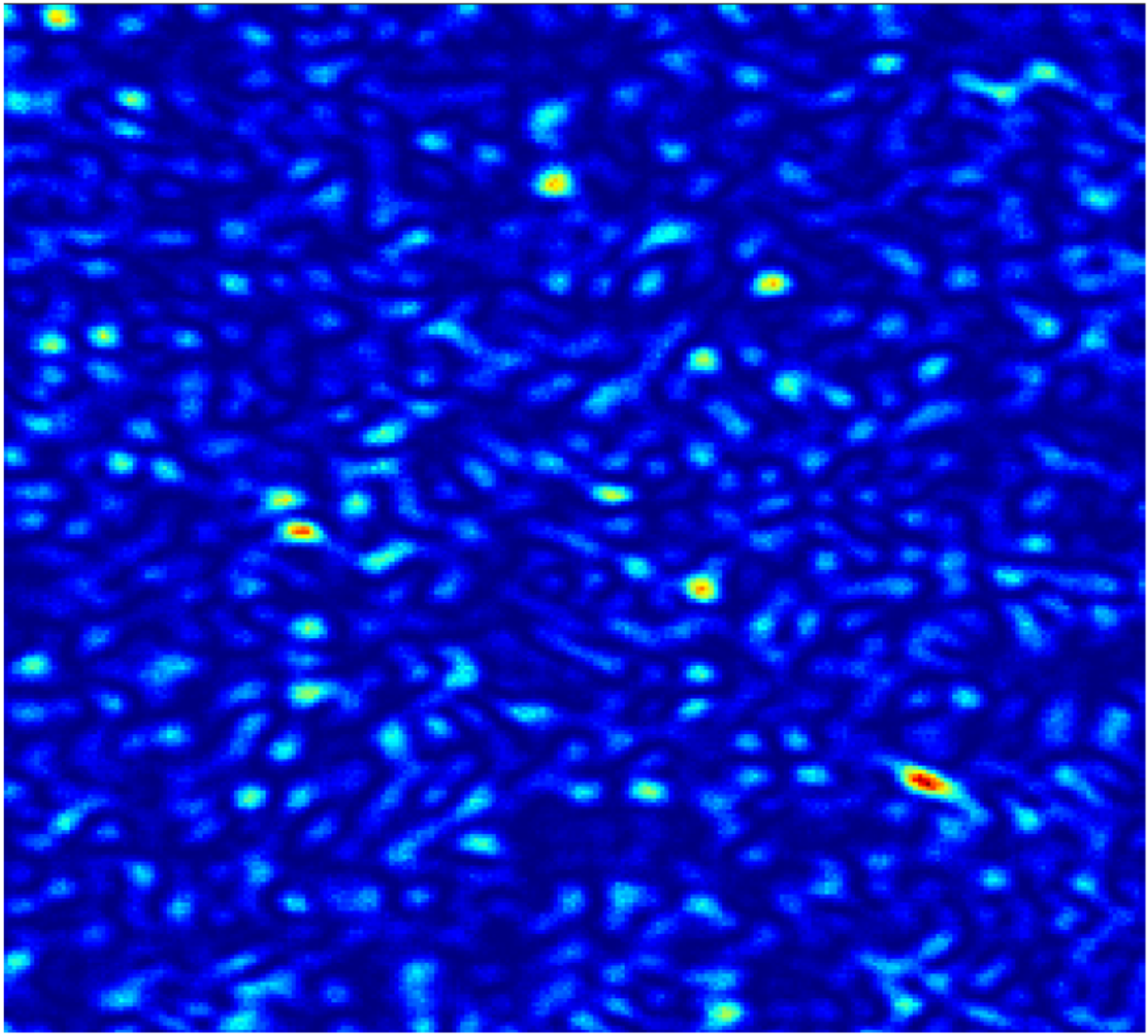
Interferometric Scattering of Light by an Ensemble of Flowing Spherical Particles: A Numerical Study

MSc Thesis Report

Kevin van As

Technische Universiteit Delft



TU Delft Delft University of Technology

Challenge the future

Interferometric Scattering of Light by an Ensemble of Flowing Spherical Particles: A Numerical Study

MSc Thesis Report

by

Kevin van As

in partial fulfillment of the requirements for the degree of

Master of Science
in Applied Physics

at the Delft University of Technology,
to be defended publicly on Monday July 13th, 2015 at 14:00.

