrt{\frac{3}{8\pi}}\boldsymbol{\omega}_y i \tag{4.12}$$

From which the following can be inferred:

$$y_1^{1,-1} = \begin{pmatrix} \sqrt{\frac{3}{8\pi}} \\ -\sqrt{\frac{3}{8\pi}}i \\ 0 \end{pmatrix} \qquad y_1^{1,0} = \begin{pmatrix} 0 \\ 0 \\ \sqrt{\frac{3}{4\pi}} \end{pmatrix} \qquad y_1^{1,1} = \begin{pmatrix} -\sqrt{\frac{3}{8\pi}} \\ -\sqrt{\frac{3}{8\pi}}i \\ 0 \end{pmatrix} \tag{4.13}$$

This scheme continues and becomes more involved for higher moments. For example with $l = 3$ an $m = 0$:

$$Y_{\mathbb{C}}^{3,0}(\boldsymbol{\omega}) = \underbrace{C^{3,0}a^{3,0,0}}_{\in y_3^{3,0}}\boldsymbol{\omega}_z{}^3 + \underbrace{C^{3,0}a^{3,0,1}}_{\in y_1^{3,0}}\boldsymbol{\omega}_z \tag{4.14}$$

Here a term containing $\boldsymbol{\omega}_z{}^3$ is presented, which is an element of N_3 and another term containing $\boldsymbol{\omega}_z$, which is an element of N_1. This shows the expression of the the spherical harmonics basis functions as a sum over tensor contractions of different ranks, as Equation 4.2.

Inserting equation 4.2 into equation 4.1 gives:

$$L(\boldsymbol{\omega}) = \sum_{l=0}^{\infty}\sum_{m=-l}^{l} L^{l,m} \sum_{j=0}^{\infty} y_j^{l,m} \odot N_j(\boldsymbol{\omega}) \tag{4.15}$$

After rearranging terms that leads to:

$$L(\boldsymbol{\omega}) = \sum_{l=0}^{\infty}\sum_{m=-l}^{l}\sum_{j=0}^{\infty} L^{l,m} y_j^{l,m} \odot N_j(\boldsymbol{\omega}) \tag{4.16}$$

For each spherical harmonics basis l,m, all rank j tensors will be iterated, and contracted with their respective N_j (after applying the respective spherical harmonics coefficient as weight). Since the indices l and j run over the same range, these can be swapped. This is identical to rearranging the order of terms:

$$L(\boldsymbol{\omega}) = \sum_{l=0}^{\infty}\underbrace{\sum_{j=0}^{\infty}\sum_{m=-j}^{j} L^{j,m} y_l^{j,m}}_{=f_l} \odot N_l(\boldsymbol{\omega}) \tag{4.17}$$

For each spherical harmonics band l, all spherical harmonics bases j,m will be iterated, weighted and the rank l tensor of each particular basis contracted with N_l. Further, the sum of equal rank tensors can be factorized into the moment tensor f_l to receive the moment expansion of the radiance field:

$$L(\boldsymbol{\omega}) = \sum_{l=0}^{\infty} f_l \odot N_l(\boldsymbol{\omega}) \tag{4.18}$$

with

$$f_l = \sum_{j=0}^{\infty}\sum_{m=-j}^{j} L^{j,m} y_l^{j,m} \tag{4.19}$$

For f_0 we get:

$$f_0 = L^{0,0}y_0^{0,0} = \int_\Omega L(\boldsymbol{\omega})Y_{\mathbb{C}}^{0,0}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\frac{1}{\sqrt{4\pi}} \tag{4.20}$$

$$= \frac{1}{4\pi}\int_\Omega L(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} = \frac{1}{4\pi}\phi \tag{4.21}$$

For f_1, $L^{1,-1}y_1^{1,-1}$, $L^{1,0}y_1^{1,0}$ and $L^{1,1}y_1^{1,1}$ shall be added. Starting with $L^{1,-1}y_1^{1,-1}$, following result is given:

$$L^{1,-1}y_1^{1,-1} = \int_\Omega L(\boldsymbol{\omega})Y_{\mathbb{C}}^{1,-1}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\begin{pmatrix}\sqrt{\frac{3}{8\pi}}\\ -\sqrt{\frac{3}{8\pi}}i\\ 0\end{pmatrix} \tag{4.22}$$

$$= \frac{3}{8\pi}\int_\Omega L(\boldsymbol{\omega})\sin\theta e^{-i\phi}\mathrm{d}\boldsymbol{\omega}\begin{pmatrix}1\\ -i\\ 0\end{pmatrix} \tag{4.23}$$

Using the trigonometric identities $e^{-i\phi} = \cos\phi - i\sin\phi$ and $e^{i\phi} = \cos\phi + i\sin\phi$, the outcome is:

$$L^{1,-1}y_1^{1,-1} = \begin{pmatrix}\frac{3}{8\pi}\int_\Omega L(\boldsymbol{\omega})\sin\theta\cos\phi\mathrm{d}\boldsymbol{\omega} - \frac{3}{8\pi}i\int_\Omega L(\boldsymbol{\omega})\sin\theta\sin\phi\mathrm{d}\boldsymbol{\omega}\\ -\frac{3}{8\pi}i\int_\Omega L(\boldsymbol{\omega})\sin\theta\cos\phi\mathrm{d}\boldsymbol{\omega} - \frac{3}{8\pi}\int_\Omega L(\boldsymbol{\omega})\sin\theta\sin\phi\mathrm{d}\boldsymbol{\omega}\\ 0\end{pmatrix} \tag{4.24}$$

We then use the identity:

$$\boldsymbol{\omega} = \begin{pmatrix}\sin\theta\cos\phi\\ \sin\theta\sin\phi\\ \cos\theta\end{pmatrix}, \tag{4.25}$$

to arrive at:

$$L^{1,-1}y_1^{1,-1} = \begin{pmatrix}\frac{3}{8\pi}\int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_x\mathrm{d}\boldsymbol{\omega} - \frac{3}{8\pi}i\int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_y\mathrm{d}\boldsymbol{\omega}\\ -\frac{3}{8\pi}i\int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_x\mathrm{d}\boldsymbol{\omega} - \frac{3}{8\pi}\int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_y\mathrm{d}\boldsymbol{\omega}\\ 0\end{pmatrix} \tag{4.26}$$

Similar procedures for $L^{1,0}y_1^{1,0}$ and $L^{1,1}y_1^{1,1}$, lead to:

$$L^{1,0}y_1^{1,0} = \begin{pmatrix}0\\ 0\\ \frac{3}{4\pi}\int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_z\mathrm{d}\boldsymbol{\omega}\end{pmatrix} \tag{4.27}$$

and

$$L^{1,1}y_1^{1,1} = \begin{pmatrix}\frac{3}{8\pi}\int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_x i\mathrm{d}\boldsymbol{\omega} + \frac{3}{8\pi}i\int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_y\mathrm{d}\boldsymbol{\omega}\\ \frac{3}{8\pi}i\int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_x i\mathrm{d}\boldsymbol{\omega} + \frac{3}{8\pi}\int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_y\mathrm{d}\boldsymbol{\omega}\\ 0\end{pmatrix} \tag{4.28}$$

Subsequently f_1 can be merged. Note, how all the imaginary terms cancel out:

$$f_1 = L^{1,-1}y_1^{1,-1} + L^{1,0}y_1^{1,0} + L^{1,1}y_1^{1,1} = \frac{3}{4\pi}\begin{pmatrix} \int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_x \mathrm{d}\boldsymbol{\omega} \\ \int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_y \mathrm{d}\boldsymbol{\omega} \\ \int_\Omega L(\boldsymbol{\omega})\boldsymbol{\omega}_z \mathrm{d}\boldsymbol{\omega} \end{pmatrix} = \frac{3}{4\pi}\mathbf{E} \tag{4.29}$$

With f_0 and f_1, the moment expansion of the radiance field truncated after the first moment can be written down and used for reconstruction in the diffusion approximation:

$$\begin{aligned} L(\boldsymbol{\omega}) &= f_0 \odot N_0 + f_1 \odot N_1 \\ &= \frac{1}{4\pi}\phi + \frac{3}{4\pi}\mathbf{E}\odot\boldsymbol{\omega} \\ &= \frac{1}{4\pi}\phi + \frac{3}{4\pi}(w\cdot\mathbf{E}) \end{aligned} \tag{4.30}$$

The tensor f_0 is expressed in terms of the fluence ϕ and the tensor f_1 in terms of the flux vector **E**. These quantities are often referred to as the zero and first moment of the radiance field respectively, they generalize to higher order and are denoted L_i. They are computed using the moment projection operator μ_i:

$$\mu_i[L] = L_i = \int_\Omega L(\boldsymbol{\omega})N_i(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{4.31}$$

An important quantity for deriving flux-limited diffusion is the second moment of the radiance field, the radiative pressure tensor L_2. Its intuition is very similar to the stress tensor in continuum mechanics.

4.2 Moment Expansion of the Radiative Transfer Equation

As with the radiance field, the radiative transfer equation can be developed into its moments. The result is a concise representation of the P_N-equations, which are particularly expressive for lower truncation order, such as P_1. The derivation steps of the spherical harmonics expansion replaced the radiance field quantity in the RTE by its truncated expansion, followed by the expansion of the individual terms of the RTE. The derivation steps of the moment expansion are similar, with the difference, that the moment expansion (equation 4.31) is applied to the RTE terms (equation 2.5) first and the, replacement is done as a second step.

The n-th moment of the RTE is found by applying the moment projection operator μ_n to all the terms of the RTE:

$$\mu_n[(\boldsymbol{\omega}\cdot\nabla)L(\mathbf{x},\boldsymbol{\omega})] =$$
$$\mu_n\left[-\sigma_t(\mathbf{x})L(\mathbf{x},\boldsymbol{\omega})+\sigma_s(\mathbf{x})\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}' \to \boldsymbol{\omega})L(\mathbf{x},\boldsymbol{\omega}')\mathrm{d}\boldsymbol{\omega}'+Q(\mathbf{x},\boldsymbol{\omega})\right]$$

Since μ_n is linear, the LHS gives:

$$\begin{aligned}\mu_n[(\boldsymbol{\omega}\cdot\nabla)L(\mathbf{x},\boldsymbol{\omega})] &= \int_\Omega (\boldsymbol{\omega}\cdot\nabla)L(\mathbf{x},\boldsymbol{\omega})N_n\mathrm{d}\boldsymbol{\omega}\\ &= \nabla\int_{S^2} L(\mathbf{x},\boldsymbol{\omega})N_n\boldsymbol{\omega}\mathrm{d}\boldsymbol{\omega}\\ &= \nabla\mu_{n+1}[L]\end{aligned}$$

The term $\nabla\mu_{n+1}$ is a tensor divergence. The general moment equation of the RTE for the n-th moment is:

$$\begin{aligned}\nabla\mu_{n+1}[L] =& -\sigma_t(\mathbf{x})\mu_n[L(\mathbf{x},\boldsymbol{\omega})]\\ &+\sigma_s(\mathbf{x})\mu_n\left[\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}' \to \boldsymbol{\omega})L(\mathbf{x},\boldsymbol{\omega}')\mathrm{d}\boldsymbol{\omega}'\right]\\ &+\mu_n[Q(\mathbf{x},\boldsymbol{\omega})]\end{aligned} \tag{4.32}$$

Generally, the moment equations of the RTE relate the divergence of the $n+1$ moment of the radiance field, to the n-th moment of changes to the radiance field due to inscattering, absorption, and emission. The inscattering term convolves the radiance field with the phase function and applies the scattering coefficient as a weighting function on top. Each moment produces a number of equations, which are identical to the number of tensor components for a tensor of the rank associated with that moment.

For the derivation of the diffusion approximation in this chapter, the moment expansion of the RTE into its first two moments will be expanded. Expanding equation 4.32 with $n = 0$ gives the zero moment equation:

$$\begin{aligned}\nabla\mathbf{E}(\mathbf{x}) =& -\sigma_t(\mathbf{x})\mu_0[L(\mathbf{x},\boldsymbol{\omega})]\\ &+\sigma_s(\mathbf{x})\mu_0\left[\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}' \to \boldsymbol{\omega})L(\mathbf{x},\boldsymbol{\omega}')\mathrm{d}\boldsymbol{\omega}'\right]\\ &+\mu_0[Q(\mathbf{x},\boldsymbol{\omega})]\end{aligned} \tag{4.33}$$

The first moment equation is likewise found by using $n = 1$ in equation 4.32:

$$\begin{aligned}\nabla P =& -\sigma_t(\mathbf{x})\mu_1[L(\mathbf{x},\boldsymbol{\omega})]\\ &+\sigma_s(\mathbf{x})\mu_1\left[\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}' \to \boldsymbol{\omega})L(\mathbf{x},\boldsymbol{\omega}')\mathrm{d}\boldsymbol{\omega}'\right]\\ &+\mu_1[Q(\mathbf{x},\boldsymbol{\omega})]\end{aligned} \tag{4.34}$$

$P = \mu_2[L]$ is the radiation pressure tensor, a 3×3 - matrix. The divergence of a second rank tensor is defined by tensor calculus to be the vector, containing the divergence of each single column vector of that tensor. Therefore, $\nabla P = \partial_i P_{ij}$.

The next step is to replace the radiance field L by its two term expansion (equation 4.30). Doing this with the zero moment expansion of the RTE (equation 4.33) gives:

$$\begin{aligned}
\nabla \mathbf{E}(\mathbf{x}) =& -\sigma_t(\mathbf{x})\mu_0\left[\frac{1}{4\pi}\phi(\mathbf{x}) + \frac{3}{4\pi}\mathbf{E}(\mathbf{x})\right] \\
&+\sigma_s(\mathbf{x})\mu_0\left[\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}' \rightarrow \boldsymbol{\omega})\frac{1}{4\pi}\phi(\mathbf{x})\mathrm{d}\boldsymbol{\omega}'\right] \\
&+\sigma_s(\mathbf{x})\mu_0\left[\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}' \rightarrow \boldsymbol{\omega})\frac{3}{4\pi}\mathbf{E}(\mathbf{x})\mathrm{d}\boldsymbol{\omega}'\right] \\
&+Q_0(\mathbf{x}) \\
=& -\frac{1}{4\pi}\phi\sigma_t(\mathbf{x})\mu_0[1] - \frac{3}{4\pi}\mathbf{E}(\mathbf{x})\sigma_t(\mathbf{x})\mu_1[1] \\
&+\frac{1}{4\pi}\phi(\mathbf{x})\sigma_s(\mathbf{x})\mu_0\left[\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}' \rightarrow \boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}'\right] \\
&+\frac{3}{4\pi}\mathbf{E}(\mathbf{x})\sigma_s(\mathbf{x})\mu_1[1]\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}' \rightarrow \boldsymbol{\omega})\boldsymbol{\omega}'\mathrm{d}\boldsymbol{\omega}' \\
&+Q_0(\mathbf{x}) \\
=& -\phi(\mathbf{x})\sigma_t(\mathbf{x}) + \phi(\mathbf{x})\sigma_s(\mathbf{x}) + Q_0(\mathbf{x}) \\
=& -\phi(\mathbf{x})\sigma_a(\mathbf{x}) + Q_0(\mathbf{x})
\end{aligned}$$

Using $\mu_1[1] = 0$ and normalized phase function results in:

$$\mu_0\left[\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}' \rightarrow \boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}'\right] = \mu_0[1] = 4\pi$$

Likewise the radiance field is replaced with its two moment expansion in the first moment expansion of the RTE (equation 4.34):

$$\begin{aligned}
\nabla P =& -\sigma_t(\mathbf{x})\mu_1\left[\frac{1}{4\pi}\phi(\mathbf{x})+\frac{3}{4\pi}\mathbf{E}(\mathbf{x})\right] \\
&+\sigma_s(\mathbf{x})\mu_1\left[\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}'\rightarrow\boldsymbol{\omega})\left(\frac{1}{4\pi}\phi(\mathbf{x})+\frac{3}{4\pi}\mathbf{E}(\mathbf{x})\right)\mathrm{d}\boldsymbol{\omega}'\right] \\
&+Q_1(\mathbf{x}) \\
=& -\sigma_t(\mathbf{x})\mu_1\left[\frac{1}{4\pi}\phi(\mathbf{x})\right]-\sigma_t(\mathbf{x})\mu_1\left[\frac{3}{4\pi}\mathbf{E}(\mathbf{x})\right] \\
&+\frac{1}{4\pi}\phi(\mathbf{x})\sigma_s(\mathbf{x})\mu_1\left[\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}'\rightarrow\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}'\right] \\
&+\frac{3}{4\pi}\mathbf{E}(\mathbf{x})\sigma_s(\mathbf{x})\mu_1\left[\int_\Omega \rho(\mathbf{x},\boldsymbol{\omega}'\rightarrow\boldsymbol{\omega})\boldsymbol{\omega}\mathrm{d}\boldsymbol{\omega}'\right] \\
&+Q_1(\mathbf{x}) \\
=& -\frac{1}{4\pi}\phi(\mathbf{x})\sigma_t(\mathbf{x})\mu_1[1]-\frac{3}{4\pi}\mathbf{E}(\mathbf{x})\sigma_t(\mathbf{x})\mu_2[1] \\
&+\frac{1}{4\pi}\phi(\mathbf{x})\sigma_s(\mathbf{x})\mu_1[1] \\
&+\frac{3}{4\pi}\mathbf{E}(\mathbf{x})\sigma_s(\mathbf{x})\mu_1\left[\mu_1[f]\right] \\
&+Q_1(\mathbf{x}) \\
=& -\frac{3}{4\pi}\mathbf{E}(\mathbf{x})\sigma_t(\mathbf{x})\mu_2[1]+\frac{3}{4\pi}\mathbf{E}(\mathbf{x})g\sigma_s(\mathbf{x})\mu_2[1] \\
=& \left(-\sigma_t(\mathbf{x})\mathbf{I}+g\sigma_s(\mathbf{x})\mathbf{I}\right)\mathbf{E}(\mathbf{x})+Q_1(\mathbf{x}) \\
=& -\sigma_t'(\mathbf{x})\mathbf{E}(\mathbf{x})+Q_1(\mathbf{x})
\end{aligned}$$

Here, the following is used

$$\mu_2[1]=\frac{4\pi}{3}\mathbf{I}$$

along with that the first moment of a phase function, which only depends on the angle between incident and outgoing direction, is its mean cosine g.

The quantity σ_t' is called the reduced extinction coefficient and it is defined as:

$$\sigma_t' = \sigma_t - g\sigma_s$$

We obtain the two term expansion of the radiative transfer equation:

$$\nabla\mathbf{E}(\mathbf{x}) = -\phi(\mathbf{x})\sigma_a(\mathbf{x})+Q_0(\mathbf{x}) \tag{4.35}$$
$$\nabla P(\mathbf{x}) = -\sigma_t'(\mathbf{x})\mathbf{E}(\mathbf{x})+Q_1(\mathbf{x}) \tag{4.36}$$

These equations are the direct counterpart to the P_1-equations in the moment expansion form. The diffusion approximation can be derived by taking the first moment equations and inserting them into the zero moment equations by substituting **E**. The same procedure can be carried out for the P_1-equations (using spherical harmonics coefficients). However, the notation of the moment expansion is more expressive, which is the reason for its popularity in the literature.

Equation 4.36 is solved for **E** as follows:

$$\mathbf{E}(\mathbf{x}) = -\frac{1}{\sigma_t'(\mathbf{x})}\big(\nabla P(\mathbf{x}) - Q_1(\mathbf{x})\big) \tag{4.37}$$

Then equation 4.37 is used to substitute **E** in equation 4.35. This way, the unknown **E** is eliminated and we arrive at a single scalar partial differential equation:

$$\nabla\left(-\frac{1}{\sigma_t'(\mathbf{x})}\big(\nabla P(\mathbf{x}) - Q_1(\mathbf{x})\big)\right) = -\phi(\mathbf{x})\sigma_a(\mathbf{x}) + Q_0(\mathbf{x})\ , \tag{4.38}$$

which can be further rearranged into:

$$\nabla\left(-\frac{1}{\sigma_t'(\mathbf{x})}\nabla P(\mathbf{x})\right) = -\phi(\mathbf{x})\sigma_a(\mathbf{x}) + Q_0(\mathbf{x}) + \nabla Q_1(\mathbf{x}). \tag{4.39}$$

A single unknown remains: the second moment of the radiance field P, which is called the radiative pressure tensor. Resolving, or better approximating, this unknown is what leads to a rich variety of methods and theories, including the diffusion approximation and flux-limited diffusion.

4.3 Moment Closure Problem and Isotropic Closure

The diffusion approximation theories are derived from truncating the moment expansion of the RTE after its first moment and eliminating the unknown **E** by inserting the first moment equations into the zero moment equation, to arrive at a single partial differential equation, which contains the zero moment ϕ and the second moment P of the radiance field as unknowns. In order to be able to solve this partial differential equation for ϕ, it has to be closed by eliminating the second moment P as an unknown. Finding or making an educated guess for the closure P is called the moment closure problem. In this section, we will outline the closure for the classical diffusion approximation and revise it in chapter 4 to introduce more sophisticated closures.