Student number:	4076311	
Supervisors:	Dr. S. Kenjeres,	TU Delft (Transport Phenomena)
	Dr. N. Bhattacharya,	TU Delft (Optics)
Thesis committee:	Prof. dr. ir. C. R. Kleijn,	TU Delft (Transport Phenomena)
	Prof. dr. H. P. Urbach,	TU Delft (Optics)

TU Delft Delft University of Technology

Abstract

To study blood-related diseases, measurement methods must be developped to study cardiac and hematological parameters of a patient. Such methods should preferably be non-destructive, in-vivo, cheap, accurate and real-time. Light has great potential to achieve this.

Using e.g. a pulse oximeter, it is possible to measure a patient's heartbeat. This device makes use of light absorption, which is a scalar quantity, in which many of the information contained within the light has been lost. If we on the other hand look at scattered light, we have an entire field of information.

We wish to explore whether Mie theory, which is an exact solution that describes the light scattered by a single spherical particle, can be used to accurately describe the light scattered by an ensemble of spherical particles. We take a numerical approach in which we evolve the exact solution over space. If we then apply a far-field assumption, it becomes possible to iteratively apply Mie theory to compute the scattered field of an ensemble of spherical particles. This way, an interferometric multiscattering code is obtained.

Based on the results of this study, we are confident that this method is useful for studying light scattering by an ensemble of spheres. More time is however required to make a more quantitative validation to determine the accuracy of the method.

Preface

My name is Kevin van As, and I am a student in the Excellence Track of Solid & Fluid Mechanics, which is a track within the MSc Applied Physics at TU Delft, The Netherlands. Before the present studies, I graduated with distinction as a BSc of Applied Physics at TU Delft. My BSc Thesis was on turbulent flow around fractal trees, in which we added leaves to the existing model.

During a MSc course, 'Advanced Electrodynamics', I got to talk to one of the teachers and asked her what kind of research she was conducting. The answer was 'studying blood flow experimentally'. This teacher was Dr. Nandini Bhattacharya, of the Optics research group. I happened to know that Dr. Sasa Kenjeres from the Transport Phenomena group, who was the supervisor of my BSc Thesis project, studies blood flow as well, but then numerically. Then I asked her if she knew him. It turned out that they didn't know each other.

As a consequence of this question of mine, the two of them got in touch. A collaboration began, and interdisciplinary projects were defined. One of those projects, is the present MSc Thesis project, for which I was asked. It is funny how those things get back to you. I figured that this project would be particularly interesting, because it would take a first step in combining two separate areas of physics: fluid dynamics and optics. That, and I'm half a programmer, which made this project very suitable for my interests.

My time was largely spent developing the Optics code, which required quite some time for me to get a clear understanding of how Mie theory could be implemented. The developed code works a lot different than anything I was used to, since it does in fact evolve an exact solution over space. This is different from e.g. in computational fluid dynamics, where partial differential equations are discretised and solved on a grid. For the Optics code, there is no grid. The exact solution lives in a continuous space.

Linking it to an existing Fluids code was relatively easy, since I already had some experience in an existing computational fluid dynamics program: OpenFOAM. I simply wrote an external script to extract the required information from its file system, and converted it to the input format of the Optics code.

All this programming left extremely little time for producing results. In fact, all results related to the combined physics have been generated in the final 2-3 weeks, which includes writing that part of the report. I wish I had more time to produce more results for this MSc Thesis, because the current results feel minimalistic. I am however very much satisfied with the results, given the amount of time used to generate them.

For the hurried reader

In order to only grab the essence of the present work, without having to read through all the details, the hurried reader is advised to engage in the journey outlined below. It should be noted that each chapter (except for the theory) has a summarising section at the end of the chapter.

1. It is always a good idea to start with the introduction in Chap. 1, in particular the research question printed in italic.

2. The theory chapter, Chap. 2, may almost be skipped altogether. If any knowledge from the theory chapter is required at some point in the report, the reader will be referred to it. It is nonetheless useful to know what Mie theory is: it is an exact solution which describes how light is scattered by a single spherical particle, cf. Eq. (2.42). And how the scattering angle, θ_s, is defined: see Fig. 2.2.

3. Regarding the code description in Chap. 3, Fig. 3.1 illustrates what the developed Optics code does in a sketch. Fig. 3.4 shows how the amplitude scattering matrix depends on the scattering angle. This is of crucial importance to explain some of the results. Finally, Sec. 3.3 describes how fluid dynamics is combined with optics.