When trying to establish an educated guess for P, there is some information available that could be used, namely the lower moments. The zero moment ϕ expresses the power at **x**. That is, the total amount of energy arriving from all directions. With that, the radiance field can be factorized into the total power ϕ, and its distribution over solid angle $\hat{L}$:

$$L(\mathbf{x},\boldsymbol{\omega}) = \hat{L}(\mathbf{x},\boldsymbol{\omega})\phi(\mathbf{x})$$

The total power ϕ is an unknown, which shall not be eliminated. Therefore, an educated guess for how this power is distributed over solid angle will be made. This is given by the normalized radiance field $\hat{L}$. It is a probability distribution and its first moment is called the normalized flux $\widehat{\mathbf{E}}$ which is important with more advanced diffusion theories:

$$\widehat{\mathbf{E}}(\mathbf{x}) = \hat{L}_1(\mathbf{x}) = \int_\Omega \frac{L(\mathbf{x},\boldsymbol{\omega})}{\phi(\mathbf{x})}\boldsymbol{\omega}\mathrm{d}\boldsymbol{\omega} = \frac{1}{\phi(\mathbf{x})}\int_\Omega L(\mathbf{x},\boldsymbol{\omega})\boldsymbol{\omega}\mathrm{d}\boldsymbol{\omega} = \frac{\mathbf{E}(\mathbf{x})}{\phi(\mathbf{x})}$$

Seperating the radiance field into power and its distribution over solid angle, allows us to factorize its second moment, the unknown P, in the same fashion:

$$\begin{aligned} P(\mathbf{x}) &= \int_\Omega \hat{L}(\mathbf{x},\boldsymbol{\omega})\phi(\mathbf{x})N_2\mathrm{d}\boldsymbol{\omega} \\ &= \int_\Omega \hat{L}(\mathbf{x},\boldsymbol{\omega})N_2\mathrm{d}\boldsymbol{\omega}\phi(\mathbf{x}) \\ &= \hat{L}_2(\mathbf{x},\boldsymbol{\omega})\phi(\mathbf{x}) \end{aligned}$$

The second moment of the radiance field is expressed as the second moment of the angular radiance distribution, multiplied by the power ϕ at **x**. This means, that finding a closure at the end results in finding or estimating a spherical distribution function. The second moment of that estimated distribution is called the Eddington tensor T:

$$T(\mathbf{x}) \approx \hat{L}_2(\mathbf{x},\boldsymbol{\omega}) = \frac{P(\mathbf{x})}{\phi(\mathbf{x})} = \widehat{P}(\mathbf{x}) \tag{4.40}$$

The Eddington tensor is used to distribute the known power ϕ over angle and approximate P:

$$T(\mathbf{x})\phi(\mathbf{x}) \approx P(\mathbf{x})$$

The key assumption behind the classical diffusion approximation is that the radiance field L is constant and does not change for different angles $\boldsymbol{\omega}$:

$$L(\mathbf{x},\boldsymbol{\omega}) = L(\mathbf{x})$$

This assumption allows for finding an estimate T for the second moment of the radiance distribution. Unit power $\phi = 1$ is assumed:

$$1 = \phi(\mathbf{x}) = \int_\Omega L(\mathbf{x},\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}$$

Since L is constant over solid angle ($L(\mathbf{x},\boldsymbol{\omega}) = L(\mathbf{x})$), it can be extracted out of the integral to get:

$$1 = \phi(\mathbf{x}) = L(\mathbf{x}) \int_\Omega \mathrm{d}\boldsymbol{\omega} \implies L(\mathbf{x}) = \frac{1}{4\pi}$$

The constant radiance field is used to compute the components of the radiative pressure tensor:

$$P(\mathbf{x}) = \int_\Omega L(\mathbf{x}) N_2 \mathrm{d}\boldsymbol{\omega} = \frac{1}{4\pi} \int_\Omega N_2 \mathrm{d}\boldsymbol{\omega} = \frac{1}{4\pi}\frac{4\pi}{3}\mathbf{I} = \frac{1}{3}\mathbf{I}$$

Since the unit power $\phi(\mathbf{x}) = 1$ was assumed without loss of generality, the result is:

$$T(\mathbf{x})\phi(\mathbf{x}) = P(\mathbf{x}) \implies T(\mathbf{x}) = \frac{1}{3}\mathbf{I}$$

Inserting $P = T\phi = \frac{1}{3}\mathbf{I}\phi$ into equation 4.39, gives the diffusion equation for anisotropic emission sources

$$\nabla\left(-\frac{1}{3\sigma_t'(\mathbf{x})}\nabla\phi(\mathbf{x})\right) = -\phi(\mathbf{x})\sigma_a(\mathbf{x}) + Q_0(\mathbf{x}) + \nabla Q_1(\mathbf{x}) \tag{4.41}$$

and becomes the popular diffusion approximation formula for isotropic emission sources ($Q_1 = \mathbf{0}$):

$$\nabla(D\nabla\phi(\mathbf{x})) = -\phi(\mathbf{x})\sigma_a(\mathbf{x}) + Q_0(\mathbf{x}) \tag{4.42}$$

with the classic linear diffusion coefficient

$$D = -\frac{1}{3\sigma_t'(\mathbf{x})}. \tag{4.43}$$

Inserting the isotropic second moment into the first moment expansion of the radiative transfer equation (equation 4.37) gives an expression for the flux-vector, as defined by the diffusion approximation theory:

$$\mathbf{E}(\mathbf{x}) \approx -D(\mathbf{x})\nabla\phi(\mathbf{x}) \tag{4.44}$$

This allows for some intuition about the diffusion approximation. Instead of depending on the global radiance field L, the flux-vector depends on the local gradient of the fluence to determine the transport of radiative energy at position $\mathbf{x}$. The problem is now uniquely defined and can be solved with standard solvers. In the next section, we discuss how results from diffusion approximation compare against the results from the P_N-method.

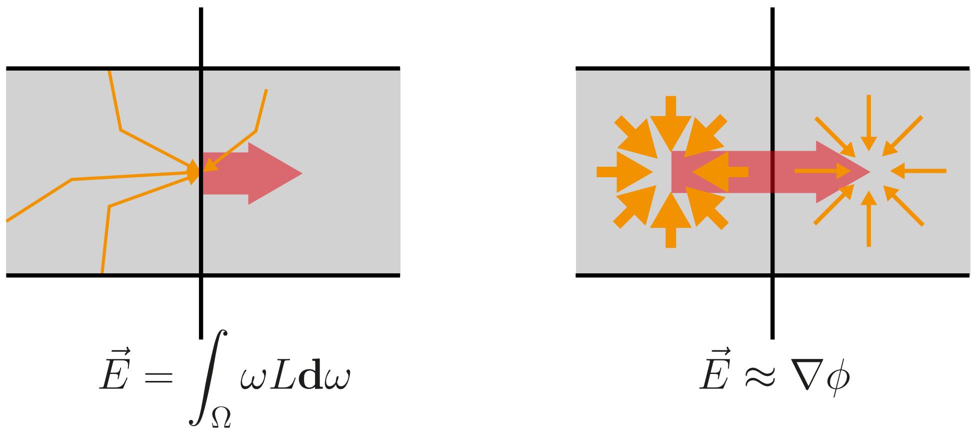

$\vec{E} = \int_{\Omega} \omega L \mathbf{d}\omega$
$\vec{E} \approx \nabla\phi$

Figure 4.1: *Ficks first law of diffusion (visualized on the right) states that the diffusive flux can be related to the difference in concentration (of fluence ϕ), allowing to approximate a global quantity (left) by a local quantity. This is the fundamental idea behind diffusion approximation.*

4.4 Diffusion Approximation

The previous section derived a diffusion equation from the P_1-equations by assuming an isotropic distribution of the radiance field for the moment closure problem. We solve this diffusion equation very efficiently using a multigrid solver similarly to Stam et al. [82] and discuss the results in this section.

The multigrid solver is implemented in a straightforward fashion. However, since Stam only mentions the use of a multigrid solver very briefly, we give more details on its implementation in appendix B.

In terms of the rendering integration, it is important to note that the diffusion approximation only gives the solution to the zero moment $L^{0,0}$, which is the same as the fluence ϕ. Higher moments are not needed, if the phase function is isotropic, as discussed in section 3.6 and shown in equation 3.43. However, in the case of an anisotropic phase function, equation 3.43 applies, which requires the evaluation of $\widehat{L}_m$, the full reconstruction of the truncated spherical harmonics expansion of the radiance field. For the diffusion approximation, this was derived in section 4.1 and resulted in equation 4.30. While the first term is straightforward to evaluate using the zero moment from the diffusion solve, the second term requires the flux vector. This needs first to be computed using the definition, which had been derived using the isotropic moment closure in section 4.3, where equation 4.44 is used. With the definition, the first moment can be reconstructed from the gradient of the zero moment and used to evaluate $\widehat{L}_m$.

4.4.1 Point Source Problem

The solver is first run on the point source problem described in section 3.7.1 and compared against the results from the P_N-method.

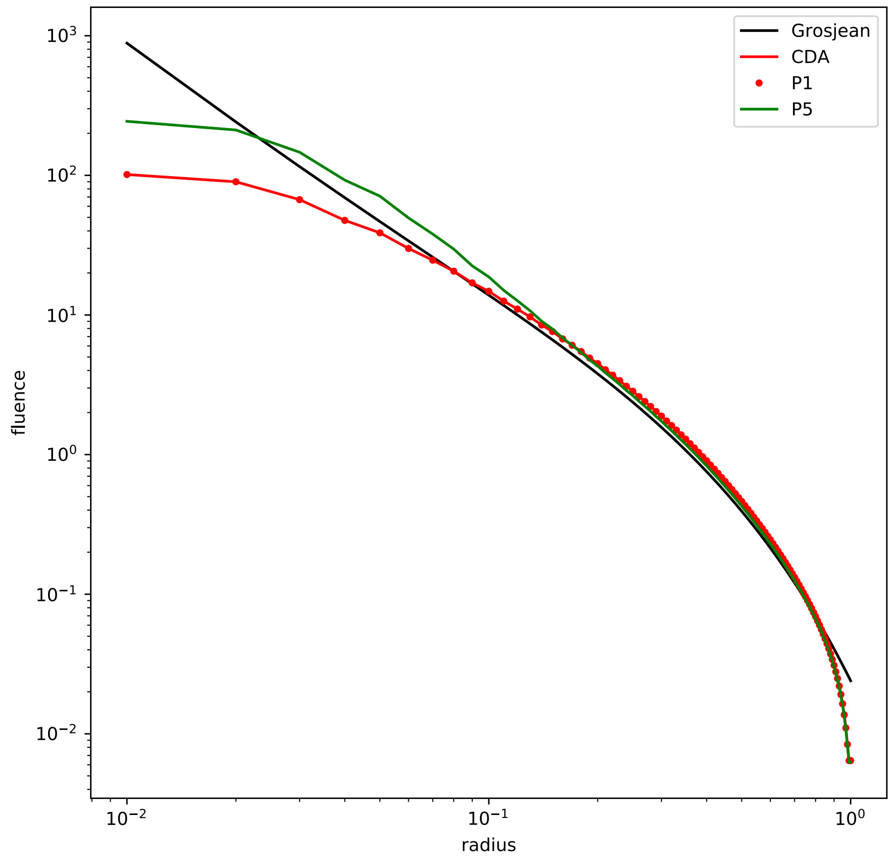
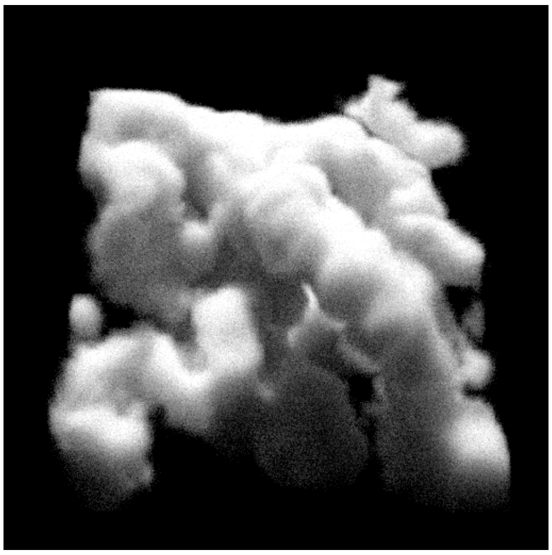
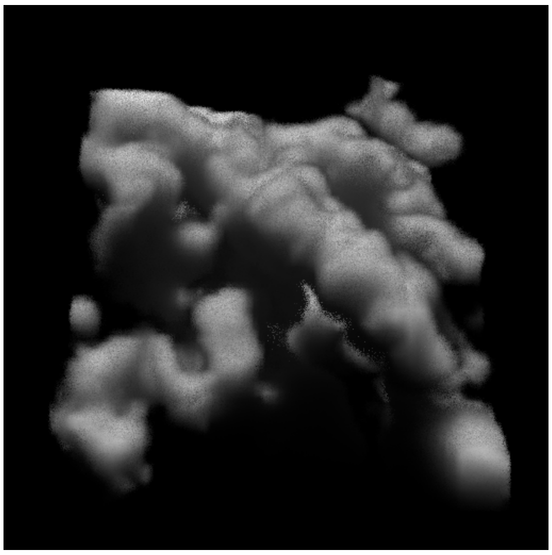


Figure 4.2: *Solution of the classical diffusion approximation (CDA) for the point source problem (red line) compared against ground truth (black), P_1 (red dots) and P_5 (green).*

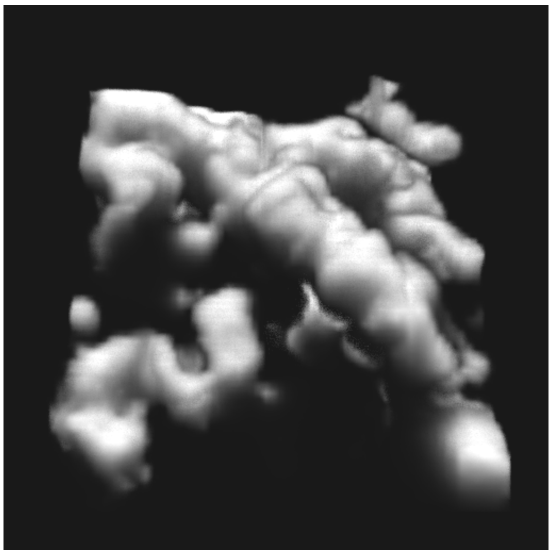
As expected, the accuracy of the P_N-result is better than for the diffusion approximation as soon as the truncation order of the P_N-method is increased. However, for the truncation order of $N = 1$, identical results are shown. This is explained by the fact that the diffusion approximation is derived by collapsing the P_1 equations by using substitution. Therefore, the diffusion equation is mathematically equivalent to the P_1-equations and consequently is satisfied by the exact same solution. This result further validates that the P_N-solver has been implemented correctly.



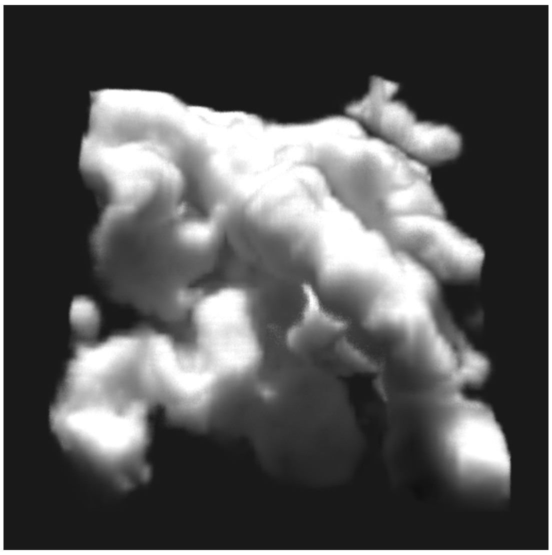
(a) *Pathtraced*



(b) *Diffusion Approximation*



(c) P_1



(d) P_5

Figure 4.3: *Diffusion approximation results for the procedural cloud dataset (multiple scattered light) compared against Monte-Carlo reference,* P_1 *and* P_2.

4.4.2 Procedural Cloud

Finally, the multigrid solver for diffusion is run on the procedural cloud problem from section 3.7.3. The presence of vacuum regions in the dataset caused the P_N-method not to converge, but it was still possible to run iterations that would reduce the residual error albeit slowly. However, for the diffusion approximation, the presence of vacuum regions causes the method to break down completely, as the diffusion coefficient (equation 4.43) requires division by the extinction coefficient and therefore produces a division by zero. This aspect is easily confused, when reading the neutron transport literature. Papers such as Hansen et al. [39] state that the normal form of the P_N-equations can deal with voids. This is true in the sense that it does not produce a division by zero directly. However, it still is not convergent.

As with the point source problem, it is shown that the P_1-results, match the diffusion results while higher truncation order produces more accurate results for the P_N-method. In particular, the diffusion approximation (or P_1) fail to reproduce the illumination of the indirectly illuminated region at the bottom of the dataset.

4.5 Moment Closure Problem Revisited

In section 4.3, it was established that a choice for the second moment of the radiance distribution was required to close the system of equations. It was found by collapsing the moment expansion of the radiative transfer equation truncated after the first moment. The classical diffusion approximation was derived by assuming an isotropic radiance distribution for the second moment. In this section, we take a closer look at the moments and carve out an important feature, which explains the inaccuracy of classical diffusion approximation in certain scenarios.

The distribution of power ϕ over solid angle was given by the normalized radiance $\widehat{L}$. Since $\widehat{L}$ is a probability distribution, it has to integrate to one, which is verified by

$$\int_\Omega \hat{L}(\mathbf{x},\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} = \int_\Omega \frac{L(\mathbf{x},\boldsymbol{\omega})}{\phi(\mathbf{x})}\mathrm{d}\boldsymbol{\omega} = \frac{1}{\phi(\mathbf{x})}\int_\Omega L(\mathbf{x},\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} = 1 = \hat{L}_0$$

Further, the normalized radiance $\widehat{L}$ has to be non-negative by being a probability distribution:

$$\hat{L}(\mathbf{x},\boldsymbol{\omega}) \geq 0 \qquad \text{for all unit directions } \boldsymbol{\omega}\ ,$$

which is true by definition:

$$L(\mathbf{x},\boldsymbol{\omega}) \geq 0 \implies \int_\Omega L(\mathbf{x},\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} = \phi(\mathbf{x}) \geq 0 \implies \frac{L(\mathbf{x},\boldsymbol{\omega})}{\phi(\mathbf{x})} = \widehat{L}(\mathbf{x},\boldsymbol{\omega}) \geq 0$$

Akhiezer [1] established, that the probability distribution constraints result in a set of inequalities between moments of the constrained function. In particular, the non-negativity constraint on the radiance field L and its normalized distribution imposes a constraint between the zero moment and the first moment. In case of the radiance field, this constraint can be derived considering the dot product between direction vectors $\boldsymbol{\omega}'$ and $\boldsymbol{\omega}$, both of unit length. This results in the equation:

$$-1 \leq \boldsymbol{\omega}' \cdot \boldsymbol{\omega} \leq 1 \implies 0 \leq 1 - \boldsymbol{\omega}' \cdot \boldsymbol{\omega} \leq 1 \implies \int_\Omega \big(1 - \boldsymbol{\omega}' \cdot \boldsymbol{\omega}\big) L(\mathbf{x}, \boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} \geq 0$$

which can be rearranged into:

$$\begin{aligned} \int_\Omega \big(1 - \boldsymbol{\omega}' \cdot \boldsymbol{\omega}\big) L(\mathbf{x}, \boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} \geq 0 \Leftrightarrow \\ \int_\Omega L(\mathbf{x}, \boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} - \int_\Omega \boldsymbol{\omega}' \cdot \boldsymbol{\omega} L(\mathbf{x}, \boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} \geq 0 \Leftrightarrow \\ \int_\Omega L(\mathbf{x}, \boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega} - \boldsymbol{\omega}' \cdot \int_\Omega L(\mathbf{x}, \boldsymbol{\omega}) \boldsymbol{\omega} \mathrm{d}\boldsymbol{\omega} \geq 0 \Leftrightarrow \\ \phi(\mathbf{x}) - \|\mathbf{E}(\mathbf{x})\| \geq 0 \Leftrightarrow \\ \phi(\mathbf{x}) \geq \|\mathbf{E}(\mathbf{x})\| \end{aligned}$$

This constraint states, that the total power at position $\mathbf{x}$ must never exceed the length of the flux-vector. The flux-vector is found by integrating the radiance field over solid angle multiplied by a weighting factor between $[-1,1]$ (the direction vector component). From this it follows, that it must never be larger than the fluence, which is the integral over the radiance field itsself.

A very similar constraint can be derived for the normalized radiance $\widehat{L}$, by following the steps above we obtain the following:

$$1 \geq \left\| \frac{\mathbf{E}(\mathbf{x})}{\phi(\mathbf{x})} \right\| \tag{4.45}$$

In the case of the classical diffusion approximation $\mathbf{E} \approx -D\nabla\phi$ (see equation 4.44). Inserting this into equation 4.45 gives:

$$1 \geq \left\| \frac{\nabla\phi(\mathbf{x})}{3\sigma_t(\mathbf{x})\phi(\mathbf{x})} \right\| \tag{4.46}$$

It shows, that with the classical diffusion approximation, this constraint can be violated when the extinction coefficient σ_t becomes very small, i.e. in the presence of a very thin medium. It breaks down completely in case of vacuum $\sigma_t = 0$. The constraint can also be violated, if the fluence gradient becomes very large compared to $\sigma_t\phi$. This happens very close to point light sources or near strong density gradients in the medium.

Violation of the constraint above implies, that for the classical diffusion approximation, the flux vector is much larger than what is physically plausible. Since the flux vector governs light transport with diffusion, the literature often refers to classical diffusion as to suffer from unphysical fluxes or unphysical light transport.

A Transport Regime Measure

The right hand side of equation 4.46 is an important measure, which allows to detect at every point within the domain, where and to what extent the diffusion approximation fails. For this, only local information about the moments and extinction coefficient is needed. It is an important building block for the technique presented in this chapter, which seeks to improve the accuracy of the diffusion approximation.

The flux-limit in equation 4.46 can be factorized to show a clearer intuition

$$\left\| \frac{\nabla\phi(\mathbf{x})}{3\sigma_t(\mathbf{x})\phi(\mathbf{x})} \right\| = \frac{1}{3} l(\mathbf{x}) \frac{\|\nabla\phi(\mathbf{x})\|}{\phi(\mathbf{x})} \tag{4.47}$$

It consists of two factors, of which the first is the mean free path l (in units of length), which is the inverse of the extinction coefficient and parameterizes the medium. If the mean free path is small and approaches zero, photons travel only very short distances on average before they encounter another interaction with the medium. In this case, the medium is very dense. If the mean free path is large, the medium is very thin and photons travel long distances before they encounter another interaction with the medium. In the case of vacuum, where no medium is present, photons travel unhindered and never interact. In this case, the mean free path is infinite.

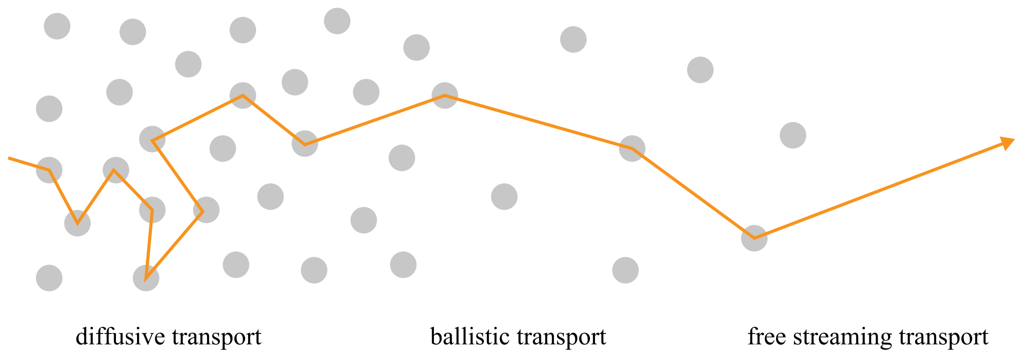


Figure 4.4: *Intuition behind parameterizing the transport regime according to mean free path (average distance between two consecutive scattering events). Small mean free paths imply higher medium density, while larger mean free paths imply lower density or vacuum in case of infinity.*

The second factor in equation 4.47 is the ratio between the length of the flux-vector (according to diffusion) and the zero moment. If the magnitude of the flux-vector is small, while the total power ϕ is large, the incoming radiance is distributed equally over solid angle, which means that light is coming uniformly from all directions. If the flux-vector magnitude is large in relation to the total power, the energy is concentrating in certain directions. In the extreme case, when the total power is equal to the flux-vector magnitude, the light comes only from a single direction.

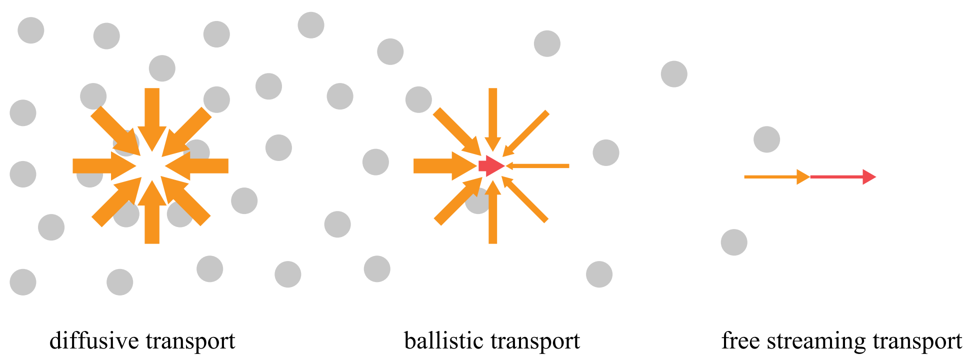


Figure 4.5: *The intuition behind parameterizing the transport regime according to the ratio between the total amount of light arriving at a point and the directionality given by the magnitude of the flux vector (red in the figure). A small ratio results from small directionality in relation to large amounts of light arriving. This implies a uniform distribution of light coming from all directions, which is the case for diffusive transport. The ballistic transport regime is characterized by increased directionality in relation to the amount of light. In free streaming transport, the magnitude of the flux vector equals the amount of light arriving. This is the case when light comes from a single direction.*

The two factors in equation 4.47 parameterize the transport regime according to the medium and the distribution of incoming light over solid angle respectively. Combined, they are a local measure for the transport regime in general (as perceived by diffusion). If the medium is very dense (small mean free path) and light arrives equally distributed from all directions (small moment ratio), diffusive transport is dominating and the flux-limit is not violated. The diffusion approximation is a good approximation in these scenarios. If the medium is very thin (large mean free path) and the moment ratio approaches one, we have ballistic transport. In the case of vacuum, when the moment ration is exactly one, we have the free streaming limit or vacuum, where light travels infinite distances unhindered.

The $1/3$ factor in equation 4.47 is ignored. We define the ratio R as a transport measure according to the diffusion approximation:

$$R(\mathbf{x}) = \frac{\|\nabla\phi(\mathbf{x})\|}{\sigma_t(\mathbf{x})\phi(\mathbf{x})} \qquad \begin{array}{l} R(\mathbf{x}) \to 0 : \text{diffussive transport} \\ R(\mathbf{x}) \to 1 : \text{streaming transport} \end{array} \tag{4.48}$$

In section 4.3, the diffusion approximation was derived by assuming an isotropic distribution of radiance and therefore the same requirement in the fux-limit constraint is seen where a small moment ratio is required and implies that light is arriving equally from all directions.

The idea of flux-limited diffusion is to avoid violation of the flux-limit and consequently prevent unphysical transport. The key idea behind flux-limited diffusion is to not only assume isotropic distribution but also allow streaming transport and use the measure R, to locally realize the right mix between diffusive and streaming transport. The next important building block is therefore the question of how the diffusion equation looks like in the presence of streaming transport. This is outlined in the next section. Section 4.7 will then develop flux-limited diffusion as a combination with classical diffusion.

4.6 Streaming Limit Approximation

In the previous section, the transport measure R was introduced, which contains the factor $||\nabla\phi||/\phi$. This factor approaches one as transport becomes less diffusive and enters the streaming regime. In the case of $||\nabla\phi||/\phi = 1$, the length of the flux-vector matches the amount of total power. In this case, light is coming from a single direction. The radiance distribution of this configuration is given as

$$\hat{L}(\boldsymbol{\omega}) = \delta_\Omega(\boldsymbol{\omega},\mathbf{n}). \tag{4.49}$$

where δ_Ω is the angular Dirac delta distribution into direction $\mathbf{n}$. The spatial dependency is omitted and not relevant for this discussion. The moments of this distribution are:

$$\hat{L}_0(\boldsymbol{\omega}) = \int_\Omega \delta_\Omega(\boldsymbol{\omega},\mathbf{n})\mathrm{d}\boldsymbol{\omega} = 1 \tag{4.50}$$

$$\hat{L}_1(\boldsymbol{\omega}) = \int_\Omega \delta_\Omega(\boldsymbol{\omega},\mathbf{n})\boldsymbol{\omega}\mathrm{d}\boldsymbol{\omega} = \mathbf{n} \tag{4.51}$$

$$\hat{L}_2(\boldsymbol{\omega}) = \int_\Omega \delta_\Omega(\boldsymbol{\omega},\mathbf{n})\boldsymbol{\omega}_i\boldsymbol{\omega}_j\mathrm{d}\boldsymbol{\omega} = \mathbf{n}_i\mathbf{n}_j \tag{4.52}$$

The zero moment expresses that all power given by ϕ comes from a single direction. The second equation shows, that the first moment of a delta distribution is identical to the vector which defines that distribution ($\mathbf{n}$ in our case). It shall be concluded that the length of the flux-vector equals the zero moment and its direction is $\mathbf{n}$ (all under the assumption of a delta radiance distribution):

$$\mathbf{E} = \hat{L}_1\phi = \mathbf{n}\phi \implies ||\mathbf{E}|| = \phi \tag{4.53}$$

$$\implies \mathbf{E} \parallel \mathbf{n} \qquad (\mathbf{E} \text{ and } \mathbf{n} \text{ are parallel}) \tag{4.54}$$

The second moment equation shows that the second moment is found by the outer product of the defining vector. Therefore, assuming a delta distribution results in the following Eddington tensor (equation 4.40):

$$T_{ij} = \mathbf{n}_i \mathbf{n}_j \tag{4.55}$$

The form of the Eddington tensor for a delta distribution of radiance in angular domain is given. In the end, this tensor is to be used to approximate the second moment of the radiance field P in equation 4.39 by $T\phi \approx P$. However, this way the direction $\mathbf{n}$ is introduced as another unknown. The next step in the derivation is, to reformulate T and eliminate of the unknown $\mathbf{n}$, which is possible under certain assumptions.

Inserting the approximation $T\phi$ into the flux-vector definition (equation 4.37) and assuming an isotropic emission Q gives:

$$\mathbf{E} = -\frac{1}{\sigma_t'}\mathrm{div}(T\phi)$$

The key assumption used is that the spatial variations of T can be neglected. Fundamentally, the approach is to assume, that the radiance field L can be separated into a product of two functions: one depending on the angle and another function depending on the position. This allows expressing the divergence of the second moment as a matrix product with a gradient vector:

$$\mathbf{E} = -T\left(\frac{1}{\sigma_t'}\nabla\phi\right) \tag{4.56}$$

Then, the normalized flux will be addressed, which is given by scaling the flux-vector with $1/\phi$:

$$\widehat{\mathbf{E}} = \frac{\mathbf{E}}{\phi} = -T\underbrace{\left(\frac{\nabla\phi}{\sigma_t'\phi}\right)}_{=\mathbf{R}} \tag{4.57}$$

Equation 4.51 showed that $\mathbf{n}$ has the same direction as the flux-vector $\mathbf{E}$. This indicates that the result of the transformation T will be a vector, that is parallel to $\mathbf{n}$. Since the Eddington tensor has been constructed with $T_{ij} = \mathbf{n}_i\mathbf{n}_j$, we can show that $\mathbf{n}$ is the only eigenvector of T with eigenvalue $||\mathbf{n}||$.

Consider an arbitrary vector $\mathbf{u}$. The construction of $T_{ij} = \mathbf{n}_i\mathbf{n}_j$ can be also expressed as the outer product between $\mathbf{n}$ with itsself. Transforming $\mathbf{u}$ with T gives:

$$T\mathbf{u} = \left(\mathbf{n}\cdot\mathbf{n}^T\right)\mathbf{u} = \lambda\mathbf{u} \tag{4.58}$$

We can see that transforming any vector with T results in a scaling of that vector with an unknown value λ. Therefore this value must be an eigenvalue of T and all other eigenvalues of T need to be zero. We can further derive an expression for the eigenvalue by forming the dot product between **n** and $T\mathbf{u}$:

$$\mathbf{n}\cdot T\mathbf{u} = \mathbf{n}\cdot(\mathbf{n}\cdot\mathbf{n})\mathbf{u} = \mathbf{n}\lambda\mathbf{u} \implies \lambda = \mathbf{n}\cdot\mathbf{n} = ||\mathbf{n}||^2 \tag{4.59}$$

We see that **n** is the only eigenvector of T, with an eigenvalue greater than zero. It can be concluded that (under the assumption of negligible spatial variation of T) the transformation T is applying a scaling operation and the dimensionless gradient **R** is parallel to E. Therefore the product of tensor T is expressed with **R** being a scaling operation and equation 4.57 then becomes:

$$\widehat{\mathbf{E}} = -\lambda\left(\frac{\nabla\phi}{\sigma_t'\phi}\right) \tag{4.60}$$

Hence, under the above assumption about the spatial variation of T, the normalized flux-vector $\hat{L}_1$ could be expressed with respect to the zero moment ϕ and its spatial derivatives. What remains to be done is to find an expression for the eigenvalue λ. Then an expression for **E** (by using $\mathbf{E} = \hat{L}_1\phi$) would be found, that has no self-dependence and therefore can be used to substitute **E** into the zero moment equation.

Apparently, finding λ is easy in the case of a delta distribution of radiance. In that case it is known from equation 4.53, that $||\mathbf{E}|| = \phi$ and therefore $||\mathbf{E}/\phi|| = 1$. This requires that:

$$||\mathbf{E}|| = \left|\left|-\lambda\left(\frac{\nabla\phi}{\sigma_t'\phi}\right)\right|\right| = 1 \implies \lambda = \frac{\sigma_t'\phi}{||\nabla\phi||}$$

Note that is part of the vector norm, which would make the sign of lambda ambiguous (it could be positive or negative: in both cases, the length of the normalized flux-vector would be one). However, as mentioned earlier, the way $T = \mathbf{n}_i\mathbf{n}_j$ is defined allows the conclusion that λ is the only eigenvalue of T and that it must be greater than one. This gives the final expression for the flux-vector **E**:

$$\mathbf{E} = \widehat{\mathbf{E}}\phi = -\lambda\left(\frac{\nabla\phi}{\sigma_t'\phi}\right)\phi = -\frac{\nabla\phi}{||\nabla\phi||}\phi,$$

which states that the flux-vector is the unit vector pointing into the direction of the gradient of ϕ scaled by ϕ itself. Inserting this into the collapsed P_1-equation (equation 4.39) gives:

$$\nabla\cdot\left(\frac{\nabla\phi}{||\nabla\phi||}\phi\right) = \sigma_a\phi - Q_0 \tag{4.61}$$

If we consider a purely scattering medium without emission ($\boldsymbol{\omega}_a = 0$ and $Q_0 = 0$) and define $\mathbf{u} = \frac{\nabla\phi}{||\nabla\phi||}$ to be the unit fluence gradient vector field, then it becomes apparent that this equation resembles some form of a steady state advection equation:

$$\nabla\cdot(\mathbf{u}\phi) = 0 \tag{4.62}$$

In case of a radiance delta distribution in angular domain and with the assumption of negligible spatial variation of T, the moment equation expresses how the zero moment quantity ϕ is moved around by its normalized gradient.

This section highlights, that streaming transport is best represented by advection in case of the two-term expansion of the radiative transfer equations. The core idea behind flux-limited diffusion is, to mix diffusive transport and advective transport, depending on local properties at $\mathbf{x}$. This is developed in the next section.

4.7 The Variable Eddington Factor

In the two previous sections, diffusion theories have been derived for the two extreme limits of transport. That is purely isotropic radiance, which happens in the limit of thick, highly scattering media or a delta distribution, which in turn happens in the limit of very transparent, low order scattering media. This section will introduce the variable Eddington Factor, which has its origins in the astrophysics domain (Brunner [8]). It serves as a conceptual framework in which flux-limited diffusion is embedded. While diffusive transport has been related to a diffusion equation, streaming transport has been shown to relate to advection. The Variable Eddington Factor allows for realizing and mixing both types of transport within a domain.

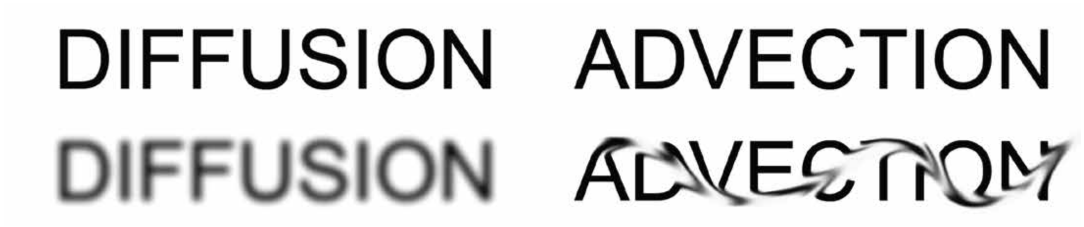


Figure 4.6: *Illustration of diffusive (left) and advective (right) transport of particles. With diffusion, particles spread locally depending on the gradient of the particle distribution, which for flux-limited diffusion is the zero moment, the fluence. Advection is the movement of particles along a directed force field. For flux-limited diffusion, this force field is defined by the first moment, the flux vector.*

Classical diffusion assumed an isotropic distribution of radiance for the second moment, while pure streaming transport assumed a delta distribution. The Variable Eddington Factor theory assumes a radiance distribution, which is rotationally symmetric around a dominant vector $\mathbf{n}$. This assumption allows deriving a form for T, which represents isotropic distribution as well as a pure delta distribution and those forms inbetween, which are rotationally symmetric around a dominant direction **x**.

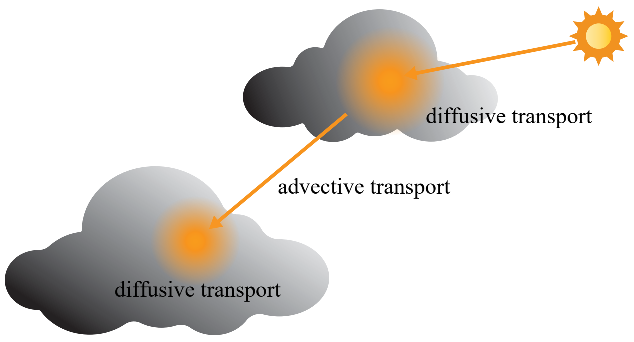


Figure 4.7: *Radiative transfer in typical scenes contains a mix of transport regimes as shown in this figure. Diffuse transport in dense media with high absorption and advective transport in thin media. The idea behind the Variable Eddington Factor is to express radiative transfer as a mix between those two extreme transport modes.*

The Variable Eddington Factor form of T is derived by considering a principal direction of transport, given by the vector $\mathbf{n}$ of unit length. An important assumption is that the radiance distribution will be radially symmetric around this direction. This means, that the value of $\hat{L}$ will be invariant to rotation about axis $\mathbf{n}$. It follows, that its first and second moment $\hat{L}_1$ and $\hat{L}_2$ will also be invariant to rotation about $\mathbf{n}$. If one approximates $\hat{L}_2$ using the Eddington tensor T with $\hat{L}_2 \approx T\phi$, it can be concluded that $\mathbf{n}$ will be an eigenvector of T with eigenvalue χ:

$$T\mathbf{n} = \chi\mathbf{n}$$

The plane perpendicular to $\mathbf{n}$ is an eigenspace of T. By requiring that all eigenvalues sum up to one, the eigenvalues of the two eigenvectors spanning that plane by distributing the remaining eigenvalue $1-\chi$ evenly between both is expressed as:

$$\frac{1}{2}(1-\chi)\mathbf{I} \tag{4.63}$$

From the free streaming limit case (section 4.6), it is known that $\chi = 1$, when the radiance distribution becomes a delta distribution (equation 4.50 and 4.51). In that case, the eigenvalues associated with the eigenvectors perpendicular to $\mathbf{n}$ become zero and the term above vanishes.

Tensor diagonalization allows for explicitly adding the eigenvector $\mathbf{n}$ to the matrix by adding $\mathbf{n}_i\mathbf{n}_j$ (with $\mathbf{n}$ being of unit length). The scaling of its coefficients is found by considering that the term in equation 4.63 introduces a cartesian eigenspace represented by three base eigenvectors. The sum of their eigenvalues will be $3/2(1-\chi)$.

Since eigenvalues of the final tensor T need to add up to one, the eigenvalue associated with the matrix $\mathbf{n}_i\mathbf{n}_j$ will be $\left(1-\frac{3}{2}(1-\chi)\right)$. This gives for T:

$$\begin{aligned} T &= \frac{1}{2}(1-\chi)\mathbf{I}+\left(1-\frac{3}{2}(1-\chi)\right)\mathbf{n}\otimes\mathbf{n} \\ &= \frac{1}{2}(1-\chi)\mathbf{I}+\frac{1}{2}(3\chi-1)\mathbf{n}\otimes\mathbf{n} \end{aligned} \tag{4.64}$$

This form of T is called the Variable Eddington Tensor (VET). It can be understood as an *interpolation* between an isotropic distribution tensor ($1/3\mathbf{I}$) and a delta distribution tensor ($\mathbf{n}_i\mathbf{n}_j$). The variable $1/3 \leq \chi \leq 1$ is the interpolation variable and is called the variable Eddington factor (VEF). Theories, which respect this structure of the Eddington tensor are referred to as theories under the variable Eddington factor formalism (VEF-formalism).

If $\chi = 1/3$, then equation 4.64 will result in the isotropic distribution tensor, which is the base assumption for classical diffusion:

$$\frac{1}{2}\frac{2}{3}\mathbf{I}+\frac{1}{2}\left(\frac{3}{3}-1\right)\mathbf{n}\otimes\mathbf{n} = \frac{1}{3}\mathbf{I}+0\mathbf{n}\otimes\mathbf{n} = \frac{1}{3}\mathbf{I} \tag{4.65}$$

For $\chi = 1$, equation 4.64 produces the Eddington tensor, which was derived for the pure streaming limit distribution, where all light comes from a singular direction:

$$\frac{1}{2}0\mathbf{I}+\mathbf{n}\otimes\mathbf{n} = \mathbf{n}\otimes\mathbf{n} \tag{4.66}$$

The Eddington factor χ can also be interpreted as a measure of anisotropy of the radiance distribution $\widehat{L}$ with respect to direction $\mathbf{n}$. It can be expressed in terms of the radiance distribution by the squared mean cosine (given by Levermore [55]):

$$\chi = \int_{S^2} (\boldsymbol{\omega}\cdot\mathbf{n})^2 \hat{L}(\mathbf{x},\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{4.67}$$

With the variable Eddington factor approach, only a function for the interpolation variable χ (the actual Eddington factor) needs to be identified. This function should have 1/3 and 1 as its limits. With such an interpolation function, the limit transport cases (diffusion and free streaming) will be a subset of any theory, which adheres to the variable Eddington factor concept.

The Eddington factor framework, sets up the Eddington factor and the limits of its parameterization, χ. The specific expression for this factor is not defined and it is clear that there are many options for the function χ, which respect the given function limits. This is why there is such a rich variety of theories in other domains, which all propose their own version of that interpolation function. Some of them are ad-hoc schemes (Bowers et al. [6], Kershaw [47] and Larsen et al. [54]), which are derived from heuristics, while others have a clear connection to transport theory (Levermore at al. [56]) or are derived from entropy theory (Minerbo [67]).

The general strategy for all these different variable Eddington factor theories is:

1. Find a model or theory, from which a certain radially symmetric form of the radiance distribution $\hat{L}$ about the normalized flux-vector$\mathbf{E}/\phi$ can be found or justified.
2. Then derive an expression for $\chi(\mathbf{E}/\phi)$ from the model for $\hat{L}$, which can be used to construct T. Further assumptions are applied, to be able to express $\mathbf{E}$ in terms of $\nabla\phi$. This is required in order to not have T depend on the flux-vector directly as it still needs to be possible to resolve equation 4.37 for the flux-vector.