4. Before moving on to the results, I'd recommend reading the conclusions in Chap. 7 first. This will give the reader a better idea what to expect from the present research.

5. Chap. 4 shows a small validation study based on an exact solution with a far-field approximation applied to it. Observing the figures and reading its conclusion should be sufficient.

6. In Chaps. 5 and 6 the most important results of this thesis are shown, by means of a second validation study. The reader interested in the results of this study should focus on these chapters.

7. If the reader is interested in possible follow-ups, Sec. 7.2 provides a very extensive recommendations section. Since this MSc Thesis project consisted largely of developing a new code, many follow-ups are possible.

Acknowledgements

I'd like to thank some people in particular, in addition to my two supervisors, for their contributions to this project.

From the Optics Research Group, these are Prof. dr. Paul Urbach, Mahsa Nemati and Gyllion Loozen, mostly for the regular meetings during the beginning of the project. From the Transport Phenomena Research Group, these are Anton Kidess for his help with getting OpenFOAM running and some Linux-related problems, and Koen Schutte, e.g. for saving the day when I worked some of my magic in such a way that I lost write access in my own directories (yeah…that wasn't supposed to happen).

Contents

List of Figures

List of Tables

Nomenclature

a	$[\mathrm{m}^1]$ Spherical particle radius, page 5
α	$[-]$ Dimensionless number: Womersley; Represents the ratio between transient inertial force and viscous force, page 13
α	$[\mathrm{rad}]$ Viewing angle (of the camera), page 39
a_n	$[-]$ Mie coefficient for the even term of the scattered field, page 6
b_n	$[-]$ Mie coefficient for the odd term of the scattered field, page 6
C	$[-]$ Speckle contrast, page 43
c	$[\mathrm{m}^1\mathrm{s}^{-1}]$ Speed of light in vacuum, page 4
χ	$[-]$ Electric susceptibility, page 3
c_n	$[-]$ Mie coefficient for the odd term of the internal field, page 6
D	$[\mathrm{m}^1]$ Characteristic length scale, page 13
$\Delta\varphi$	$[\mathrm{rad}]$ Phase difference, page 17
d_n	$[-]$ Mie coefficient for the even term of the internal field, page 6
$\vec{E}$	$[\mathrm{Vm}^{-1} = \mathrm{NC}^{-1} = \mathrm{m}^1\mathrm{kg}^1\mathrm{s}^{-3}\mathrm{A}^{-1}]$ Electric field, page 3
E_0	Electric field - magnitude when using complex notation, page 5
$\vec{E}_1$	Electric field - inside the scatterer, page 5
$\vec{E}_2$	Electric field - total outside the scatterer, page 4
$\vec{E}_i$	Electric field - incident, page 4
$e^{-\mathrm{i}\omega t}$	This report uses the minus sign convention in the complex wave notation, page 3
ϵ	$[\mathrm{m}^{-3}\mathrm{kg}^{-1}\mathrm{s}^4\mathrm{A}^2]$ Complex (electric) permittivity, page 3
ϵ_0	$[\mathrm{m}^{-3}\mathrm{kg}^{-1}\mathrm{s}^4\mathrm{A}^2]$ (Electric) permittivity of vacuum, page 3
$\vec{E}_s$	Electric field - scattered, page 4
f_i^{ext}	$[\mathrm{Nm}^{-1} = \mathrm{kg}^1\mathrm{s}^{-2}]$ External force per unit length, page 12
$\vec{H}$	$[\mathrm{m}^{-1}\mathrm{A}^1]$ Magnetic field, page 3
H	$[\mathrm{m}^1]$ Characteristic length scale (thickness), page 25
t	$[\mathrm{s}^1]$ Time, page 3
$h_n^{(1)}$	$[-]$ Spherical Bessel function of the third kind (or Hankel function of the first kind) and of order n, page 6
I	$[\mathrm{Wm}^{-2} = \mathrm{kg}^1\mathrm{s}^{-3}]$ Intensity, page 18
$\langle I \rangle_a$	$[\mathrm{Wm}^{-2} = \mathrm{kg}^1\mathrm{s}^{-3}]$ Mean intensity, as taken over the $\hat{a}$ direction., page 41
j_n	$[-]$ Spherical Bessel function of the first kind and of order n, page 6

k $[\mathrm{m}^{-1}]$ Wave number, $\frac{2\pi}{\lambda}$, page 4

k_0 $[\mathrm{m}^{-1}]$ Wave number in vacuum, $\frac{2\pi}{\lambda_0}$, page 4

κ $[-]$ $n\kappa$ is the complex part of the complex refractive index, page 4

L $[\mathrm{m}^1]$ Characteristic length scale (length), page 25

L $[\mathrm{m}^1]$ Distance between the center of the aperture and the camera; Used only in Chap. 4., page 31

L $[\mathrm{m}^1]$ Length of cylinder, page 10

λ $[\mathrm{m}^1]$ Wavelength, page 4

$\vec{M}$ $[-]$ Vector Spherical harmonic 1, page 5

M $[-]$ Number of pixels of the camera, page 19

m Sum index for the azimuthal term (Mie theory), page 5

m $[-]$ Relative refractive index $\frac{N_1}{N_2}$, page 7

m $[\mathrm{kg}^1]$ Mass, page 25

μ $[-]$ Short for $\cos\theta$, page 7

μ $[\mathrm{NA}^{-2} = \mathrm{m}^1\mathrm{kg}^1\mathrm{s}^{-2}\mathrm{A}^{-2}]$ (Magnetic) permeability, page 3

μ $[\mathrm{Pa}\cdot\mathrm{s} = \mathrm{m}^{-1}\mathrm{kg}^1\mathrm{s}^{-1}]$ (Dynamic) viscosity, page 12

μ_0 $[\mathrm{NA}^{-2} = \mathrm{m}^1\mathrm{kg}^1\mathrm{s}^{-2}\mathrm{A}^{-2}]$ (Magnetic) permeability of vacuum, page 4

$\vec{N}$ $[-]$ Vector Spherical harmonic 2, page 5

N $[-]$ Complex refractive index, page 4

N $[-]$ Number of particles, page 10

n Sum index for the radial term (Mie theory), page 5

n $[-]$ Real refractive index, page 4

n_C Cut-off index of the infinite Mie sum, page 16

$\hat{n}$ $[-]$ Unit normal vector to some surface, page 5

ν $[\mathrm{m}^2\mathrm{s}^{-1}]$ Kinematic viscosity, page 13

ω $[\mathrm{s}^{-1}]$ Angular frequency, page 3

P $[\mathrm{Pa} = \mathrm{m}^{-1}\mathrm{kg}^1\mathrm{s}^{-2}]$ Pressure, page 12

p $[-]$ Scattering order; The number of particles the light scattered upon before hitting the camera., page 17

ϕ $[-]$ Volume fraction; Hematocrit, page 10

φ $[\mathrm{rad}]$ Azimuthal cylindrical angle, page 8

φ $[\mathrm{rad}]$ Azimuthal spherical angle, page 5

π_n $[-]$ Short for $\frac{P_n^1(\cos\theta)}{\sin\theta}$, page 7

P_n^m $[-]$ Associated Legendre polynomial, page 5

$P(r)$ $[-]$ Probability density function with argument r, page 11

1

Introduction

Blood is of major importance for a human being's survival. Blood does not only supply all organs with energy, it is as well the vessel of the immune system. Therefore, any cardiac or hematological diseases can be critical. There are many ways to study these diseases, but ultimately all methods rely on the input of information. Similarly, whenever a patient goes to see a doctor with certain symptons, information is required for the doctor to properly diagnose the patient. That information, must come from measurements.

There exist numerous ways to obtain such information. For example, we could extract blood from a patient and analyse its content in a laboratory. Or we could make use of sound, medical sonography, and look at sound reflection and Doppler shifts to acquire information on the local blood flow velocity. Another possibility is magnetic resonance angiography (which is a subset of MRI), which excites hydrogen atoms and observes the emitted photons as a consequence of the natural de-excitation of the atoms. And then there is light. Using photoplethysmography (PPG), which sends light through the skin and blood vessel and measures the light absorption, it is possible to very cheaply measure a heartbeat: a pulse oximeter.