In this thesis, results for different theories have been presented and implemented, but only flux-limited diffusion from Levermore et. al [56] was discussed, since it is the most popular and also has the strongest connection to transport theory. Discussing all other theories is beyond the scope of this thesis and also not really necessary, as it is shown in section 4.10 that the particular choice of theory is not of significant importance for applications in computer graphics.

4.8 Flux-limiters

Levermore et. al [56] construct their theory by starting from the time-dependent form of the radiative transfer equation. They further assume a constant phase function $f_p = 1/(4\pi)$:

$$\frac{1}{c}\frac{\partial(\phi\hat{L})}{\partial t}+(\boldsymbol{\omega}\cdot\nabla)(\phi\hat{L}) = -\sigma_t\phi\hat{L}+\frac{1}{4\pi}\sigma_s\phi+Q \tag{4.68}$$

After applying the moment expansion, the first moment equations are as follows:

$$\frac{1}{c}\frac{\partial\phi}{\partial t}+\nabla\mathbf{E} = -\sigma_a\phi+Q_0 \tag{4.69}$$

As a next step, equation 4.69 is resolved for $\partial\phi/\partial t$ on the left hand side and used to substitute $\partial\phi/\partial t$ in equation 4.68 (after applying the product rule to the time derivative term). By further assuming the absence of self-emission ($Q = 0$) and that the space and time derivatives of the radiance distribution $\hat{L}$ can be neglected, the result is:

$$\big((\boldsymbol{\omega}\cdot\nabla)\phi - \mathbf{f}\cdot\nabla\phi + \sigma_t'\phi\big)\hat{L} = \frac{1}{4\pi}\sigma_t'\phi \tag{4.70}$$

Solving equation 4.70 for $\hat{L}$ gives:

$$\hat{L} = \frac{1}{4\pi}\frac{1}{1+\widehat{\mathbf{E}}\cdot\mathbf{R}-\boldsymbol{\omega}\cdot\mathbf{R}} \tag{4.71}$$

with

$$\mathbf{R} = -\frac{\nabla\phi}{\sigma_t'\phi} \tag{4.72}$$

The definition of the flux-vector in equation 4.57 is used, which was derived in section 4.6:

$$\widehat{\mathbf{E}} = \frac{\mathbf{E}}{\phi} = T\mathbf{R} \tag{4.73}$$

Following the same argument about the radial symmetry of $\hat{L}$ and its relation to the flux-vector and eigenvectors of T, T is replaced with a proportionality function $\lambda(R)$, where $R = \|\mathbf{R}\|$:

$$\widehat{\mathbf{E}} = \lambda(R)\mathbf{R} \tag{4.74}$$

Using this in equation 4.71 gives:

$$\hat{L} = \frac{1}{4\pi}\frac{1}{1+\lambda(R)R^2-\boldsymbol{\omega}\cdot\mathbf{R}} \tag{4.75}$$

By enforcing that $\hat{L}$ integrates to one over the solid angle of the unit sphere, the following expression holds for $\lambda(R)$:

$$\lambda(R) = \frac{1}{R}\left(\coth R-\frac{1}{R}\right) \tag{4.76}$$

Using this result and equation 4.74, yields the following expression for the flux vector:

$$\mathbf{E} = \widehat{\mathbf{E}}\phi = -\frac{1}{\sigma_t'}\lambda(R)\nabla\phi \tag{4.77}$$

Inserting this into the zero moment equation (equation 4.35) gives the flux-limited diffusion equation, a diffusion-type equation of the following form:

$$\nabla\left(\underbrace{-\frac{1}{\sigma_t'}\lambda(R)}_{D_F}\nabla\phi\right) = -\sigma_a\phi+Q_0 \tag{4.78}$$

The flux-limited diffusion coefficient D_F is non-linear and therefore turns the zero moment equation into a non-linear diffusion equation. $\lambda(R)$ is called the flux-limiter. In the diffusion limit, R approaches zero and the flux-limiter approaches $\lambda(R) = 1/3$, which will turn equation 4.78 into the classical diffusion equation for isotropic media.

$$\lim_{R\to 0} D_F = -\frac{1}{3\sigma_t'} \implies \nabla\left(-\frac{1}{3\sigma_t'(\mathbf{x})}\nabla\phi(\mathbf{x})\right) = -\phi(\mathbf{x})\sigma_a(\mathbf{x})+Q_0(\mathbf{x}) \tag{4.79}$$

In the transport limit, R will approach infinity and the diffusion coefficient will cause the equation to become an advection equation as seen in the delta radiance distribution case (equation 4.61):

$$\lim_{R\to\infty} D_F = -\frac{\phi}{\|\nabla\phi\|} \implies \nabla\left(-\phi\frac{\nabla\phi(\mathbf{x})}{\|\nabla\phi\|}\right) = -\phi(\mathbf{x})\sigma_a(\mathbf{x})+Q_0(\mathbf{x}) \tag{4.80}$$

It can be seen that the flux-limited diffusion coefficient will normalize the fluence gradient $\nabla\phi$ to unit length and scale with the total power ϕ. Flux-limited diffusion not only suppresses the flux in the free streaming transport regime, it also saturates it at the appropriate value to ensure correct free propagation (at the level of the approximation).

The flux-limiter introduced by Levermore et al. [56] was shown to relate to the variable Eddington factor approach in a seperate study ([55, 91]) to be:

$$\chi = \lambda(R)+\lambda(R)^2R^2 \tag{4.81}$$

As mentioned in the previous section, the theory sets up the limits, which flux-limiters have to respect. This allows different models and theories to find and justify particular choices of flux-limiters. Table 4.1 presents the most prominent flux-limiters.

Table 4.1: *Various prominent flux-limiters which all respect the Eddington factor limits and represent different flux-limited diffusion theories.*

Flux-limiter	$\lambda(R)$
sum [6]	$(3+R)^{-1}$
max [6]	$\max(3,R)^{-1}$
Kershaw [47]	$2(3+\sqrt{9+4R^2})^{-1}$
Larsen-n [54]	$(3^n+R^n)^{-\frac{1}{n}}$
Levermore-Pomraning [56]	$\frac{1}{R}\left(\coth(R)-\frac{1}{R}\right)$

In this section, a non-linear form of the diffusion equation is derived which respects the flux-limit constraint. The following section will present a novel solver, which solves flux-limited diffusion on a finite difference grid over a given domain.

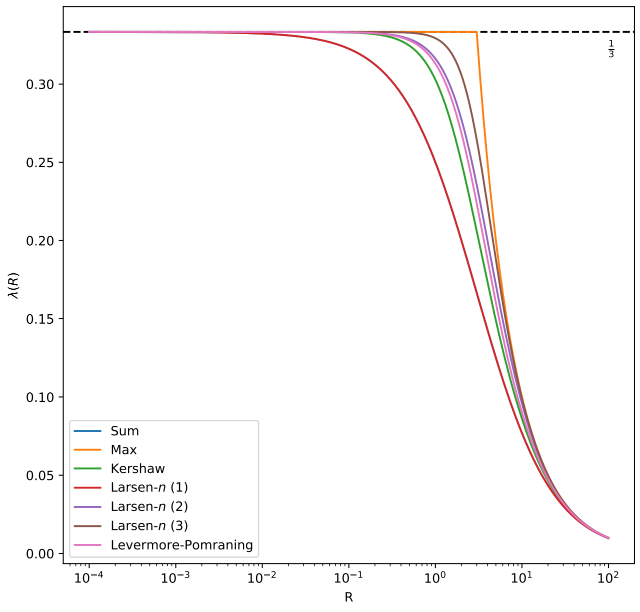


Figure 4.8: *Plot of the various flux-limiters shown in table 4.1.*

4.9 Non-Linear Gauss-Seidel Solver with Successive Over-relaxation

We now introduce a new method for solving the modified diffusion equation, provided by flux-limited diffusion theory. Most prominent with the flux-limited diffusion equation is the non-linear diffusion coefficient. Unfortunately, this non-linearity prevents the application of any components of the solver-framework that was developed in chapter 3 and application of the multigrid solver developed in chapter 4.4, as both are exclusively geared towards coupled systems of linear partial differential equations.

The equation, which has to be solved is the flux-limited diffusion equation (equation 4.78). The flux-limiter λ by Levermore et al. [56] will be used (equation 4.76), which in term depends on the transport measure R (equation 4.48). In summary, the solver developed in this section tries to find a solution for ϕ, which satisfies the following set of equations:

$$\nabla\big(D_F(\mathbf{x})\nabla\phi\big) = -\sigma_a\phi + Q_0 \quad \text{with} \quad D_F(R(\mathbf{x})) = -\frac{1}{\sigma_t'}\lambda(R(\mathbf{x})) \tag{4.82}$$

$$\lambda(R(\mathbf{x})) = \frac{1}{R(\mathbf{x})}\left(\coth R(\mathbf{x}) - \frac{1}{R(\mathbf{x})}\right) \tag{4.83}$$

$$R(\mathbf{x}) = \frac{\|\nabla\phi(\mathbf{x})\|}{\sigma_t(\mathbf{x})\phi(\mathbf{x})} \tag{4.84}$$

This is a diffusion equation with a non-linear diffusion coefficient D_F depending on the flux-limiter λ. The non-linearity arises from the fact that the flux-limiter depends on the solution, which after substitution will create non-linear terms in the partial differential equation.

Discretization

Discretizing equation 4.82 using a finite difference grid is straightforward and results in the following partial differential equation:

$$\frac{1}{h_x^2}D_{i-\frac{1}{2}}\phi_{i-1} + \frac{1}{h_x^2}D_{i+\frac{1}{2}}\phi_{i+1} + \frac{1}{h_y^2}D_{j-\frac{1}{2}}\phi_{j-1} \tag{4.85}$$

$$+\frac{1}{h_y^2}D_{j+\frac{1}{2}}\phi_{j+1} + \frac{1}{h_z^2}D_{k-\frac{1}{2}}\phi_{k-1} + \frac{1}{h_z^2}D_{k+\frac{1}{2}}\phi_{k+1} \tag{4.86}$$

$$-\left(\frac{1}{h_x^2}D_{i-\frac{1}{2}} + \frac{1}{h_x^2}D_{i+\frac{1}{2}} + \frac{1}{h_y^2}D_{j-\frac{1}{2}} + \frac{1}{h_y^2}D_{j+\frac{1}{2}} + \frac{1}{h_z^2}D_{k-\frac{1}{2}} + \frac{1}{h_z^2}D_{k+\frac{1}{2}}\right)\phi_{ijk} \tag{4.87}$$

$$= -\sigma_{a,ijk}\phi_{ijk} + q_{ijk} \tag{4.88}$$

The subscript to the diffusion coefficient D_F has been omitted for readability. Since it is defined at voxel centers, its off-center values are found by interpolation. For example,

$$D_{i+\frac{1}{2}} = \frac{1}{2}\big(D_i + D_{i+1}\big)\,. \tag{4.89}$$

The flux-limiter in equation 4.83 is simply discretized by using the discretized transport measure R_{ijk}:

$$\lambda(R_{ijk}) = \frac{1}{R_{ijk}}\left(\coth R_{ijk} - \frac{1}{R_{ijk}}\right) \tag{4.90}$$

Note, that the transport measure R will cause a division by zero if it becomes zero. This problem will be addressed by the way the transport measure R in equation 4.84

is discretized:

$$R_{ijk} = \frac{\max\left(\left\|\nabla\phi_{ijk}\right\|,\epsilon\right)}{\max\left(\sigma_{t,ijk}\phi_{ijk},\epsilon\right)} \quad \text{with} \quad \nabla\phi_{ijk} = \frac{1}{2}\begin{pmatrix} \frac{1}{h_x}\phi_{i+1}-\frac{1}{h_x}\phi_{i-1} \\ \frac{1}{h_y}\phi_{j+1}-\frac{1}{h_y}\phi_{j-1} \\ \frac{1}{h_z}\phi_{k+1}-\frac{1}{h_z}\phi_{k-1} \end{pmatrix} \tag{4.91}$$

The introduction of a minimum threshold ϵ in the nominator and denominator will prevent R from becoming zero or breaking down due to division by zero. The minimum threshold σ_{min} of the extinction coefficient will be applied on top.

Non-linear Gauss-Seidel Solver

The solver follows an iterative segregated approach for solving non-linear partial differential equations (Mazumder [60]). The idea is to update the diffusion coefficient from the current solution ϕ_{ijk} (initialized with a first guess) and then update the solution using the updated diffusion coefficient values D_{ijk} and repeat this update procedure until convergence to a final solution.

The iteration method used is the Gauss-Seidel fixpoint iteration scheme. The non-linearity is integrated by updating the diffusion coefficient in place during iteration over all voxels for a single Gauss-Seidel step.

The Gauss-Seidel update step is found by isolating ϕ_{ijk} in the discretized diffusion equation 4.88:

$$\phi_{ijk} = \frac{\frac{1}{h_x^2}D_{i-\frac{1}{2}}\phi_{i-1}+\frac{1}{h_x^2}D_{i+\frac{1}{2}}\phi_{i+1}+\frac{1}{h_y^2}D_{j-\frac{1}{2}}\phi_{j-1}+\frac{1}{h_y^2}D_{j+\frac{1}{2}}\phi_{j+1}+\frac{1}{h_z^2}D_{k-\frac{1}{2}}\phi_{k-1}+\frac{1}{h_z^2}D_{k+\frac{1}{2}}\phi_{k+1}+q_{ijk}}{-\left(\frac{1}{h_x^2}D_{i-\frac{1}{2}}+\frac{1}{h_x^2}D_{i+\frac{1}{2}}+\frac{1}{h_y^2}D_{j-\frac{1}{2}}+\frac{1}{h_y^2}D_{j+\frac{1}{2}}+\frac{1}{h_z^2}D_{k-\frac{1}{2}}+\frac{1}{h_z^2}D_{k+\frac{1}{2}}-\sigma_{a,ijk}\right)} \tag{4.92}$$

Then, the discrete transport measure R_{jk} is updated, using the newly retrieved value for ϕ_{ijk}. With this, the new diffusion coefficient is computed:

$$D_{ijk} = \frac{\lambda\left(R_{ijk}\right)}{\sigma_t'} \tag{4.93}$$

This updating step is executed for every voxel of the finite difference grid and values are updated in place.

The values ϕ_{ijk} and D_{ijk} are initialized to small tolerances $\phi_{ijk} = \epsilon\bar{j}\Delta l$ and $D_{ijk} = \epsilon\Delta l$ for all voxels. Initially, these grid values do not satisfy equation 4.92 and equation 4.93, but after several iterations, they converge to a consistent solution of both equations over the whole grid.

Boundary conditions are accounted for by updating the values for ϕ_{ijk} and D_{ijk} at the boundary voxels. For Dirichlet boundary conditions the values are set directly. For Neumann boundary conditions the boundary voxels are set according to the next inner voxel (section 3.5.4). This update is done at the start of each Gauss-Seidel iteration.

A criteria is set to stop the Gauss-Seidel iterations. For real-time applications a specific user-decided number of iterations might be a reasonable choice. Here, convergence to machine precision may be compromised to retain interactivity. Another option is to compute the root mean square of the residual and stop the Gauss-Seidel algorithm as soon as this falls below a user-defined threshold.

To improve the rate of convergence, successive over-relaxation (SOR) is used (Hadjidimos [36]). Defining the over-relaxation parameter $\boldsymbol{\omega}$, where $0 < \boldsymbol{\omega} < 2$, the update of ϕ_{ijk} at each stencil is modified to:

$$\phi_{ijk} \leftarrow \boldsymbol{\omega}\phi'_{ijk} + (1 - \boldsymbol{\omega})\phi_{ijk} \tag{4.94}$$

where ϕ'_{ijk} is the updated value from equation 4.92.

To further speed up the computation time a Red-Black Gauss-Seidel update process is used. This allows to update half of the voxels in parallel (Olshanskii et al. [70]). The voxels are separated into two disjunct groups according to a checkerboard pattern (hence the name Red-Black). Voxels within a group can be updated in parallel, because the stencil only requires information from the neighboring voxels in each dimension and those always belong to the other voxel group. Firstly, all voxels of one group are processed in parallel and their ϕ_{ijk} and D_{ijk} values are updated. Then all the voxels of the other group are updated in parallel. The two passes constitute one iteration. The full algorithm is outlined in listing 4.1.

4.10 Results

With the solver introduced in the previous section it can now be run for comparison on the problems, which were used for the P_N-method in chapter 3 and the diffusion approximation in chapter 4.4.

The rendering integration aspects to be considered are identical to the diffusion approximation (section 4.4). As with the diffusion approximation, flux-limited diffusion will give a solution for the zero moment ϕ from which the second moment can be recovered using equation 4.44. Both moments can be used to reconstruct the truncated spherical harmonics expansion of the radiance field by using equation 4.30. The dif-

Algorithm 4.1 Non-linear Gauss-Seidel SOR FLD solver

```
Initialize voxel grids $\phi$, $D$
repeat
    Update boundary voxels
    parallel for all red non-boundary voxels $ijk$ do
        Compute $R_{ijk}$ (equation 4.91)
        Compute $\lambda(R_{ijk})$ (equation 4.90)
        Compute and update $D_{ijk}$ (equation 4.93)
        Compute $D_{i\pm\frac{1}{2}}$, $D_{j\pm\frac{1}{2}}$ and $D_{k\pm\frac{1}{2}}$ (according to equation 4.89)
        Compute $\phi'_{ijk}$ (equation 4.92)
        Compute and update $\phi_{ijk}$ using equation 4.94
    end parallel for
    parallel for all black non-boundary voxels $ijk$ do
        Same as above but just for the black voxels
    end parallel for
    If required, compute root mean square of residual
until convergence criteria is met
```

ference to the diffusion approximation solution is that the fluence should be more accurate as the flux-limit constraint had been accounted for.

4.10.1 Point Source Problem

At first we consider the point source problem in order to validate the results from the flux-limited diffusion solver and assess its accuracy by comparing against an analytical ground truth solution. As seen in figure 4.9, the result from flux-limited diffusion agree well with the ground truth and diffusion approximation results. Like with the diffusion approximation, a perfect match to the ground truth is not to be expected as flux-limited diffusion is still an approximation based on the spherical harmonics expansion of the radiative transfer equation truncated after the first moment. However, very close to the point source, the flux-limited diffusion is more accurate than classical diffusion. This is explained by the fact, that the fluence gradient becomes arbitrarily large as the point light is approached, due to the geometry term. This causes the transport measure R to likewise become arbitrarily large, which shows that a pure streaming transport will dominate when coming close to the point source.

In comparison to the results from P_N-method and P_5 in particular, the P_N-method with higher truncation order is more accurate than flux-limited diffusion. This an important observation since it means that increasing the truncation order will alleviate the problems caused by not accounting for the flux-limit constraint. However, flux-limited

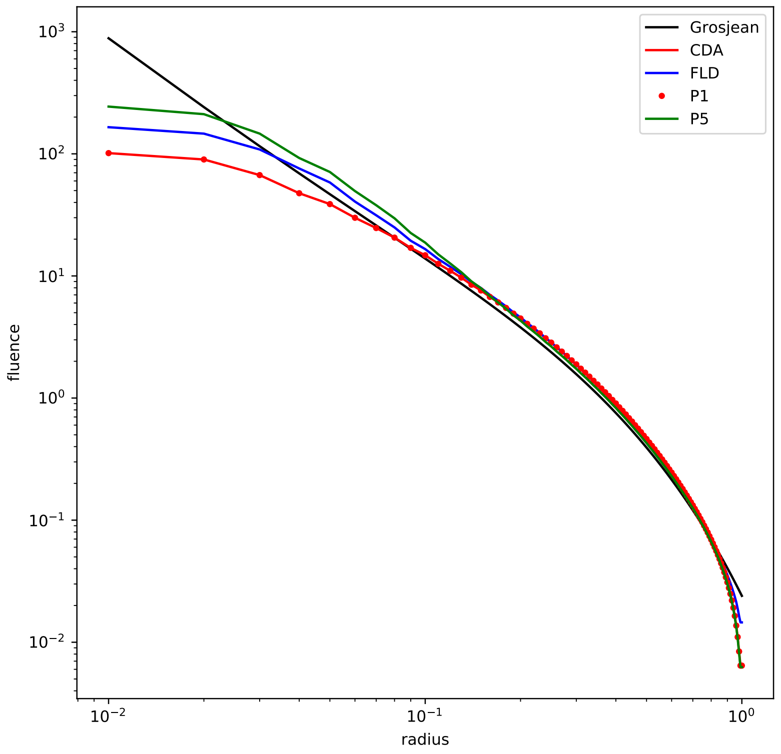


Figure 4.9: *Comparing the solution from flux-limited diffusion (blue) against classical diffusion approximation (red), P_5-solution (green) and groundtruth (black). Flux-limited diffusion much closer to the groundtruth and P_5-result at much lower computational cost.*

diffusion has a significcant advantage over P_N due to its computational and memory efficiency.

Table 4.2: *Performance characteristics of flux-limited diffusion and P_1 for the point source problem.*

Method	FLD	P_1
Number of rows/columns in A	262.144	1.048.576
Size of linear system (in MB)	1.3	8.4
Solve time	9s	10min

4.10.2 Procedural Cloud

Finally, the flux-limited diffusion is run on the procedural cloud dataset described in section 3.7.3 to assess its performance in a more practical setting. As seen in figure 4.10, flux-limited diffusion does a significantly better job at conserving energy than classical diffusion or P_5. This is explained by the fact, that the dataset contains very strong density gradients and close-to vacuum regions in which diffusive transport is not able to capture light transport well. While P_5 produces better results than classical diffusion, it is somehow surprising to see how much better flux-limited diffusion is able to capture the directly lit regions of the cloud.