An ideal technique should be able to perform such measurements in a way which is non-destructive, in-vivo, cheap, accurate and real-time. One technique of the above has great potential to achieve this: light. Light absorption alone, like in a pulse oximeter, is however very limiting, although incredibly cheap. Nilsson et. al. [1] state that the accuracy of absorption measurements may be improved by correct compensation for light scattering. Light scattering does, however, contain far more information than is captured by the scalar quantity absorption. By analogy, if one wishes to know the velocity, it is not sufficient to measure the speed either. Then directional information is lost. The same applies here: if we can measure a full scattered field, then in the usage of a scalar quantity information is bound to get lost.

Hence, in the present research we wish to study light scattering as an in-vivo measurement technique for blood flow. Specifically, light scattering by moving red blood cells. The ultimate goal is to answer the question:

> *"What cardiac parameters may be extracted from the full intensity profile of the light scattered from whole blood?"*

To answer this question, many people have already tried different approaches. For example, Fercher and Briers [2] introduced an experimental technique for laser speckle contrast imaging. Or, He et. al. [3] approach the problem numerically by applying the Finite-Difference Time-Domain method to the shape of a single red blood cell. Then they compare results from the same method applied to two red blood cells with approximating methods, like superposition (which assumes negligible multiscattering) or the discrete dipole approximation.

With that being said, this question is too difficult to answer in a single MSc Thesis project. Instead, we will look at the potential of one specific method of analysis. In this research, we will apply Mie theory to numerically study light scattering. The main advantage of this method is that it is an interferometric method, unlike e.g. Monte-Carlo methods, and that we are able to

take multiscattering into account. Since this code does not yet exist and will thus be created from scratch in this MSc Thesis project, our research question will be:

"Is it possible to use Mie theory for accurately simulating the interferometric scattering by multiple flowing spherical particles?"

In order to answer this question, an Optics code was developed (see Chap. 3), which as a starting point implements Mie theory (see Chap. 2) with a far-field assumption applied to it. The phases have been carefully kept track of in order to obtain an interferometric code. Multiscattering was implemented by means of an iterative method. The developed Optics code was then linked to an existing Computational Fluid Dynamics code. This allowed us to study flowing particles with little additional efforts.

To validate the code, two methods were used. In Chap. 4, static spherical particles were arranged in two lines, and then the resulting diffraction pattern was compared with the double slit experiment. In Chaps. 5 and 6, spherical particles in a cylindrical flow were considered, of which the scattered intensity field gives a speckle pattern. The characteristics of the speckle pattern were compared with what is known about speckles and with experiments.

2

Theory

In this chapter, the relevant theory will be summarised. Some more detailed derivations are delegated to an appendix. In Sec. 2.1, Mie theory will be derived, which computes the light scattered by a spherical scatterer of any size. In Sec. 2.2, the relevant properties of blood are described, followed by the governing equations, which describe the flow: the Navier-Stokes equations. In Sec. 2.3 some relevant exact solutions to the Navier-Stokes equations are given. In Sec. 2.4, Lagrangian Particle Tracking (LPT) is described, together with the possibly relevant forces.

2.1. Mie theory

While studying the colour effects of colloidal gold particles, in 1908 Gustav Mie (1868-1957) derived a solution for diffraction by a sphere [4]. This paper was written in German. Ever since, two English translations have appeared in the literature [5, 6], which wasn't until 1976. In fact, Mie's paper was almost ignored altogether until 1945 [7], likely because his theory was impractical, given the infinite series and Ricatti-Bessel functions involved. His solution had to wait for the digital era and until stable algorithms were developed.

Nowadays, many books on Electrodynamics and Optics derive the Mie solution. In van de Hulst [8], a brief derivation using Gaussian units is given, which is a nice starting point. Stratton [9] contains a more rigorous derivation, as do Bohren & Huffman [10].

In this section, the book of Bohren & Huffman will be followed closely, without referring it repetitively. The most important steps and thoughts in the derivation will be summarised.

2.1.1. Maxwell's Equations and the Wave Equation

Let $\vec{E}$ resp. $\vec{H}$ be the electric and magnetic fields. Then, assuming a periodic field $\vec{A} = \mathrm{Re}\left\{\vec{A}_c e^{-\mathrm{i}\omega t}\right\}$ ($\vec{A}_c \in \mathbb{C}^3$), with $\vec{A} \in \{\vec{E}, \vec{H}\}$, the Maxwell Equations may be written in the form:

$$\nabla \cdot \left(\epsilon \vec{E}_c\right) = 0, \tag{2.1}$$

$$\nabla \times \vec{E}_c = \mathrm{i}\omega\mu\vec{H}_c, \tag{2.2}$$

$$\nabla \cdot \vec{H}_c = 0, \tag{2.3}$$

$$\nabla \times \vec{H}_c = -\mathrm{i}\omega\epsilon\vec{E}_c, \tag{2.4}$$

where ω is the frequency of the periodic solution, μ is the (magnetic) permeability and the complex (electric) permittivity is

$$\epsilon = \epsilon_0 (1 + \chi) + \mathrm{i}\sigma/\omega, \tag{2.5}$$

where σ is the conductivity and χ is the electric susceptibility. All these parameters depend on the propagation medium.

Starting from Maxwell's Equations, it is possible to derive the vector wave equation, which both $\vec{E}$ and $\vec{H}$ must satisfy:

$$\nabla^2 \vec{A}_c + k^2 \vec{A}_c = \vec{0}, \tag{2.6}$$

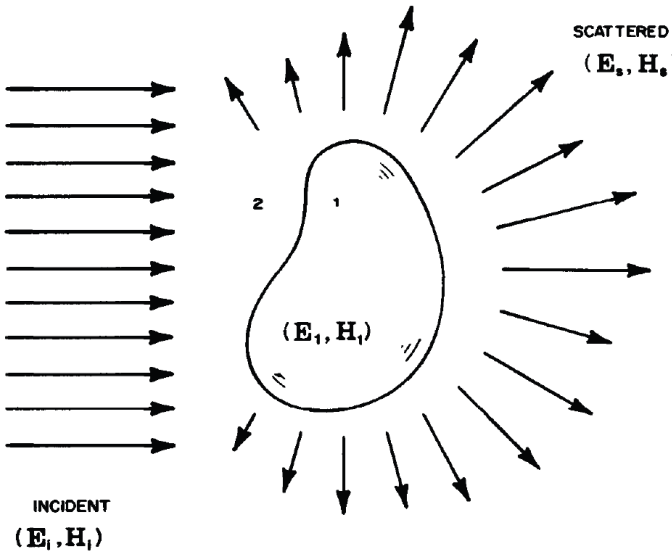


Figure 2.1: Geometry of the scattering problem by an arbitrary particle, cf. Bohren & Huffman Fig. 3.1 (p58) [10]. In Mie theory, a spherical particle will be considered.

where $k^2 = \omega^2\mu\epsilon \to k = \omega(N/c) \equiv Nk_0$ is the wave number, where the (complex) refractive index was introduced, given in terms of the electromagnetic constants/coefficients:

$$N = \sqrt{\frac{\epsilon\mu}{\epsilon_0\mu_0}} = n(1 + \mathrm{i}\kappa) \tag{2.7}$$

where the subscript 0 indicates the value of the parameter in vacuum. Also, $c = 1/\sqrt{\epsilon_0\mu_0}$ is the speed of light in vacuum.

The scalar wave equation is just Eq. (2.6):

$$\nabla^2\psi + k^2\psi = 0. \tag{2.8}$$

As tempting as it is, the individual components of $\vec{A}_c$ do not satisfy the scalar wave equation separately. From (2.8) this is not obvious, since the coupling of the components occur in the Boundary Conditions (BCs), to be considered below.

2.1.2. Derivation of the Scattered Field

Consider the scattering geometry in Fig. 2.1. A Plane Wave (PW) is incident on a particle, which will exert a force on the electrons within the particle. Their movement will both alter the internal and external field. In scattering theory, we consider the external field as a superposition of the incident, $\vec{E}_i$, and scattered field, $\vec{E}_s$:

$$\vec{E}_2 = \vec{E}_i + \vec{E}_s. \tag{2.9}$$

At the particle's boundary, these internal and external fields are coupled via electromagnetic BCs. In Mie theory, a spherical scatterer is observed, which gives a *relatively* easy exact solution.