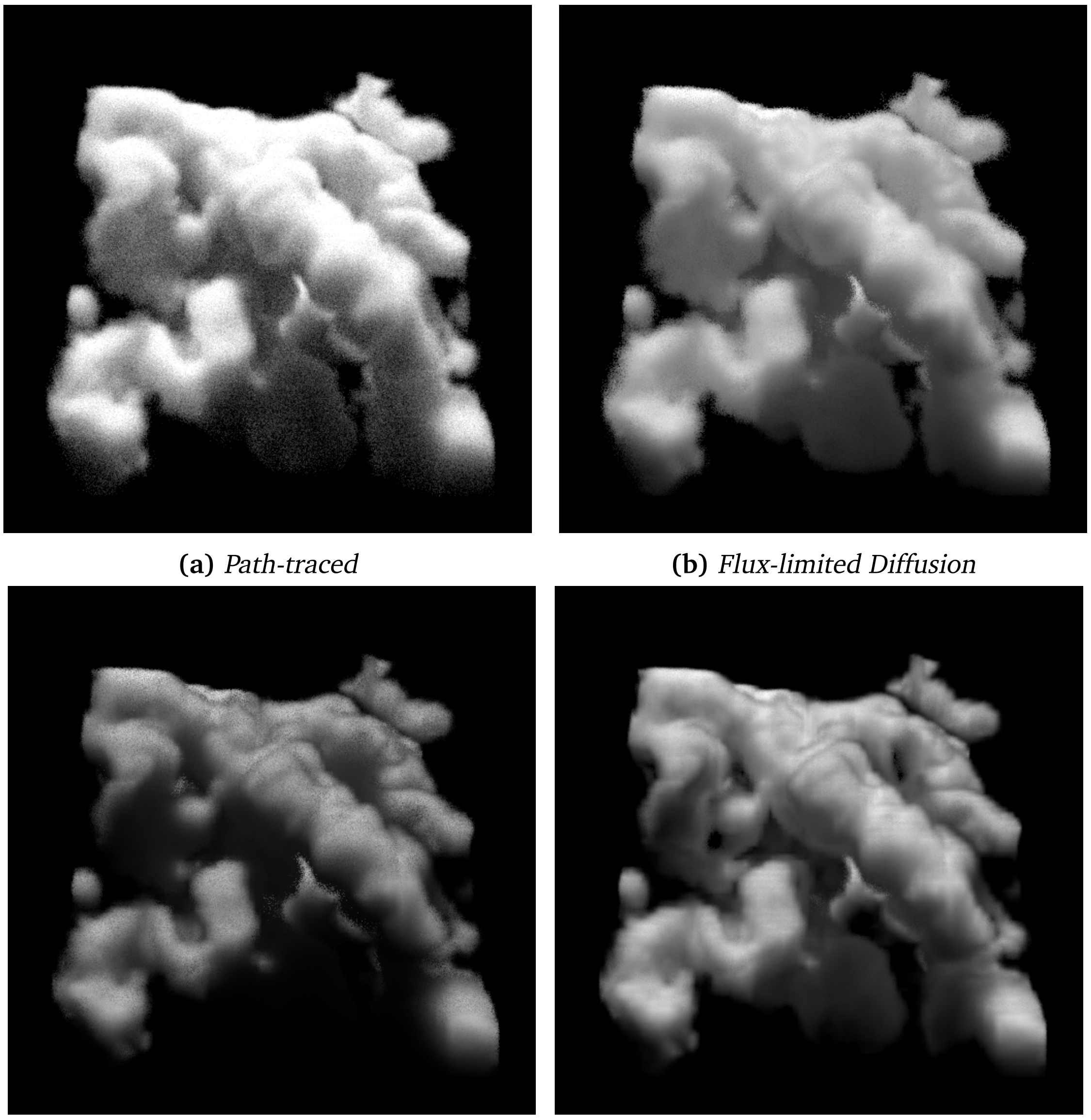
Further, flux-limited diffusion captures the indirectly illuminated parts at the bottom of the dataset well. Here, flux-limited diffusion also offers a signficiant improvement over classical diffusion approximation. Upon close inspection and comparison with P_5, it can be seen that the flux-limited diffusion solution appears very flat while the P_5 solution exhibits finer variations which closer resemble the path-traced resuls. This is explained by the fact that the angular domain is better resolved with P_5. The energy-conserving nature of flux-limited diffusion still produced a result, which more closely resembles the path-traced result than P_5.

In terms of performance checteristics it is to be expected that flux-limited diffusion requires more computational effort than the diffusion approximation, due to the non-linear nature of flux-limited diffusion coefficient, which must be updated for every voxel in each Gauss-Seidel iteration. This impact appears moderate when compared to the visual improvement it brings. When compared to P_N it can be concluded that flux-limited diffusion offers a much better result at a significantly lower computational cost.

4.11 Discussion

This chapter introduced flux-limited diffusion to the problem of rendering in computer graphics. It has its origins in the astrophysics domain (Levermore et al. [56]) and can be seen as an extension to the classical diffusion approximation, which was covered in section 4.4. In addition to revisiting the theory in the context of computer graphics, the major contribution of this chapter is the introduction of a new method for solving the non-linear flux-limited diffusion equation, which resulted in a publication [48].

The point source results shown in the previous section indicate that flux-limited diffusion produces significantly more accurate results than classical diffusion at a moderate increase in computational cost. Therefore, classical diffusion is only an option if a maximum in performance is required with a high tolerance towards significant approximation error. Otherwise, flux-limited diffusion is the better alternative. In addition, flux-limited diffusion outperforms the P_N-method for some higher levels than P_1 (on which flux-limited diffusion is based upon). It produces equally accurate results and appears to better preserve energy at a much smaller resource footprint.



(a) *Path-traced* **(b)** *Flux-limited Diffusion* **(c)** *Diffusion Approximation* **(d)** P_5

Figure 4.10: *Flux-limited diffusion results for the procedural cloud dataset compared against previous methods. FLD produces results which are similarily close to the pathtraced solution than P_5 at a fraction of its computational cost.*

The flux-limited diffusion theory was shown to be based on the idea of spatially blending advection and diffusion. This mixing depends on a local quantity. While this thesis is concerned with generating realistic images, an alternative route to explore may be non-photrealistic rendering. In particular the work on diffusion curves by Orzan [71] could be combined with flux-limited diffusion. The idea behind diffusion curves is to use diffusion to define the pixel colors in an image. User-defined curves – essentially emitters in two-dimensional space – determine the final image content. Recently Prevost et al. [77] extended this idea by using raytracing in the two-dimensional image space. By using flux-limited diffusion this technique may be extended with advection and may allow for more expressive imaging.

Chapter 5

Conclusion

Stochastic methods for light transport simulation are in the main focus of contemporary research in rendering. Boosting the convergence of these methods is the main thrust of today's efforts in research groups world wide. In particular, path-guiding based approaches are becoming increasingly popular. They have shown to improve stochastic methods significantly by using a global approximation of the radiance field to allow better informed decisions during stochastic sampling.

At the same time, a shift in compute hardware manufacturing from clock frequency growth to growth in core count with the next generation of multi-core and many-core hardware on the horizon is seen. High performance computing centers, which have large scale multi-core parallelization at the core of their business, are opening up and undergo a transformation towards becoming providers for public massive multi-core cloud computing. Developing algorithms that cater to the characteristics of this new hardware is deemed as one of the major challenges in future rendering research.

Both trends, path-guiding and multi-core hardware, are the main motivation for revisiting deterministic methods for light transport simulation, which have received little attention in the past years. In particular, there is a large body of research related to deterministic methods in fields such as astrophysics or nuclear sciences. Going through that research and adopting it for application in computer graphics has been fruitful in the past and is the main spirit behind the research done as part of this thesis. The chapters on the P_N-method (chapter 3) and flux-limited diffusion (chapter 4) constitute and reflect the majority of the results.

The first major contribution of this thesis to the field of computer graphics is the full introduction of P_N-theory to the field. While the theory has been used for a long time in astrophysics and nuclear sciences, it has never been applied to the problem of rendering in computer graphics. In that regard the contribution of this thesis is to serve as a bridge to better understand related research from other fields. The P_N-theory is dense and unwieldy to work with. This thesis revisits the theory and presents it to the audience in modern graphics research. A side product of this is a new and very concise form of the real-valued P_N-equations.

While the theory has been existing in other fields and was only established as new to the graphics domain, the method for solving the P_N-equations, which was developed as part of this thesis, is a novel contribution standing on its own and resulted in the publication by Koerner et al. [51]. The positive reviews and encouraging comments from the graphics community after the corresponding conference talk reconfirmed that this contribution closes an important gap in the graphics field.

The diffusion approximation is derived from P_N-theory by truncating the spherical harmonics expansion after the first moment. Therefore it is a deterministic method for light transport simulation based on the spherical harmonics expansion of the radiative transfer equation. It plays a very important role in many rendering techniques and was applied to many different problems in realtime and production rendering. This is why diffusion theory has been included in this thesis. Here, the contribution to graphics by this thesis lies primarily in illustrating the connection between the moment expansion and the more popular spherical harmonics expansion. This allows for a clear derivation of the classical well known diffusion equation.

The second major contribution in this thesis is the introduction of flux-limited diffusion in chapter 4 to the field of computer graphics. With the P_N-method, flux-limited diffusion has been invented in the astrophysics domain and also became popular in nuclear sciences and medical sciences. However, it has never been applied to the problem of rendering in computer graphics before. First, the theory is derived as a consequence of addressing deficiencies in diffusion theory. Therefore, the theory section builds upon the theoretical foundations laid out in previous chapters. In addition, to exposing the theory to the graphics community the contribution from this chapter is also a new method for solving the flux-limited diffusion equation, which allows for practical rendering applications. These contributions have been published by Koerner et al. [48]. In particular, the solver has been adopted in the industry (see figure 5.1).

With P_N-theory, diffusion and flux-limited diffusion, this thesis gives a comprehensive treatment of the domain of deterministic methods that are based on the spherical harmonics expansion of the radiative transfer equation. For every chapter, the theory is derived and a solver is being devised for solving the corresponding equations. This

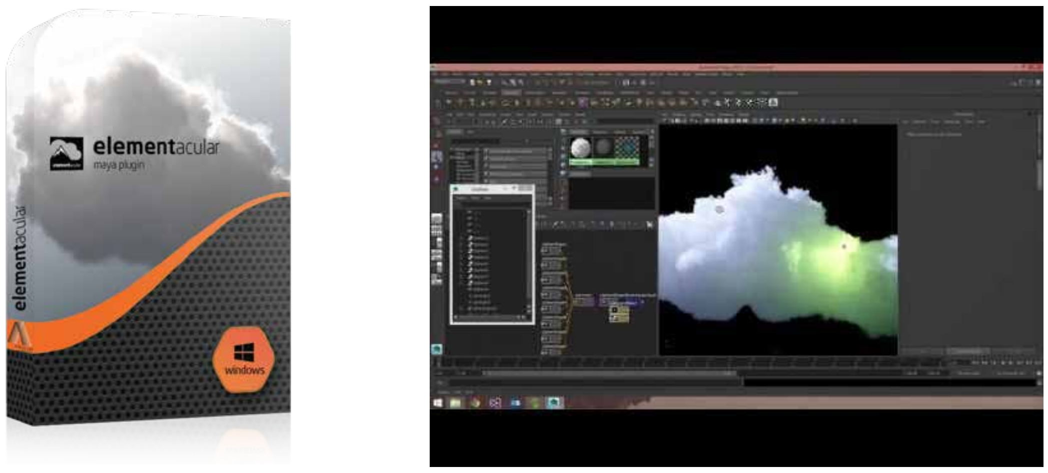


Figure 5.1: *Flux-limited diffusion is used to compute multiple scattering in Elementacular*[1]*, a commercial plugin for interactive cloud modelling. Due to the incremental nature of interactive edits, the result from previous frames is a great initial solution for the solver, which then only needs a few iterations to re-solve the underlying equation system.*

thesis becomes a contribution for understanding the methods and their application in practical applications.

Outlook

However, while being comprehensive, the coverage of spherical harmonics based methods for light transport simulation in this thesis is neither exhaustive nor complete. Although it has a long history, the field is still being actively researched and advanced, primarily by the nuclear science community.

The P_N-theory provides a solid framework for discretizing the radiative transfer equation in angular domain. However, for practical applications concerned with stead-state problems, the P_N-method suffers from having to solve very large linear systems if the truncation order is increased. This makes it impractical for real world applications. In addition, the theory and all its derivatives, such as the diffusion approximation and flux-limited diffusion, suffer from the presence of vacuum regions in the problem domain. More specifically, all three methods will not be able to converge for these problems. This is a major issue for application in graphics, where vacuum regions are ubiquitous.

[1]`http://elementacular.com` (date of access: 10/02/2019)

A direction for future research that has not been explored fully yet is to devise a multigrid method for solving the large coupled system of P_N-equations. Such a multigrid scheme could make the method more competitive in comparison to other approaches. An initial investigation showed that applying a naive multigrid approach to the P_N-problem does not give any performance improvements. We believe this is due to the coupling of the P_N-equation which makes error propagate between unknowns across cells and does not simply diffuse. This needs to be respected by the interpolation and restriction steps.

The introduction of the diffusion approximation has led to a variety of methods in computer graphics and a similar development can be envisioned for the P_N-method. The extension to finite-elements discretization or handling of more complex boundary conditions could allow the application of the P_N-method to problems such as subsurface scattering.

To deal with the vacuum problem, one direction for future research needs to be the evaluation of the various variations of the P_N-method, which have been proposed in other domains, such as nuclear sciences. Evaluating these variations was one idea behind the design of the solver in chapter 3, as all the variations are presented as a coupled system of partial differential equations which could be all fed as input to the solver developed as part of this chapter.

There are no obvious open questions regarding the diffusion approximation. This comes at no surprise since diffusion has been introduced to graphics a long time ago and received a lot of attention from graphics researchers throughout the years. Its traits are known and well studied.

Flux-limited diffusion offers a great balance between computational efficiency and accuracy, and therefore has seen some interest for industry application recently. Its linear counterpart, the non-linear Gauss-Seidel solver, does not scale well with the resolution of the finite difference grid. Here, it seems like a promising avenue to investigate non-linear multigrid solvers. Being able to apply flux-limited diffusion to much larger problem sizes could significantly broaden its applicability.

Another interesting direction could be to extend flux-limited diffusion or even the P_N-theory in general to anisotropic media. The literature on P_N-theory known to the author is based on isotropic media and deriving a P_N-theory for anisotropic media could open up directions for new deterministic methods in this field. While anisotropic media appears to play a minor role in graphics at the moment, some research, such as the work by Dupuy et al. [23], indicates that anisotropic media might be an important building block for bridging the divide between surface rendering (rendering equation) and volume rendering (radiative transfer).

The place of deterministic methods in rendering is uncertain due to their complexities, such as discretization bias, convergence issues and boundary conditions. Particularly, deterministic methods lack the appeal of conceptional simplicity that stochastic methods, such as Monte-Carlo, have. This not only prevents their adoption in the industry, but also makes them unattractive as a research subject as such. This thesis represents a significant effort to fill prominent knowledge gaps and demonstrate that deterministic methods and spherical harmonics based techniques in particular can be easily understood and provide very powerful tools in rendering.

Bibliography

[1] N. I. Akhiezer. *The classical moment problem and some related questions in analysis*. University mathematical monographs. Oliver & Boyd, 1965. doi: https://doi.org/10.1112/jlms/s1-41.1.757.

[2] A. Arbree, B. Walter, and K. Bala. *Heterogeneous Subsurface Scattering Using the Finite Element Method*. IEEE Transactions on Visualization and Computer Graphics, 17(7):956–969, 2011. doi: 10.1109/TVCG.2010.117.

[3] P. J. Armitage. *Dynamics of protoplanetary disks*. Annual Review of Astronomy and Astrophysics, 49:195–236, 2011. doi: https://doi.org/10.1146/annurev-astro-081710-102521.

[4] M. Billeter, E. Sintorn, and U. Assarsson. *Real-time Multiple Scattering Using Light Propagation Volumes*. In Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, pages 119–126, 2012. doi: https://doi.org/10.1145/2159616.2159636.

[5] T. E. Booth. *Monte Carlo Variance Comparison for Expected-Value versus Sampled Splitting*. Nuclear Science and Engineering, 89(4):305–309, 1985. doi: https://doi.org/10.13182/NSE85-A18622.

[6] R. L. Bowers and J. R. Wilson. *A numerical model for stellar core collapse calculations*. Astrophysical Journal Supplement Series, 50:115–159, 1982. doi: https://doi.org/10.1086/190822.

[7] B. J. Brinkworth. *A diffusion model of the transport of radiation from a point source in the lower atmosphere*. British Journal of Applied Physics, 15(6):733–741, jun 1964. doi: https://doi.org/10.1088/0508-3443/15/6/318. URL: https://doi.org/10.1088/0508-3443/15/6/318.

[8] T. A. Brunner. *Forms of approximate radiation transport*. Sandia National Laboratories Report, 2002. doi: https://doi.org/10.2172/800993.

[9] T. A. Brunner and J. P. Holloway. *Two-dimensional time dependent Riemann solvers for neutron transport*. Journal of Computational Physics, 210(1):386–399, 2005. doi: https://doi.org/10.1016/j.jcp.2005.04.011.

[10] T. Camminady, M. Frank, K. Küpper, and J. Kusch. *Ray effect mitigation for the discrete ordinates method through quadrature rotation*. J. Comput. Phys., 382: 105–123, 2019. doi: https://doi.org/10.1016/j.jcp.2019.01.016.

[11] B. G. Carlson and C. Lee. *Mechanical Quadrature and the Transport Equation*. Los Alamos Scientific Laboratories Reports, June 1961. URL: `https://www.osti.gov/biblio/4841446`.

[12] S. Chandrasekhar. *Radiative Transfer*. Dover Publications, 1960. doi: https://doi.org/10.1002/qj.49707633016.

[13] P. Clarberg and T. Akenine-Moeller. *Exploiting Visibility Correlation in Direct Illumination*. In Proceedings of the Nineteenth Eurographics Conference on Rendering, pages 1125–1136. Eurographics Association, 2008. doi: https://doi.org/10.1111/j.1467-8659.2008.01250.x.

[14] D. Cline. *A Practical Introduction to Metropolis Light Transport*. Technical report, 2005. URL: `https://api.semanticscholar.org/CorpusID:16452773`.

[15] D. Cline, J. Talbot, and P. Egbert. *Energy Redistribution Path Tracing*. ACM Transactions on Graphics (Proceedings of SIGGRAPH), 24(3):1186–1195, July 2005. doi: https://doi.org/10.1145/1073204.1073330.

[16] R. L. Cook, T. Porter, and L. Carpenter. *Distributed ray tracing*. Computer Graphics (Proceedings of SIGGRAPH), 18(3):137–145, Jan. 1984. doi: https://doi.org/10.1145/964965.808590.

[17] D. E. Cullen. *Why are the Pn and Sn Methods Equivalent?* Lawrence Livermore National Laboratory, 145518, 2001. URL: `http://home.comcast.net/~redcullen1/Papers/PnvsSn/pnvssn.pdf`.

[18] L. Da Vinci. *A Treatise on Painting*. 1651. URL: `http://www.gutenberg.org/ebooks/46915`.

[19] C. Dachsbacher and M. Stamminger. *Reflective Shadow Maps*. In Proceedings of Symposium on Interactive 3D Graphics and Games, pages 203–208, 2005. doi: https://doi.org/10.1145/1053427.1053460.

[20] F. Dai and Y. Xu. *Approximation theory and harmonic analysis on spheres and balls*. Springer, Apr. 2013. doi: https://doi.org/10.1007/978-1-4614-6660-4.

[21] E. D'Eon. *Computer graphics and particle transport: our common heritage, recent cross-field parallels and the future of our rendering equation*. In Proceedings of the Fourth Symposium on Digital Production, pages 37–39, 2014. doi: https://doi.org/10.1145/2633374.2633376.

[22] E. d'Eon and G. Irving. *A Quantized-Diffusion Model for Rendering Translucent Materials*. ACM Transactions on Graphics (Proceedings of SIGGRAPH), 30(4): 1–14, July 2011. doi: https://doi.org/10.1145/2010324.1964951.

[23] J. Dupuy, E. Heitz, and E. D'Eon. *Additional Progress Towards the Unification of Microfacet and Microflake Theories*. In Proceedings of Eurographics Symposium on Rendering - Experimental Ideas & Implementations, page 55–63. The Eurographics Association, 2016.

[24] S. R. Dwivedi. *A new importance biasing scheme for deep-penetration Monte Carlo*. Annals of Nuclear Energy, 9(7):359 – 368, 1982. ISSN 0306-4549. doi: https://doi.org/10.1016/0306-4549(82)90038-X. URL: http://www.sciencedirect.com/science/article/pii/030645498290038X.

[25] C. Eisenacher, G. Nichols, A. Selle, and B. Burley. *Sorted deferred shading for production path tracing*. In Computer Graphics Forum, volume 32, pages 125–132. Wiley Online Library, 2013. doi: https://doi.org/10.1111/cgf.12158.

[26] L. Fascione. *Rendering Research at Weta Digital*. In Keynote, Digital Production Symposium, Los Angeles, California, Aug. 2015. ISBN 9781450337182.

[27] G. B. Folland. *Fourier analysis and its applications*, volume 4. American Mathematical Society, 1992. doi: https://doi.org/10.1007/b97452.

[28] M. Frank, H. Hensel, and A. Klar. *Toward fast and accurate methods for dose calculation in radiotherapy*. Proceedings in Applied Mathematics and Mechanics, 7 (1):2120009–2120010, 2008. doi: https://doi.org/10.1002/pamm.200700407.

[29] M. Frank, C. Hauck, and K. Küpper. *Convergence of filtered spherical harmonic equations for radiation transport*. Communications in Mathematical Sciences, 14:1443–1465, Jan. 2016.

[30] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-oriented Software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995. ISBN 9780201633610.

[31] R. Geist, K. Rasche, J. Westall, and R. Schalkoff. *Lattice-Boltzmann Lighting*. In Computer Graphics Forum (Proceedings of Eurographics Symposium on Rendering), pages 355–362. Eurographics Association, 2004. doi: https://doi.org/10.1016/B978-0-12-384988-5.00025-5.

[32] P. Grinfeld. *Introduction to Tensor Analysis and the Calculus of Moving Surfaces*. Springer New York, 2013. ISBN 978-1-4614-7867-6.