Boundary Conditions. BCs are contained within the Maxwell Equations and found by integrating them over a small volume or a small loop around a boundary. Integrating over a volume yields a condition for the normal component via Eqs. (2.1 and 2.3). The tangential component follows from Eqs. (2.2 and 2.4). These BCs are respectively:

$$\left[\epsilon_2\vec{E}_2 - \epsilon_1\vec{E}_1\right]\cdot\hat{n}\Big|_{r=a} = 0, \tag{2.10}$$

$$\left[\mu_2\vec{H}_2 - \mu_1\vec{H}_1\right]\cdot\hat{n}\Big|_{r=a} = 0, \tag{2.11}$$

$$\left[\vec{E}_2 - \vec{E}_1\right]\times\hat{n}\Big|_{r=a} = \vec{0}, \tag{2.12}$$

$$\left[\vec{H}_2 - \vec{H}_1\right]\times\hat{n}\Big|_{r=a} = \vec{0}, \tag{2.13}$$

where $\hat{n}$ is the outwards normal of the particle: $\hat{n} = \hat{r}$ for a sphere. Index 1 refers to the internal field and index 2 refers to the external field, cf. Fig. 2.1. a denotes the radius of the spherical particle.

Solve the Scalar Wave Equation. The scalar wave equation (2.8) in spherical coordinates is most easily solved using separation of variables (r, θ, φ). The linearly independent basis set of solutions are:

$$\begin{aligned} \psi_{enm} &= \cos(m\varphi) P_n^m(\cos\theta)\, z_n(kr), \\ \psi_{onm} &= \sin(m\varphi) P_n^m(\cos\theta)\, z_n(kr), \end{aligned} \tag{2.14}$$

where e and o denote 'even' and 'odd' respectively. P_n^m is the associated Legendre polynomial and z_n denotes any spherical Bessel function. As usual when solving a linear differential equation, the general solution is a superposition of all solutions in the basis set of solutions.

Solve the Vector Wave Equation. As mentioned, $\vec{E}$ and $\vec{H}$ do not satisfy the scalar wave equation. However, the solution to the scalar wave equation may be used to construct solutions to the vector wave equation (proof by direct substitution):

$$\begin{aligned} \vec{M} &= \nabla \times (\vec{r}\psi), \\ k\vec{N} &= \nabla \times \vec{M}, \\ k\vec{M} &= \nabla \times \vec{N}, \end{aligned} \tag{2.15}$$

which gives a total of four vector solutions for each $\{n, m\}$. These solutions are called Vector Spherical Harmonics (VSHs). Written out in its entirety, using[1] $\rho \equiv kr$:

$$\begin{aligned} \vec{M}_{\substack{e\\o\end{Bmatrix}mn} = &\begin{Bmatrix} -\sin m\varphi \\ \cos m\varphi \end{Bmatrix} \frac{m}{\sin\theta} P_n^m(\cos\theta)\, z_n(\rho)\, \hat{\theta} \\ &+ \begin{Bmatrix} -\cos m\varphi \\ -\sin m\varphi \end{Bmatrix} \frac{\mathrm{d}P_n^m(\cos\theta)}{\mathrm{d}\theta} z_n(\rho)\, \hat{\varphi}, \end{aligned} \tag{2.16}$$

$$\begin{aligned} \vec{N}_{\substack{e\\o\end{Bmatrix}mn} = &\begin{Bmatrix} \cos m\varphi \\ \sin m\varphi \end{Bmatrix} n(n+1) P_n^m(\cos\theta) \frac{z_n(\rho)}{\rho} \hat{r} \\ &+ \begin{Bmatrix} \cos m\varphi \\ \sin m\varphi \end{Bmatrix} \frac{\mathrm{d}P_n^m(\cos\theta)}{\mathrm{d}\theta} \frac{1}{\rho} \frac{\mathrm{d}(\rho z_n(\rho))}{\mathrm{d}\rho} \hat{\theta} \\ &+ \begin{Bmatrix} -\sin m\varphi \\ \cos m\varphi \end{Bmatrix} \frac{P_n^m(\cos\theta)}{\sin\theta} \frac{m}{\rho} \frac{\mathrm{d}(\rho z_n(\rho))}{\mathrm{d}\rho} \hat{\varphi}. \end{aligned} \tag{2.17}$$

Expand the Incident Field in terms of Vector Spherical Harmonics. The incident field is a PW, which is without loss of generality[2] written as:

$$\vec{E}_i = E_0 e^{\mathrm{i}k_2 z} \hat{x} = E_0 e^{\mathrm{i}k_2 r\cos\theta} \left(\sin\theta\cos\varphi\hat{r} + \cos\theta\cos\varphi\hat{\theta} - \sin\varphi\hat{\varphi}\right), \tag{2.18}$$

which may be expanded in terms of VSHs, which are the basis functions of the studied system:

$$\vec{E}_i = E_0 \sum_{n=1}^{\infty} \mathrm{i}^n \frac{2n+1}{n(n+1)} \left(\vec{M}_{o1n}^{(1)} - \mathrm{i}\vec{N}_{e1n}^{(1)}\right), \tag{2.19}$$

which follows from the equivalent of Fourier's trick: take the functional inner product and by the orthogonality of the VSHs (2.15) many terms drop out. The coefficients may then be computed

[1] Note that ρ depends on k, which depends on the medium via the refractive index: $k = Nk_0$.