[33] C. C. Grosjean. *A high accuracy approximation for solving multiple scattering problems in infinite homogeneous media*. Il Nuovo Cimento (1955-1965), 3(6): 1262–1275, June 1956. doi: 10.1007/BF02785007.

[34] G. Guennebaud, B. Jacob, et al. *Eigen v3*, 2010. URL: `https://eigen.tuxfamily.org/`.

[35] R. Habel, P. H. Christensen, and W. Jarosz. *Photon Beam Diffusion: A Hybrid Monte Carlo Method for Subsurface Scattering*. Computer Graphics Forum (Proceedings of Eurographics Symposium on Rendering), 32(4), June 2013. doi: https://doi.org/10.1111/cgf.12148.

[36] A. Hadjidimos. *Successive overrelaxation (SOR) and related methods*. Journal of Computational and Applied Mathematics, 123(1):177–199, 2000. doi: https://doi.org/10.1016/S0377-0427(00)00403-9.

[37] A. Haghighat and J. Wagner. *Monte Carlo variance reduction with deterministic importance functions*. Progress in Nuclear Energy, 42:25–53, Dec. 2003. doi: https://doi.org/10.1016/S0149-1970(02)00002-1.

[38] B. C. Hall. *Quantum Theory for Mathematicians*. Springer-Verlag New York, New York, first edition, 2013. ISBN 9781461471165.

[39] J. Hansen, J. Peterson, J. Morel, J. Ragusa, and Y. Wang. *A Least-Squares Transport Equation Compatible with Voids*. Journal of Computational and Theoretical Transport, 43(1-7):374–401, 2014. doi: https://doi.org/10.1080/00411450.2014.927364.

[40] A. Ishimaru. *Wave Propagation and Scattering in Random Media. Single Scattering and Transport Theory*, volume 1. Academic Press, 1978. doi: https://doi.org/10.1016/B978-0-12-374701-3.X5001-7.

[41] W. Jakob, J. T. Moon, A. Arbree, K. Bala, and S. Marschner. *A Radiative Transfer Framework for Rendering Materials with Anisotropic Structure*. ACM Transactions on Graphics (Proceedings of SIGGRAPH), 29(10):1–13, July 2010. doi: https://doi.org/10.1145/1778765.1778790.

[42] H. W. Jensen, S. R. Marschner, M. Levoy, and P. Hanrahan. *A Practical Model for Subsurface Light Transport*. In Proceedings of SIGGRAPH 01, pages 511–518. ACM, 2001. doi: https://doi.org/10.1145/383259.383319.

[43] J. T. Kajiya and B. P. Von Herzen. *Ray Tracing Volume Densities*. Computer Graphics (Proceedings of SIGGRAPH), 18(3):165–174, Jan. 1984. doi: https://doi.org/10.1145/964965.808594.

[44] M. H. Kalos and P. A. Whitlock. *Monte Carlo methods. Vol. 1: basics*. Wiley-Interscience, New York, NY, USA, 1986. ISBN 978-0-471-89839-9.

[45] A. S. Kaplanyan and C. Dachsbacher. *Cascaded Light Propagation Volumes for Real-time Indirect Illumination*. Proceedings of Symposium on Interactive 3D Graphics and Games, pages 99–107, 2010. doi: https://doi.org/10.1145/1730804.1730821.

[46] A. Keller, L. Fascione, M. Fajardo, I. Georgiev, P. Christensen, J. Hanika, C. Eisenacher, and G. Nichols. *The Path Tracing Revolution in the Movie Industry*. In ACM SIGGRAPH Courses, pages 1–7, New York, NY, USA, 2015. ACM. doi: https://doi.org/10.1145/2776880.2792699.

[47] D. S. Kershaw. *Flux Limiting Nature's Own Way - A New Method for Numerical Solution of the Transport Equation*. Lawrence Livermore National Laboratory, July 1976. doi: https://doi.org/10.2172/104974.

[48] D. Koerner, J. Portsmouth, F. Sadlo, T. Ertl, and B. Eberhardt. *Flux-Limited Diffusion for Multiple Scattering in Participating Media*. Computer Graphics Forum, 33:178, 2014. doi: https://doi.org/10.1111/cgf.12342.

[49] D. Koerner, J. Novak, P. Kutz, R. Habel, and W. Jarosz. *Subdivision Next-Event Estimation for Path-Traced Subsurface Scattering*. In Eurographics Symposium on Rendering - Experimental Ideas & Implementations. The Eurographics Association, 2016. doi: https://doi.org/10.2312/sre.20161214.

[50] D. Koerner, A. Harjunmaa, and A. Baumann. *High-Performance Computing for Artistic Content Creation*. FMX—International Conference on Effects, VR, Games and Transmedia, Stuttgart, Germany, May 2017. URL: `https://fmx.de/program2017/event/10211`.

[51] D. Koerner, J. Portsmouth, and W. Jakob. *PN-Method for Multiple Scattering in Participating Media*. In Eurographics Symposium on Rendering - Experimental Ideas & Implementations. The Eurographics Association, 2018. doi: https://doi.org/10.2312/sre.20181170.

[52] J. Křivánek and E. d'Eon. *A Zero-variance-based Sampling Scheme for Monte Carlo Subsurface Scattering*. In ACM SIGGRAPH Talks, 2014. doi: https://doi.org/10.1145/2614106.2614138.

[53] E. P. Lafortune and Y. D. Willems. *Bi-Directional Path Tracing*. In Computer Graphics (Proceedings of SIGGRAPH), pages 145–153, 1993.

[54] E. W. Larsen and J. B. Keller. *Asymptotic solution of neutron transport problems for small mean free paths*. Journal of Mathematical Physics, 15:75–81, 1974. doi: https://doi.org/10.1063/1.1666510.

[55] C. D. Levermore. *Relating Eddington factors to flux limiters*. Journal of Quantitative Spectroscopy and Radiative Transfer, 31(2):149–160, 1984. doi: https://doi.org/10.1016/0022-4073(84)90112-2.

[56] C. D. Levermore and G. C. Pomraning. *A flux-limited diffusion theory*. Astrophysical Journal, 248:321–334, 1981. doi: https://doi.org/10.1086/159157.

[57] D. Li, X. Sun, Z. Ren, S. Lin, Y. Tong, B. Guo, and K. Zhou. *TransCut: Interactive Rendering of Translucent Cutouts*. IEEE Transactions on Visualization and Computer Graphics, 19(3):484–494, 2013. doi: https://doi.org/10.1109/TVCG.2012.127.

[58] T.-M. Li, J. Lehtinen, R. Ramamoorthi, W. Jakob, and F. Durand. *Anisotropic Gaussian Mutations for Metropolis Light Transport Through Hessian-Hamiltonian Dynamics*. ACM Transactions on Graphics, 34(6):1–13, Oct. 2015. doi: https://doi.org/10.1145/2816795.2818084.

[59] N. Max. *Efficient Light Propagation for Multiple Anisotropic Volume Scattering*, pages 87–104. Springer, Berlin, Heidelberg, 1995. ISBN 978-3-642-87825-1.

[60] S. Mazumder. *Numerical Methods for Partial Differential Equations: Finite Difference and Finite Volume Methods*. Elsevier Science & Technology Books, 2015. ISBN 9780128498941.

[61] R. G. McClarren. *Theoretical Aspects of the Simplified Pn Equations*. Transport Theory and Statistical Physics, 39(2-4):73–109, 2010. doi: https://doi.org/10.1080/00411450.2010.535088.

[62] S. Mehta, R. Ramamoorthi, M. Meyer, and C. Hery. *Analytic Tangent Irradiance Environment Maps for Anisotropic Surfaces*. Computer Graphics Forum, 31(4):1501–1508, June 2012. doi: https://doi.org/10.1111/j.1467-8659.2012.03146.x.

[63] N. Metropolis. *The Beginning of the Monte Carlo Method*. Los Alamos Scientific Laboratories, Sept. 1977. URL: https://api.semanticscholar.org/CorpusID:18607470.

[64] N. Metropolis and S. M. Ulam. *The Monte Carlo Method*. Journal of the American Statistical Association, 44(247):335–341, Sept. 1949. doi: https://doi.org/10.2307/2280232.

[65] A. Meurer, C. P. Smith, M. Paprocki, O. Čertík, S. B. Kirpichev, M. Rocklin, A. Kumar, S. Ivanov, J. K. Moore, S. Singh, T. Rathnayake, S. Vig, B. E. Granger, R. P. Muller, F. Bonazzi, H. Gupta, S. Vats, F. Johansson, F. Pedregosa, M. J. Curry, A. R. Terrel, v. Roučka, A. Saboo, I. Fernando, S. Kulal, R. Cimrman, and A. Scopatz. *SymPy: Symbolic computing in Python*. PeerJ Computer Science, 3:103, Jan. 2017. URL: https://www.sympy.org/en/index.html.

[66] B. Miller, K. Museth, D. Penney, and N. B. Zafar. *Cloud modeling and rendering for ”Puss in Boots”*. ACM SIGGRAPH Talks, 2012. URL: `https://api.semanticscholar.org/CorpusID:63405879`.

[67] G. N. Minerbo. *Maximum entropy Eddington factors*. Journal of Quantitative Spectroscopy and Radiative Transfer, 20:541–545, 1978. doi: https://doi.org/10.1016/0022-4073(78)90024-9.

[68] J. T. Moon, B. Walter, and S. R. Marschner. *Rendering Discrete Random Media Using Precomputed Scattering Solutions*. In Proceedings of the 18th Eurographics Conference on Rendering Techniques, pages 231–242. Eurographics Association, 2007. ISBN 9783905673524. doi: https://doi.org/10.2312/EGWR/EGSR07/231-242.

[69] T. Müller, M. Gross, and J. Novák. *Practical Path Guiding for Efficient Light-Transport Simulation*. Computer Graphics Forum, 36(4):91–100, June 2017. doi: https://doi.org/10.1111/cgf.13227.

[70] M. Olshanskii and E. Tyrtyshnikov. *Iterative Methods for Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2014. doi: https://doi.org/10.1137/1.9781611973464.

[71] A. Orzan, A. Bousseau, H. Winnemöller, P. Barla, J. Thollot, and D. Salesin. *Diffusion Curves: A Vector Representation for Smooth-Shaded Images*. In ACM Transactions on Graphics (Proceedings of SIGGRAPH), volume 27, 2008. doi: https://doi.org/10.1145/2483852.2483873.

[72] M. Pauly, T. Kollig, and A. Keller. *Metropolis Light Transport for Participating Media*. In Proceedings of Eurographics Workshop on Rendering Techniques, pages 11–22, London, UK, 2000. Springer-Verlag. doi: https://doi.org/10.1007/978-3-7091-6303-0_2.

[73] V. Pegoraro, I. Wald, and S. G. Parker. *Sequential Monte Carlo Adaptation in Low-Anisotropy Participating Media*. Computer Graphics Forum, 27(4):1097–1104, 2008. doi: https://doi.org/10.1111/j.1467-8659.2008.01247.x.

[74] V. Pegoraro, M. Schott, and P. Slusallek. *A Mathematical Framework for Efficient Closed-Form Single Scattering*. In Proceedings of the 37th Graphics Interface Conference, pages 151–158, 2011. ISBN 9781450306935.

[75] B. Preiss. *Data Structures and Algorithms with Object-Oriented Design Patterns in Java*. Wiley, Jan. 2000. ISBN 978-0-471-34613-5.

[76] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, third edition, 2007. ISBN 978-0521880688.

[77] R. Prévost, W. Jarosz, and O. Sorkine-Hornung. *A Vectorial Framework for Ray Traced Diffusion Curves*. Computer Graphics Forum, 34(1):253–264, Feb. 2015. doi: https://doi.org/10/f6497s.

[78] D. Radice, E. Abdikamalov, L. Rezzolla, and C. D. Ott. *A new spherical harmonics scheme for multi-dimensional radiation transport I: Static matter configurations*. Journal of Computational Physics, 242:648–669, 2013. doi: https://doi.org/10.1016/j.jcp.2013.01.048.

[79] K. F. Riley, M. P. Hobson, and S. J. Bence. *Mathematical Methods for Physics and Engineering: A Comprehensive Guide*. Cambridge University Press, 2006. ISBN 978-0521679718.

[80] M. Schäfer, M. Frank, and C. D. Levermore. *Diffusive Corrections to PN Approximations*. Multiscale Modeling and Simulation, 9:1–28, 2011. doi: https://doi.org/10.1137/090764542.

[81] B. Seibold and M. Frank. *StaRMAP—A Second Order Staggered Grid Method for Spherical Harmonics Moment Equations of Radiative Transfer*. ACM Transactions on Mathematical Software, 41(1):1–28, Oct. 2014. doi: https://doi.org/10.1145/2590808.

[82] J. Stam. *Multiple Scattering as a Diffusion Process*. In Proceedings of Eurographics Workshop on Rendering Techniques, pages 41–50. Springer-Verlag, 1995. doi: https://doi.org/10.1007/978-3-7091-9430-0_5.

[83] J. Tessendorf and M. Kowalski. *Resolution Independent Volumes*. In ACM SIGGRAPH Courses, SIGGRAPH, Los Angeles, California, 2011. doi: https://doi.org/10.13140/RG.2.2.36215.06562.

[84] J. N. Tsitsiklis. *Efficient algorithms for globally optimal trajectories*. IEEE Transactions on Automatic Control, 40(9):1528–1538, 1995. doi: https://doi.org/10.1109/9.412624.

[85] E. Veach. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, Stanford, CA, USA, Dec. 1997.

[86] E. Veach and L. Guibas. *Optimally Combining Sampling Techniques for Monte Carlo Rendering*. In Proceedings of SIGGRAPH, page 419–428, 1995. doi: https://doi.org/10.1145/218380.218498.

[87] E. Veach and L. J. Guibas. *Bidirectional Estimators for Light Transport*. In Proceedings of Eurographics Rendering Workshop, pages 147–162, 1994. doi: https://doi.org/10.1007/978-3-642-87825-1_11.

[88] J. Vorba and J. Křivánek. *Adjoint-Driven Russian Roulette and Splitting in Light Transport Simulation*. ACM Transactions on Graphics, 35(4):1–11, July 2016. doi: https://doi.org/10.1145/2897824.2925912.

[89] J. Vorba, O. Karlík, M. Šik, T. Ritschel, and J. Křivánek. *On-line Learning of Parametric Mixture Models for Light Transport Simulation*. ACM Transactions on Graphics (Proceedings of SIGGRAPH), 33(4):1–11, July 2014. doi: https://doi.org/10.1145/2601097.2601203.

[90] J. Wang, S. Zhao, X. Tong, S. Lin, Z. Lin, Y. Dong, B. Guo, and H.-Y. Shum. *Modeling and rendering of heterogeneous translucent materials using the diffusion equation*. ACM Transactions on Graphics, 27(1):9, 2008. doi: https://doi.org/10.1145/1330511.1330520.

[91] P. P. Whalen. *Variable Eddington factors and flux-limiting diffusion coefficients*, volume 14. Los Alamos National Laboratories, Jan. 1982.

[92] G. C. Wick. *Über ebene Diffusionsprobleme*. Zeitschrift für Physik, 121(11-12):702–718, 1943. doi: https://doi.org/10.1007/BF01339167.

[93] Y. Zhang and K.-L. Ma. *Fast Global Illumination for Interactive Volume Visualization*. In Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, pages 55–62, New York, NY, USA, 2013. ACM.

Appendix **A**

Appendix: Derivation of Real-valued P_N-equation Terms

This chapter gives the derivation of the real-valued P_N-equation. All manipulation steps, substitutions and applications of identities were carried out using the computer algebra representation of the equations presented in chapter 3.5.1. The equations were rendered to LaTeX after each step.

The real-valued spherical harmonics basis functions are different for $m < 0$, $m = 0$ and $m > 0$ which results in three different projections.

We start by projecting the radiance field L:

$$
\begin{aligned}
\hat{L}(\mathbf{x},\boldsymbol{\omega}) &= \begin{cases} \sum_{l,m} L^{l,m}(\mathbf{x}) \frac{i}{\sqrt{2}}\left(Y_{\mathbb{C}}^{l,m} - (-1)^m Y_{\mathbb{C}}^{l,-m}\right), & \text{for } m<0 \\ \sum_{l,m} L^{l,m}(\mathbf{x}) Y_{\mathbb{C}}^{l,m}, & \text{for } m=0 \\ \sum_{l,m} L^{l,m}(\mathbf{x}) \frac{1}{\sqrt{2}}\left(Y_{\mathbb{C}}^{l,-m} - (-1)^m Y_{\mathbb{C}}^{l,m}\right), & \text{for } m>0 \end{cases} \\
&= \begin{cases} \sum_{l} \sum_{m=-l}^{-1} L^{l,m}(\mathbf{x}) \frac{i}{\sqrt{2}}\left(Y_{\mathbb{C}}^{l,m} - (-1)^m Y_{\mathbb{C}}^{l,-m}\right), & \text{for } m<0 \\ L^{l,0}(\mathbf{x}) Y_{\mathbb{C}}^{l,0}, & \text{for } m=0 \\ \sum_{l} \sum_{m=1}^{l} L^{l,m}(\mathbf{x}) \frac{1}{\sqrt{2}}\left(Y_{\mathbb{C}}^{l,-m} - (-1)^m Y_{\mathbb{C}}^{l,m}\right), & \text{for } m>0 \end{cases} \\
&= \sum_{l=0}^{N} \left(\sum_{m=-l}^{-1} L^{l,m}(\mathbf{x}) \left(\frac{i}{\sqrt{2}} Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) - \frac{i}{\sqrt{2}} (-1)^m Y_{\mathbb{C}}^{l,-m}(\boldsymbol{\omega}) \right) \right. \\
&\quad + L^{l,0}(\mathbf{x}) Y_{\mathbb{C}}^{l,0}(\boldsymbol{\omega}) \\
&\quad \left. + \sum_{m=1}^{l} L^{l,m}(\mathbf{x}) \left(\frac{1}{\sqrt{2}} Y_{\mathbb{C}}^{l,-m}(\boldsymbol{\omega}) + \frac{1}{\sqrt{2}} (-1)^m Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) \right) \right) \\
&= \frac{i}{\sqrt{2}} \left(\sum_{l=0}^{N} \sum_{m=-l}^{-1} L^{l,m}(\mathbf{x}) Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) \right) - \frac{i}{\sqrt{2}} \left(\sum_{l=0}^{N} \sum_{m=-l}^{-1} L^{l,m}(\mathbf{x}) (-1)^m Y_{\mathbb{C}}^{l,-m}(\boldsymbol{\omega}) \right) \\
&\quad + \sum_{l=0}^{N} L^{l,0}(\mathbf{x}) Y_{\mathbb{C}}^{l,0}(\boldsymbol{\omega}) \\
&\quad + \frac{1}{\sqrt{2}} \left(\sum_{l=0}^{N} \sum_{m=1}^{l} L^{l,m}(\mathbf{x}) Y_{\mathbb{C}}^{l,-m}(\boldsymbol{\omega}) \right) + \frac{1}{\sqrt{2}} \left(\sum_{l=0}^{N} \sum_{m=1}^{l} L^{l,m}(\mathbf{x}) (-1)^m Y_{\mathbb{C}}^{l,m}(\boldsymbol{\omega}) \right)
\end{aligned}
\tag{A.1}
$$

The transport term of the RTE is given as

$$
(\boldsymbol{\omega}\cdot\nabla)L(\mathbf{x},\boldsymbol{\omega}) = \boldsymbol{\omega}_x \partial_x L(\mathbf{x},\boldsymbol{\omega}) + \boldsymbol{\omega}_y \partial_y L(\mathbf{x},\boldsymbol{\omega}) + \boldsymbol{\omega}_z \partial_z L(\mathbf{x},\boldsymbol{\omega}) \tag{A.2}
$$

To improve readability, we first project the term into SH by multiplying with the conjugate complex of the SH basis, and replace L by its SH expansion afterwards. This order was reversed, when we derived the complex-valued P_N-equation in section 3.2.

We now multiply equation A.2 with the real-valued SH basis and integrate over solid angle. However, the SH basis is different for $m' < 0$, $m' = 0$ and $m' > 0$, and therefore will give us different P_N-equations depending on m'. We will go through the derivation in detail for the $m' < 0$ case and give the results for the other cases at the end.

Multiplying the expanded transport term with the SH basis for $m' < 0$ and integrating over solid angle gives:

$$
\begin{aligned}
&\int \left(\frac{-i}{\sqrt{2}} \overline{Y^{l',m'}}(\boldsymbol{\omega}) - \frac{-i}{\sqrt{2}} (-1)^{m'} \overline{Y^{l',-m'}}(\boldsymbol{\omega}) \right) \big(\boldsymbol{\omega}_x \partial_x L(\mathbf{x},\boldsymbol{\omega}) + \boldsymbol{\omega}_y \partial_y L(\mathbf{x},\boldsymbol{\omega}) + \boldsymbol{\omega}_z \partial_z L(\mathbf{x},\boldsymbol{\omega}) \big) \mathrm{d}\boldsymbol{\omega} \\
=& \int -\frac{i}{\sqrt{2}} \overline{Y^{l',m'}}(\boldsymbol{\omega}) \boldsymbol{\omega}_x \partial_x L(\mathbf{x},\boldsymbol{\omega}) - \frac{i}{\sqrt{2}} \overline{Y^{l',m'}}(\boldsymbol{\omega}) \boldsymbol{\omega}_y \partial_y L(\mathbf{x},\boldsymbol{\omega}) - \frac{i}{\sqrt{2}} \overline{Y^{l',m'}}(\boldsymbol{\omega}) \boldsymbol{\omega}_z \partial_z L(\mathbf{x},\boldsymbol{\omega}) \\
&+ \frac{i}{\sqrt{2}} (-1)^{m'} \overline{Y^{l',-m'}}(\boldsymbol{\omega}) \boldsymbol{\omega}_x \partial_x L(\mathbf{x},\boldsymbol{\omega}) + \frac{i}{\sqrt{2}} (-1)^{m'} \overline{Y^{l',-m'}}(\boldsymbol{\omega}) \boldsymbol{\omega}_y \partial_y L(\mathbf{x},\boldsymbol{\omega}) \\
&+ \frac{i}{\sqrt{2}} (-1)^{m'} \overline{Y^{l',-m'}}(\boldsymbol{\omega}) \boldsymbol{\omega}_z \partial_z L(\mathbf{x},\boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega}
\end{aligned}
$$

After expanding the integrand and splitting the integral, we apply the recursive relation from equation 3.12 to get:

$$
\begin{aligned}
&\frac{i}{\sqrt{2}}\frac{1}{2}c^{l'-1,m'-1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} - \frac{i}{\sqrt{2}}\frac{1}{2}d^{l'+1,m'-1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&-\frac{i}{\sqrt{2}}\frac{1}{2}e^{l'-1,m'+1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} + \frac{i}{\sqrt{2}}\frac{1}{2}f^{l'+1,m'+1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&-\frac{i}{\sqrt{2}}\frac{i}{2}c^{l'-1,m'-1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} + \frac{i}{\sqrt{2}}\frac{i}{2}d^{l'+1,m'-1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&-\frac{i}{\sqrt{2}}\frac{i}{2}e^{l'-1,m'+1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} + \frac{i}{\sqrt{2}}\frac{i}{2}f^{l'+1,m'+1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&-\frac{i}{\sqrt{2}}a^{l'-1,m'}\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} - \frac{i}{\sqrt{2}}b^{l'+1,m'}\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&-\frac{i}{\sqrt{2}}(-1)^{m'}\frac{1}{2}c^{l'-1,-m'-1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&+\frac{i}{\sqrt{2}}(-1)^{m'}\frac{1}{2}d^{l'+1,-m'-1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&+\frac{i}{\sqrt{2}}(-1)^{m'}\frac{1}{2}e^{l'-1,-m'+1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&-\frac{i}{\sqrt{2}}(-1)^{m'}\frac{1}{2}f^{l'+1,-m'+1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&+\frac{i}{\sqrt{2}}(-1)^{m'}\frac{i}{2}c^{l'-1,-m'-1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&-\frac{i}{\sqrt{2}}(-1)^{m'}\frac{i}{2}d^{l'+1,-m'-1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&+\frac{i}{\sqrt{2}}(-1)^{m'}\frac{i}{2}e^{l'-1,-m'+1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&-\frac{i}{\sqrt{2}}(-1)^{m'}\frac{i}{2}f^{l'+1,-m'+1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&+\frac{i}{\sqrt{2}}(-1)^{m'}a^{l'-1,-m'}\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,-m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&+\frac{i}{\sqrt{2}}(-1)^{m'}b^{l'+1,-m'}\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,-m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}
\end{aligned}
$$

Before we further expand the radiance field L into its SH expansion, we will simplify coefficients by using the following relations:

$$a^{l,m} = a^{l,-m}, \qquad b^{l,m} = b^{l,-m}, \qquad c^{l,m} = e^{l,-m}, \qquad d^{l,m} = f^{l,-m} \tag{A.3}$$

This allows us to rewrite the equation above as:

$$\begin{aligned}
&-i\alpha_c \int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} + (-1)^{m'} i\alpha_c \int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \\
&+i\alpha_d \int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} - (-1)^{m'} i\alpha_d \int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \\
&-i\alpha_e \int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} + (-1)^{m'} i\alpha_e \int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \\
&+i\alpha_f \int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} - (-1)^{m'} i\alpha_f \int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \\
&+\alpha_c \int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} + (-1)^{m'} \alpha_c \int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \\
&-\alpha_e \int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} - (-1)^{m'} \alpha_e \int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \\
&+\alpha_f \int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} + (-1)^{m'} \alpha_f \int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \\
&-\alpha_d \int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} - (-1)^{m'} \alpha_d \int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \\
&-\alpha_a \int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} + (-1)^{m'} \alpha_a \int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'-1,-m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \\
&-\alpha_b \int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} + (-1)^{m'} \alpha_b \int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y^{l'+1,-m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}
\end{aligned}$$

with

$$\alpha_c = \frac{i}{\sqrt{2}}\frac{1}{2}c^{l'-1,m'-1}, \qquad \alpha_e = \frac{i}{\sqrt{2}}\frac{1}{2}e^{l'-1,m'+1}, \qquad \alpha_d = \frac{i}{\sqrt{2}}\frac{1}{2}d^{l'+1,m'-1}$$

$$\alpha_f = \frac{i}{\sqrt{2}}\frac{1}{2}f^{l'+1,m'+1}, \qquad \alpha_a = \frac{i}{\sqrt{2}}a^{l'-1,m'}, \qquad \alpha_b = \frac{i}{\sqrt{2}}b^{l'+1,m'}$$

In the next step, we substitute the radiance field function L with its spherical harmonics expansion and arrive at the following expression after further expansions and transformations:

$$-i\alpha_c\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}+(-1)^{m'}i\alpha_c\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{A.4}$$

$$+i\alpha_d\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}-(-1)^{m'}i\alpha_d\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{A.5}$$

$$-i\alpha_e\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}+(-1)^{m'}i\alpha_e\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{A.6}$$

$$+i\alpha_f\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}-(-1)^{m'}i\alpha_f\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{A.7}$$

$$+\alpha_c\partial_x\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}+(-1)^{m'}\alpha_c\partial_x\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{A.8}$$

$$-\alpha_e\partial_x\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}-(-1)^{m'}\alpha_e\partial_x\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{A.9}$$

$$+\alpha_f\partial_x\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}+(-1)^{m'}\alpha_f\partial_x\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{A.10}$$

$$-\alpha_d\partial_x\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}-(-1)^{m'}\alpha_d\partial_x\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{A.11}$$

$$-\alpha_a\partial_z\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}+(-1)^{m'}\alpha_a\partial_z\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,-m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{A.12}$$

$$-\alpha_b\partial_z\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}+(-1)^{m'}\alpha_b\partial_z\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,-m'}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{A.13}$$

The real-valued P_N-equation have an intricate structure which causes many terms to cancel out. We take the first two terms (terms A.4) of the P_N-equations and apply the following orthogonality property of SH:

$$\int_\Omega Y_{\mathbb{R}}^{l_1,m_1}\overline{Y_{\mathbb{C}}^{l_2,m_2}}\mathrm{d}\boldsymbol{\omega} = \begin{cases} \frac{i}{\sqrt{2}}\left(\delta_{\substack{l_1=l_2\\ m_1=m_2}} - (-1)^{m_1}\delta_{\substack{l_1=l_2\\ m_1=-m_2}}\right), & \text{for } m_1 < 0 \\ \delta_{\substack{l_1=l_2\\ m_1=m_2}}, & \text{for } m_1 = 0 \\ \frac{1}{\sqrt{2}}\left(\delta_{\substack{l_1=l_2\\ m_1=-m_2}} + (-1)^{m_1}\delta_{\substack{l_1=l_2\\ m_1=m_2}}\right), & \text{for } m_1 > 0 \end{cases} \tag{A.14}$$

this way we get for the first two terms:

$$\begin{aligned}
&-i\alpha_c\partial_y\sum_{l=0}^{N}\sum_{m=-l}^{-1}L^{l,m}(\mathbf{x})\frac{1}{\sqrt{2}}i\delta_{\substack{l=l'-1\\ m=m'-1}} + i\alpha_c\partial_y\sum_{l=0}^{N}\sum_{m=-l}^{-1}L^{l,m}(\mathbf{x})\frac{1}{\sqrt{2}}i(-1)^m\delta_{\substack{l=l'-1\\ m=-m'+1}}\\
&-i\alpha_c\partial_y\sum_{l=0}^{N}L^{l,0}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{\substack{l=l'-1\\ 0=m'-1}} - i\alpha_c\partial_y\sum_{l=0}^{N}\sum_{m=1}^{l}L^{l,m}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{\substack{l=l'-1\\ m=-m'+1}}\\
&-i\alpha_c\partial_y\sum_{l=0}^{N}\sum_{m=1}^{l}L^{l,m}(\mathbf{x})\frac{1}{\sqrt{2}}(-1)^m\delta_{\substack{l=l'-1\\ m=m'-1}} + (-1)^{m'}i\alpha_c\partial_y\sum_{l=0}^{N}\sum_{m=-l}^{-1}L^{l,m}(\mathbf{x})\frac{1}{\sqrt{2}}i\delta_{\substack{l=l'-1\\ m=-m'+1}}\\
&-(-1)^{m'}i\alpha_c\partial_y\sum_{l=0}^{N}\sum_{m=-l}^{-1}L^{l,m}(\mathbf{x})\frac{1}{\sqrt{2}}i(-1)^m\delta_{\substack{l=l'-1\\ m=m'-1}} + (-1)^{m'}i\alpha_c\partial_y\sum_{l=0}^{N}L^{l,0}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{\substack{l=l'-1\\ 0=-m'+1}}\\
&+(-1)^{m'}i\alpha_c\partial_y\sum_{l=0}^{N}\sum_{m=1}^{l}L^{l,m}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{\substack{l=l'-1\\ m=m'-1}} + (-1)^{m'}i\alpha_c\partial_y\sum_{l=0}^{N}\sum_{m=1}^{l}L^{l,m}(\mathbf{x})\frac{1}{\sqrt{2}}(-1)^m\delta_{\substack{l=l'-1\\ m=-m'+1}}
\end{aligned}$$

We apply the delta function for the sums which run over the variable l:

$$\sum_{l=0}^{N}\sum_{m=a}^{b}L^{l,m}\delta_{\substack{l=x\\ m=y}} = \sum_{m=a}^{b}L^{x,m}\delta_{m=y} \tag{A.15}$$

We get for the first two terms of the transport term of the P_N-equation (term A.4):

$$
\begin{aligned}
&-i\alpha_c\partial_y\sum_{m=-l'+1}^{-1}L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}i\delta_{m=m'-1}+i\alpha_c\partial_y\sum_{m=-l'+1}^{-1}L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}i(-1)^m\delta_{m=-m'+1}\\
&-i\alpha_c\partial_yL^{l'-1,0}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{m'=1}-i\alpha_c\partial_y\sum_{m=1}^{l'-1}L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{m=-m'+1}-i\alpha_c\partial_y\sum_{m=1}^{l'-1}L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}(-1)^m\delta_{m=m'-1}\\
&+(-1)^{m'}i\alpha_c\partial_y\sum_{m=-l'+1}^{-1}L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}i\delta_{m=-m'+1}-(-1)^{m'}i\alpha_c\partial_y\sum_{m=-l'+1}^{-1}L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}i(-1)^m\delta_{m=m'-1}\\
&+(-1)^{m'}i\alpha_c\partial_yL^{l'-1,0}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{m'=1}+(-1)^{m'}i\alpha_c\partial_y\sum_{m=1}^{l'-1}L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{m=m'-1}\\
&+(-1)^{m'}i\alpha_c\partial_y\sum_{m=1}^{l'-1}L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}(-1)^m\delta_{m=-m'+1}
\end{aligned}
$$

The variables l' and m' specify a particular equation within the given set of P_N-equations. We remember that m' originated from multiplying the transport term with the real-valued SH basis function $Y_{\mathbb{R}}$ for the projection. The real-valued basis function is different for the sign of m' and we derived the transport term of the P_N-equations under the assumption of $m' < 0$ (different equations have to be derived for $m' = 0$ and $m' > 0$). We are able to greatly simplify the terms above when considering the parity of m' and that $m' < 0$.

The blue terms in the equation above all vanish, since the sums run over all negative (or positive) m, up to -1 (or l), while the Kronecker deltas in the blue terms only become non-zero for values $m > 0$ (or $m < 0$). This is because we derived these terms by multiplying with the real-valued SH basis function for $m' < 0$.

Consider the seventh and 10th term from the equation above. Due to $\delta_{m=m'-1}$ or $\delta_{m=-m'+1}$, an even m is selected if m' is odd and vice versa. Therefore, we have $(-1)^m(-1)^{m'} = -1$. This causes term one and seven (red) to vanish and term four and ten (black) to collapse into one term.

Therefore, the first two terms in the expansion (terms A.4), simplify to:

$$
\begin{aligned}
&-i\alpha_c\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}+(-1)^{m'}i\alpha_c\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&=-\frac{2}{\sqrt{2}}i\alpha_c\partial_y L^{l'-1,-m'+1}(\mathbf{x})\\
&=-\frac{2}{\sqrt{2}}i\frac{i}{\sqrt{2}}\frac{1}{2}c^{l'-1,m'-1}\partial_y L^{l'-1,-m'+1}(\mathbf{x})\\
&=\frac{1}{2}c^{l'-1,m'-1}\partial_y L^{l'-1,-m'+1}(\mathbf{x})
\end{aligned}
$$

The terms in equation A.5 are derived in the same way with the difference, that the signs are reversed and that we have $l'+1$ instead of $l'-1$. However, this does not affect the simplification:

$$
\begin{aligned}
&i\alpha_d\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}-(-1)^{m'}i\alpha_c\partial_y\sum_{l,m}L^{l,m}(\mathbf{x})\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\\
&=-\frac{2}{\sqrt{2}}i\alpha_d\partial_y L^{l'-1,-m'+1}(\mathbf{x})\\
&=-\frac{2}{\sqrt{2}}i\frac{i}{\sqrt{2}}\frac{1}{2}d^{l'+1,m'-1}\partial_y L^{l'+1,-m'+1}(\mathbf{x})\\
&=\frac{1}{2}d^{l'+1,m'-1}\partial_y L^{l'+1,-m'+1}(\mathbf{x})
\end{aligned}
$$

Carrying out the same simplifications for the remaining terms, results in the following real-valued P_N-equations for $m'<0$:

$$
\begin{aligned}
&-\frac{1}{2}c^{l'-1,m'-1}\partial_y L^{l'-1,-m'+1}+\frac{1}{2}d^{l'+1,m'-1}\partial_y L^{l'+1,-m'+1}-\frac{1}{2}\beta^{m'}e^{l'-1,m'+1}\partial_y L^{l'-1,-m'-1}\\
&+\frac{1}{2}\beta^{m'}f^{l'+1,m'+1}\partial_y L^{l'+1,-m'-1}+\frac{1}{2}c^{l'-1,m'-1}\partial_x L^{l'-1,m'-1}\\
&-\frac{1}{2}\delta_{m'\neq-1}e^{l'-1,m'+1}\partial_x L^{l'-1,m'+1}+\frac{1}{2}\delta_{m'\neq-1}f^{l'+1,m'+1}\partial_x L^{l'+1,m'+1}-\frac{1}{2}d^{l'+1,m'-1}\partial_x L^{l'+1,m'-1}\\
&+a^{l'-1,m'}\partial_z L^{l'-1,m'}+b^{l'+1,m'}\partial_z L^{l'+1,m'}
\end{aligned}
$$

with

$$\beta^x = \begin{cases} \frac{2}{\sqrt{2}}, & \text{for } |x| = 1 \\ 1, & \text{for } |x| \neq 1 \end{cases} \tag{A.16}$$

We now carry out the same derivation for the assumption of $m' > 0$. We multiply equation 3.2 with the definition of the real-valued SH basis for $m' > 0$ and get:

$$\int \left(\frac{1}{\sqrt{2}} \overline{Y_{\mathbb{C}}^{l',-m'}}(\boldsymbol{\omega}) + \frac{1}{\sqrt{2}} (-1)^{m'} \overline{Y_{\mathbb{C}}^{l',m'}}(\boldsymbol{\omega}) \right) \left(\boldsymbol{\omega}_x \partial_x L(\mathbf{x},\boldsymbol{\omega}) + \boldsymbol{\omega}_y \partial_y L(\mathbf{x},\boldsymbol{\omega}) + \boldsymbol{\omega}_z \partial_z L(\mathbf{x},\boldsymbol{\omega}) \right) \mathbf{d}\boldsymbol{\omega}$$

We expand the integrand and split the integral. Then we apply the recursive relation from equation 3.12 and get:

$$
\begin{aligned}
&\frac{1}{\sqrt{2}}\frac{1}{2}c^{l'-1,-m'-1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{1}{\sqrt{2}}\frac{1}{2}d^{l'+1,-m'-1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{1}{\sqrt{2}}\frac{1}{2}e^{l'-1,-m'+1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{1}{\sqrt{2}}\frac{1}{2}f^{l'+1,-m'+1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{1}{\sqrt{2}}\frac{i}{2}c^{l'-1,-m'-1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{1}{\sqrt{2}}\frac{i}{2}d^{l'+1,-m'-1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{1}{\sqrt{2}}\frac{i}{2}e^{l'-1,-m'+1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{1}{\sqrt{2}}\frac{i}{2}f^{l'+1,-m'+1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{1}{\sqrt{2}}a^{l'-1,-m'}\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,-m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{1}{\sqrt{2}}b^{l'+1,-m'}\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,-m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{1}{\sqrt{2}}(-1)^{m'}\frac{1}{2}c^{l'-1,m'-1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{1}{\sqrt{2}}(-1)^{m'}\frac{1}{2}d^{l'+1,m'-1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{1}{\sqrt{2}}(-1)^{m'}\frac{1}{2}e^{l'-1,m'+1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{1}{\sqrt{2}}(-1)^{m'}\frac{1}{2}f^{l'+1,m'+1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{1}{\sqrt{2}}(-1)^{m'}\frac{i}{2}c^{l'-1,m'-1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{1}{\sqrt{2}}(-1)^{m'}\frac{i}{2}d^{l'+1,m'-1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{1}{\sqrt{2}}(-1)^{m'}\frac{i}{2}e^{l'-1,m'+1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{1}{\sqrt{2}}(-1)^{m'}\frac{i}{2}f^{l'+1,m'+1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{1}{\sqrt{2}}(-1)^{m'}a^{l'-1,m'}\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{1}{\sqrt{2}}(-1)^{m'}b^{l'+1,m'}\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}
\end{aligned}
$$

We simplify these using the identities from equation A.3:

$$
\begin{aligned}
&\alpha_c\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}-(-1)^{m'}\alpha_c\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\alpha_d\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}+(-1)^{m'}\alpha_d\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\alpha_e\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}+(-1)^{m'}\alpha_e\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\alpha_f\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}-(-1)^{m'}\alpha_f\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-i\alpha_c\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}-(-1)^{m'}i\alpha_c\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+i\alpha_d\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}+(-1)^{m'}i\alpha_d\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-i\alpha_e\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}-(-1)^{m'}i\alpha_e\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+i\alpha_f\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}+(-1)^{m'}i\alpha_f\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\alpha_a\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,-m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}+(-1)^{m'}\alpha_a\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\alpha_b\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,-m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}+(-1)^{m'}\alpha_b\int \partial_z L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}
\end{aligned}
$$

with

$$
\begin{gathered}
\alpha_c=\frac{1}{\sqrt{2}}\frac{1}{2}c^{l'-1,-m'-1},\qquad \alpha_e=\frac{1}{\sqrt{2}}\frac{1}{2}e^{l'-1,-m'+1},\qquad \alpha_d=\frac{1}{\sqrt{2}}\frac{1}{2}d^{l'+1,-m'-1}\\
\alpha_f=\frac{1}{\sqrt{2}}\frac{1}{2}f^{l'+1,-m'+1},\qquad \alpha_a=\frac{1}{\sqrt{2}}a^{l'-1,-m'},\qquad \alpha_b=\frac{1}{\sqrt{2}}b^{l'+1,-m'}
\end{gathered}
$$

We substitute the radiance field function L with its spherical harmonics expansion and arrive at the following expression after further expansions and transformations:

$$
\begin{aligned}
&\alpha_c \partial_x \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} - (-1)^{m'} \alpha_c \partial_x \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} \\
&-\alpha_d \partial_x \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} + (-1)^{m'} \alpha_d \partial_x \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} \\
&-\alpha_e \partial_x \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} + (-1)^{m'} \alpha_e \partial_x \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} \\
&+\alpha_f \partial_x \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} - (-1)^{m'} \alpha_f \partial_x \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} \\
&-i\alpha_c \partial_y \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'-1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} - (-1)^{m'} i\alpha_c \partial_y \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} \\
&+i\alpha_d \partial_y \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'+1,-m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} + (-1)^{m'} i\alpha_d \partial_y \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} \\
&-i\alpha_e \partial_y \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'-1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} - (-1)^{m'} i\alpha_e \partial_y \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} \\
&+i\alpha_f \partial_y \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'+1,-m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} + (-1)^{m'} i\alpha_f \partial_y \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} \\
&+\alpha_a \partial_z \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'-1,-m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} + (-1)^{m'} \alpha_a \partial_z \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'-1,m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} \\
&+\alpha_b \partial_z \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'+1,-m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega} + (-1)^{m'} \alpha_b \partial_z \sum_{l,m} L^{l,m} \int Y_{\mathbb{R}}^{l,m} \overline{Y_{\mathbb{C}}^{l'+1,m'}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}
\end{aligned}
$$

Again we apply the identity given in equation A.14. For the first two terms we for example get:

$$
\begin{aligned}
&\alpha_c\partial_x \sum_{m=-l'+1}^{-1} L^{l'-1,m}(\mathbf{x})\frac{i}{\sqrt{2}}\delta_{m=-m'-1} - \alpha_c\partial_x \sum_{m=-l'+1}^{-1} L^{l'-1,m}(\mathbf{x})\frac{i}{\sqrt{2}}(-1)^m\delta_{m=m'+1}\\
&+\alpha_c\partial_x L^{l'-1,0}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{-m'=1} + \alpha_c\partial_x\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{m=m'+1} + \alpha_c\partial_x\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}(-1)^m\delta_{m=-m'-1}\\
&-(-1)^{-m'}\alpha_c\partial_x \sum_{m=-l'+1}^{-1} L^{l'-1,m}(\mathbf{x})\frac{i}{\sqrt{2}}\delta_{m=m'+1} + (-1)^{-m'}\alpha_c\partial_x \sum_{m=-l'+1}^{-1} L^{l'-1,m}(\mathbf{x})\frac{i}{\sqrt{2}}(-1)^m\delta_{m=-m'-1}\\
&-(-1)^{-m'}\alpha_c\partial_x L^{l'-1,0}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{-m'=1} - (-1)^{-m'}\alpha_c\partial_x\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{m=-m'-1}\\
&-(-1)^{-m'}\alpha_c\partial_x\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}(-1)^m\delta_{m=m'+1}
\end{aligned}
$$

As with the $m' < 0$ case, the blue and red terms cancel each other out, leaving only the black terms. The first two terms of the real-valued P_N-equations for the transport term therefore are:

$$
\begin{aligned}
&\alpha_c\partial_x\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}\delta_{m=m'+1} - (-1)^{-m'}\alpha_c\partial_x\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})\frac{1}{\sqrt{2}}(-1)^m\delta_{m=m'+1}\\
&= \frac{2}{\sqrt{2}}\alpha_c\partial_x L^{l'-1,m'+1}(\mathbf{x}) = \frac{2}{\sqrt{2}}\left(\frac{1}{2\sqrt{2}}c^{l'-1,-m'-1}\right)\partial_x L^{l'-1,m'+1}(\mathbf{x})\\
&= \frac{1}{2}c^{l'-1,-m'-1}L^{l'-1,m'+1}(\mathbf{x})
\end{aligned}
$$

Following this through for the remaining terms gives us the real-valued P_N-equations for $m' > 0$:

$$
\begin{aligned}
&\frac{1}{2}c^{l'-1,-m'-1}\partial_x L^{l'-1,m'+1}(\mathbf{x}) - \frac{1}{2}d^{l'+1,-m'-1}\partial_x L^{l'+1,m'+1}(\mathbf{x}) - \frac{1}{2}\beta^{m'}e^{l'-1,m'-1}\partial_x L^{l'-1,m'-1}(\mathbf{x})\\
&\frac{1}{2}\beta^{m'}f^{l'+1,-m'+1}\partial_x L^{l'+1,m'-1}(\mathbf{x})\frac{1}{2}c^{l'-1,-m'-1}\partial_y L^{l'-1,-m'-1}(\mathbf{x}) - \frac{1}{2}d^{l'+1,-m'-1}\partial_y L^{l'+1,-m'-1}(\mathbf{x})\\
&\delta_{m'\neq 1}\frac{1}{2}e^{l'-1,-m'+1}\partial_y L^{l'-1,-m'+1}(\mathbf{x}) - \delta_{m'\neq 1}\frac{1}{2}f^{l'+1,-m'+1}\partial_y L^{l'+1,-m'+1}(\mathbf{x})a^{l'-1,-m'}\partial_z L^{l'-1,m'}(\mathbf{x})\\
&b^{l'+1,-m'}\partial_z L^{l'+1,m'}(\mathbf{x})
\end{aligned}
$$

Finally the $m' = 0$ case needs to be derived. The derivation starts very similar to the complex-valued P_N-equations as in this case, the real-valued SH basis function is identical to the complex-valued SH basis function. We multiply equation 3.2 with the definition of the real-valued SH basis for $m' = 0$ and get:

$$\int \overline{Y_{\mathbb{C}}^{l',m'}}(\boldsymbol{\omega})\big(\boldsymbol{\omega}_x\partial_x L(\mathbf{x},\boldsymbol{\omega})+\boldsymbol{\omega}_y\partial_y L(\mathbf{x},\boldsymbol{\omega})+\boldsymbol{\omega}_z\partial_z L(\mathbf{x},\boldsymbol{\omega})\big)\mathbf{d}\boldsymbol{\omega}$$

Expanding the integrand and applying the recursion relation (equation A.3) produces the following set of terms:

$$\begin{aligned}
&\frac{1}{2}c^{l'-1,m'-1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}-\frac{1}{2}e^{l'-1,m'+1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{1}{2}d^{l'+1,m'-1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}+\frac{1}{2}f^{l'+1,m'+1}\int \partial_x L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{i}{2}c^{l'-1,m'-1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}-\frac{i}{2}e^{l'-1,m'+1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{i}{2}d^{l'+1,m'-1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}+\frac{i}{2}f^{l'+1,m'+1}\int \partial_y L(\mathbf{x},\boldsymbol{\omega})\overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+a^{l'-1,m'}\int \overline{Y_{\mathbb{C}}^{l'-1,m'}}(\boldsymbol{\omega})\partial_z L(\mathbf{x},\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}+b^{l'+1,m'}\int \overline{Y_{\mathbb{C}}^{l'+1,m'}}(\boldsymbol{\omega})\partial_z L(\mathbf{x},\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}
\end{aligned}$$

Again we will replace the radiance field L with its real-valued SH projection and get:

$$\begin{aligned}
&\frac{1}{2}c^{l'-1,m'-1}\partial_x\sum_{l,m}L^{l,m}\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}-\frac{1}{2}e^{l'-1,m'+1}\partial_x\sum_{l,m}L^{l,m}\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{1}{2}d^{l'+1,m'-1}\partial_x\sum_{l,m}L^{l,m}\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}+\frac{1}{2}f^{l'+1,m'+1}\partial_x\sum_{l,m}L^{l,m}\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&-\frac{i}{2}c^{l'-1,m'-1}\partial_y\sum_{l,m}L^{l,m}\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}-\frac{i}{2}e^{l'-1,m'+1}\partial_y\sum_{l,m}L^{l,m}\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'-1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+\frac{i}{2}d^{l'+1,m'-1}\partial_y\sum_{l,m}L^{l,m}\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,m'-1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}+\frac{i}{2}f^{l'+1,m'+1}\partial_y\sum_{l,m}L^{l,m}\int Y_{\mathbb{R}}^{l,m}\overline{Y_{\mathbb{C}}^{l'+1,m'+1}}(\boldsymbol{\omega})\mathbf{d}\boldsymbol{\omega}\\
&+a^{l'-1,m'}\partial_z\sum_{l,m}L^{l,m}\int \overline{Y_{\mathbb{C}}^{l'-1,m'}}(\boldsymbol{\omega})Y_{\mathbb{R}}^{l,m}\mathbf{d}\boldsymbol{\omega}+b^{l'+1,m'}\partial_z\sum_{l,m}L^{l,m}\int \overline{Y_{\mathbb{C}}^{l'+1,m'}}(\boldsymbol{\omega})Y_{\mathbb{R}}^{l,m}\mathbf{d}\boldsymbol{\omega}
\end{aligned}$$

These terms also have an intricate structure where many terms cancel out and simplify. This is seen once we apply the SH orthogonality property (equationequation A.14) and further consider that $m' = 0$. We show this for the first two terms, which expand to:

$$
\begin{aligned}
&\frac{1}{2}c^{l'-1,-1}\frac{i}{\sqrt{2}}\partial_x\left(\sum_{m=-l'+1}^{-1} L^{l'-1,m}(\mathbf{x})\delta_{-1,m}\right)\\
&-\frac{1}{2}c^{l'-1,-1}\frac{i}{\sqrt{2}}\partial_x\left(\sum_{m=-l'+1}^{-1} L^{l'-1,m}(\mathbf{x})(-1)^m\delta_{-1,-m}\right)\\
&\frac{1}{2}c^{l'-1,-1}\partial_x\left(L^{l'-1,0}(\mathbf{x})\delta_{-1,0}\right)\\
&\frac{1}{2}c^{l'-1,-1}\frac{1}{\sqrt{2}}\partial_x\left(\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})\delta_{-1,-m}\right)\\
&\frac{1}{2}c^{l'-1,-1}\frac{1}{\sqrt{2}}\partial_x\left(\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})(-1)^m\delta_{-1,m}\right)\\
&-\frac{1}{2}e^{l'-1,1}\frac{i}{\sqrt{2}}\partial_x\left(\sum_{m=-l'+1}^{-1} L^{l'-1,m}(\mathbf{x})\delta_{1,m}\right)\\
&\frac{1}{2}e^{l'-1,1}\frac{i}{\sqrt{2}}\partial_x\left(\sum_{m=-l'+1}^{-1} L^{l'-1,m}(\mathbf{x})(-1)^m\delta_{1,-m}\right)\\
&-\frac{1}{2}e^{l'-1,1}\partial_x\left(L^{l'-1,0}(\mathbf{x})\delta_{1,0}\right)\\
&-\frac{1}{2}e^{l'-1,1}\frac{1}{\sqrt{2}}\partial_x\left(\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})\delta_{1,-m}\right)\\
&-\frac{1}{2}e^{l'-1,1}\frac{1}{\sqrt{2}}\partial_x\left(\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})(-1)^m\delta_{1,m}\right)
\end{aligned}
$$

Again the blue terms vanish since the delta functions will never be non-zero under the sums. The red terms cancel each other out since $c^{l,-1} = e^{l,1}$ and $-1^m = -1$ for $m = -1$. The terms in black simplify to:

$$
\begin{aligned}
&\frac{1}{2}c^{l'-1,-1}\frac{1}{\sqrt{2}}\partial_x\left(\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})\delta_{-1,-m}\right)-\frac{1}{2}e^{l'-1,1}\frac{1}{\sqrt{2}}\partial_x\left(\sum_{m=1}^{l'-1} L^{l'-1,m}(\mathbf{x})(-1)^m\delta_{1,m}\right)\\
&=\frac{1}{2}c^{l'-1,-1}\frac{1}{\sqrt{2}}\partial_x L^{l'-1,1}(\mathbf{x})-\frac{1}{2}c^{l'-1,-1}\frac{1}{\sqrt{2}}\partial_x L^{l'-1,1}(\mathbf{x})(-1)^1\\
&=\frac{1}{\sqrt{2}}c^{l'-1,-1}\partial_x L^{l'-1,1}(\mathbf{x})
\end{aligned}
$$

Similar simplifications apply to the remaining terms of the SH expansion of the transport term for $m = 0$, resulting in the final expression:

$$\begin{aligned}
&\frac{1}{\sqrt{2}}c^{l'-1,-1}\partial_x L^{l'-1,1}(\mathbf{x}) - \frac{1}{\sqrt{2}}d^{l'+1,-1}\partial_x L^{l'+1,1}(\mathbf{x})\\
&\frac{1}{\sqrt{2}}c^{l'-1,-1}\partial_y L^{l'-1,-1}(\mathbf{x}) - \frac{1}{\sqrt{2}}d^{l'+1,-1}\partial_y L^{l'+1,-1}(\mathbf{x})\\
&a^{l'-1,0}\partial_z L^{l'-1,0}(\mathbf{x}) + b^{l'+1,0}\partial_z L^{l'+1,0}(\mathbf{x})
\end{aligned}$$

Final equation

Putting all projected terms from previous subsections together, we get for $m < 0$:

$$\begin{aligned}
&-\frac{1}{2}c^{l-1,m-1}\partial_y L^{l-1,-m+1} + \frac{1}{2}d^{l+1,m-1}\partial_y L^{l+1,-m+1} - \frac{1}{2}\beta^m e^{l-1,m+1}\partial_y L^{l-1,-m-1}\\
&+\frac{1}{2}\beta^m f^{l+1,m+1}\partial_y L^{l+1,-m-1} + \frac{1}{2}c^{l-1,m-1}\partial_x L^{l-1,m-1}\\
&-\frac{1}{2}\delta_{m\neq-1} e^{l-1,m+1}\partial_x L^{l-1,m+1} + \frac{1}{2}\delta_{m\neq-1} f^{l+1,m+1}\partial_x L^{l+1,m+1} - \frac{1}{2}d^{l+1,m-1}\partial_x L^{l+1,m-1}\\
&+a^{l-1,m}\partial_z L^{l-1,m} + b^{l+1,m}\partial_z L^{l+1,m} + \sigma_t L^{l,m} - \sigma_s \lambda_l \rho^{l,0} L^{l,m} = Q^{l,m}
\end{aligned} \tag{A.17}$$

for $m = 0$:

$$\begin{aligned}
&\frac{1}{\sqrt{2}}c^{l-1,-1}\partial_x L^{l-1,1} - \frac{1}{\sqrt{2}}d^{l+1,-1}\partial_x L^{l+1,1}\frac{1}{\sqrt{2}}c^{l-1,-1}\partial_y L^{l-1,-1}\\
&-\frac{1}{\sqrt{2}}d^{l+1,-1}\partial_y L^{l+1,-1} a^{l-1,0}\partial_z L^{l-1,0} + b^{l+1,0}\partial_z L^{l+1,0}\\
&+\sigma_t L^{l,m} - \sigma_s \lambda_l \rho^{l,0} L^{l,m} = Q^{l,m}
\end{aligned} \tag{A.18}$$

and for $m > 0$:

$$\begin{aligned}
&\frac{1}{2}c^{l-1,-m-1}\partial_x - \frac{1}{2}d^{l+1,-m-1}\partial_x L^{l+1,m+1} - \frac{1}{2}\beta^m e^{l-1,m-1}\partial_x L^{l-1,m-1} \\
&+\frac{1}{2}\beta^m f^{l+1,-m+1}\partial_x L^{l+1,m-1} + \frac{1}{2}c^{l-1,-m-1}\partial_y L^{l-1,-m-1} \\
&-\frac{1}{2}d^{l+1,-m-1}\partial_y L^{l+1,-m-1} + \delta_{m\neq 1}\frac{1}{2}e^{l-1,-m+1}\partial_y L^{l-1,-m+1} \\
&-\delta_{m\neq 1}\frac{1}{2}f^{l+1,-m+1}\partial_y L^{l+1,-m+1} + a^{l-1,-m}\partial_z L^{l-1,m} + b^{l+1,-m}\partial_z L^{l+1,m} \\
&+\sigma_t L^{l,m} - \sigma_s \lambda_l \rho^{l,0} L^{l,m} = Q^{l,m}
\end{aligned} \tag{A.19}$$

with

$$\beta^x = \begin{cases} \frac{2}{\sqrt{2}}, & \text{for } |x| = 1 \\ 1, & \text{for } |x| \neq 1 \end{cases}, \quad \delta_{x\neq y} = \begin{cases} 1, & \text{for } x \neq y \\ 0, & \text{for } x = y \end{cases}$$

and

$$a^{l,m} = \sqrt{\frac{(l-m+1)(l+m+1)}{(2l+1)(2l-1)}} \qquad b^{l,m} = \sqrt{\frac{(l-m)(l+m)}{(2l+1)(2l-1)}}$$

$$c^{l,m} = \sqrt{\frac{(l+m+1)(l+m+2)}{(2l+3)(2l+1)}} \qquad d^{l,m} = \sqrt{\frac{(l-m)(l-m-1)}{(2l+1)(2l-1)}}$$

$$e^{l,m} = \sqrt{\frac{(l-m+1)(l-m+2)}{(2l+3)(2l+1)}} \qquad f^{l,m} = \sqrt{\frac{(l+m)(l+m-1)}{(2l+1)(2l-1)}}$$

$$\lambda_l = \sqrt{\frac{4\pi}{2l+1}}$$

The equations can be written in a more compact form by using $\pm$ and $\mp$ to write the equations for $m < 0$ and $m > 0$ as one (see Koerner et al.[48]).

Appendix B

Appendix: Multigrid solver

In this section a multigrid solver for the diffusion approximation is developed. Multigrid schemes are the most efficient iterative methods for solving diffusion-type equations. The core idea is to propagate the error on a coarse grid during each iteration and take that propagated error into account, when doing an iteration step on the original high resolution grid. Particularly useful is the fact that the multigrid method can be employed on any system of linear equations as long as so called restriction and interpolation operators are being provided.

The automatic discretization machinery, which was developed as part of the P_N-solver in section 3.5 takes arbitrary systems of potentially coupled, linear partial differential equations as input. The diffusion equation, which has been derived in section 4.4 is such a linear scalar partial differential equation and therefore can be fed into the automated stencil code generation from section 3.5 to produce valid stencil code (independent of mesh resolution).

The idea behind the multigrid method is to accelerate the convergence by using a coarse grid on which information propagates faster throughout the spatial domain. The coarse solution is then used for updating the original grid.

Quantities on the original fine mesh are denoted with the subscript F. Therefore the original system of equations becomes:

$$A_F \mathbf{u}_F = \mathbf{b}_F$$

The multigrid method is an iterative method. This is expressed with a superscript index i, which denotes the current multigrid iteration. The first step in the multigrid cycle is to partially converge the solution by applying a number of iterations of an iterative method, such as Conjugate-Gradient or Gauss-Seidel. The partially converged solution in the first multigrid cycle is denoted $\mathbf{u}_F^i$. Then the residual is computed on the fine mesh:

$$\mathbf{r}_F^i = \mathbf{b}_F - A_F \mathbf{u}_F^i$$

The idea behind multigrid is to use the coarse grid to smooth out the error and use the result as a correction for the next iterations on the fine mesh. To facilitate this, the residual on the fine mesh is first transferred onto the coarse mesh. This is done by downsampling the fine mesh residual (often also called interpolation). This operation can be represented as a non-symmetric matrix M. Quantities on the coarse mesh are denoted with the subscript C:

$$\mathbf{r}_C^i = M_{F \to C} \mathbf{r}_F^i$$

With bringing the residual onto the coarse mesh, the next step is to solve the correction equation to full convergence. The correction equation is derived by looking at the system on the coarse mesh, using the partially converged solution $\mathbf{u}_C^i$:

$$A_C \mathbf{u}_C^i = \mathbf{b}_C$$

Rearranging this equation allows us to use the downsampled residual as the right hand side vector:

$$\mathbf{b}_C - A_C \mathbf{u}_C^i = \mathbf{r}_C^i$$

When substituting $\mathbf{b}_C$ with $A_C\mathbf{u}_C = \mathbf{b}_C$ (using the fully converged yet unknown solution on the coarse grid $\mathbf{u}_C$) the following is shown:

$$\begin{aligned} A_C\mathbf{u}_C - A_C\mathbf{u}_C^i &= \mathbf{r}_C^i \\ A_C\left(\mathbf{u}_C - \mathbf{u}_C^i\right) &= \mathbf{r}_C^i \\ A_C\mathbf{e}_C^i &= \mathbf{r}_C^i \end{aligned} \tag{B.1}$$

The quantity $\mathbf{e}_C^i$ is the unknown error between the final solution $\mathbf{u}_C$ and its partially converged approximation $\mathbf{u}_C^i$. Equation B.1 is called the correction equation and is solved to full convergence. The computed error is upsampled onto the finer grid, using the upsampling operator $M_{C\to F}$:

$$\mathbf{e}_F^i = M_{C\to F}\mathbf{e}_C^i$$

Using the upsampled error a correction step to the solution on the fine grid can be applied:

$$\begin{aligned} \mathbf{e}_F^i &= \mathbf{u}_F - \mathbf{u}_F^i \\ \mathbf{u}_F^{i+1} &= \mathbf{u}_F = \mathbf{e}_F^i + \mathbf{u}_F^i \end{aligned}$$

This process is then repeated by starting a new multigrid iteration with the corrected approximation $\mathbf{u}_F^{i+1}$. In summary a single multigrid iteration consists of the following steps:

1. Solve the original system on the fine grid, using a small number of iteration steps to bring the solution $\mathbf{u}_F^i$ to partial convergence.

2. Compute the residual $\mathbf{r}_F^i = \mathbf{b}_F - A_F\mathbf{u}_F^i$ on the fine grid and transfer it onto the coarse grid $\mathbf{r}_C^i = M_{F\to C}\mathbf{r}_F^i$.

3. Solve the correction equation $A_C\mathbf{e}_C^i = \mathbf{r}_C^i$ for $\mathbf{e}_C^0$ to full convergence on the coarse grid.

4. Transfer the result to the fine grid $\mathbf{e}_F^i = M_{C\to F}\mathbf{e}_C^i$.

5. Use $\mathbf{e}_F^i$ to apply the correction $\mathbf{u}_F^{i+1} = \mathbf{e}_F^i + \mathbf{u}_F^i$.

This is a two-level multigrid step, which could be extended to more levels by applying the same idea recursively to the coarse mesh solve. This is often done, so that the linear system of the coarsest level is so small that a direct method for solving it can be applied very efficiently. This results in the characteristic V-cycle shown in figure B.1.

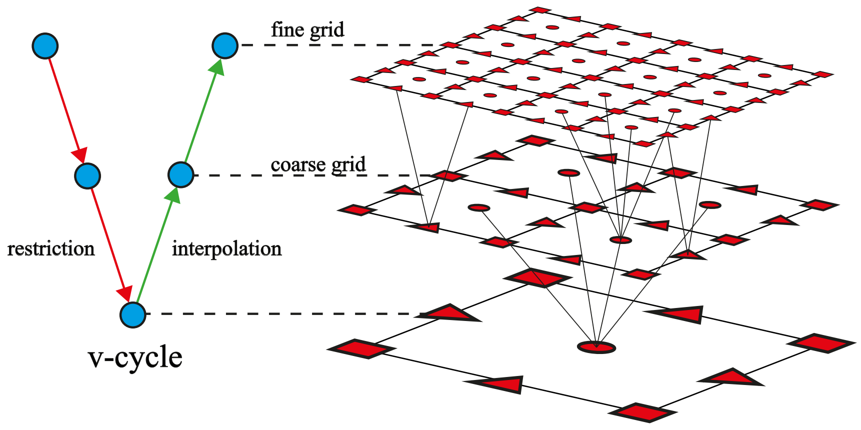


Figure B.1: *The staggered coarse grids for the multigrid v-cycle are computed from interpolating coefficients at their coarse grid locations.*

Building the upsampling and downsampling matrices $M_{C\to F}$ and $M_{F\to C}$ is very similar to the problem of interpolating discretized variables at specific staggered grid locations for the automatic discretization in section 3.5.2 (figure 3.8). In fact, the *PNSystemBuilder* class presented in section 3.5.5 has been extended to facilitate the automatic generation of the upsampling and downsampling operator matrices from a given discretization and a given vector of coefficients with their respective staggered grid locations. For the interpolation weights to not vary per voxel, the fine grid has to be exactly of double resolution. Further the number of multigrid levels defines the number of subdivisions required, and imposes additional constraints on the resolution. To support the maximum number of multigrid levels possible, the resolution has to be the power of two.