[2] A sphere is highly symmetrical, so at this point the coordinate system is arbitrary. However, by choosing the propagation direction as the z-axis and the polarisation of the electric field along the x-axis, we fix the coordinate system. If in some external lab frame the propagation direction and/or the polarisation is different (but still orthogonal), a simple rotation of axis permits usage of the Mie solution as derived with a fixed orientation, i.e., no generality was lost in fixing the coordinate system.

using the non-zero inner products of the non-orthogonal functions. The '(1)' refers to the use of the spherical Bessel function of the first kind, $z_n^{(1)} \equiv j_n$, which is the only finite Bessel function in the origin, which is required since (2.18) is finite in the origin.

Arriving at this solution is the most difficult part of the Mie derivation. To quote Bohren & Huffman [10] (p92): "This is undoubtedly the result of the unwillingness of a plane wave to wear a guise in which it feels uncomfortable; expanding a plane wave in spherical wave functions is somewhat like trying to force a square peg into a round hole.".

Notably in (2.19), only the $m = 1$ term survived, because the incident field (2.18) only possesses the $m = 1$ term. By orthogonality of the cosine and sine function, all terms other than $m = 1$ will then drop out during expansion.

Finally, the magnetic field, $\vec{H}_i$, follows from the curl of (2.19), cf. (2.2):

$$\vec{H}_i = \frac{-k_2}{\omega\mu_2} E_0 \sum_{n=1}^{\infty} \mathrm{i}^n \frac{2n+1}{n\,(n+1)} \left(\vec{M}_{e1n}^{(1)} + \mathrm{i}\vec{N}_{o1n}^{(1)}\right). \tag{2.20}$$

Expand the other Fields in terms of Vector Spherical Harmonics. The next step in the derivation is to expand the other fields in terms of VSHs, as that is the general solution to the vector wave equation (2.6). We may write:

$$\vec{E}_1 = \sum_{n=1}^{\infty} E_n \left(c_n \vec{M}_{o1n}^{(1)} - \mathrm{i}d_n \vec{N}_{e1n}^{(1)}\right), \tag{2.21}$$

$$\vec{H}_1 = \frac{-k_1}{\omega\mu_1} \sum_{n=1}^{\infty} E_n \left(d_n \vec{M}_{e1n}^{(1)} + \mathrm{i}c_n \vec{N}_{o1n}^{(1)}\right), \tag{2.22}$$

where $E_n = E_0 \mathrm{i}^n\,(2n+1)\,/n\,(n+1)$, which is merely for convenience. The spherical Bessel function of the first kind has been used, because of its finiteness at the origin, which is contained within the internal region. Again, the magnetic field followed from the curl.

Similarly, the scattered field may be expanded according to:

$$\vec{E}_s = \sum_{n=1}^{\infty} E_n \left(\mathrm{i}a_n \vec{N}_{e1n}^{(3)} - b_n \vec{M}_{o1n}^{(3)}\right), \tag{2.23}$$

$$\vec{H}_s = \frac{k_2}{\omega\mu_2} \sum_{n=1}^{\infty} E_n \left(\mathrm{i}b_n \vec{N}_{o1n}^{(3)} + a_n \vec{M}_{e1n}^{(3)}\right), \tag{2.24}$$

where the spherical Bessel function of the third kind ≡ spherical Hankel function of the first kind, $z_n^{(3)} \equiv h_n^{(1)}$ has been used. This follows from its asymptotic behaviour. It becomes an outgoing spherical wave for $kr \gg n^2$:

$$h_n^{(1)}(kr) \sim \frac{(-\mathrm{i})^n e^{\mathrm{i}kr}}{\mathrm{i}kr}, \tag{2.25}$$

which is to be expected on physical grounds for a scattering process.

The reader may wonder why $m \neq 1$ is again omitted. And why are the $\vec{N}_{omn}$ and $\vec{M}_{emn}$ terms omitted for $\vec{E}$? This follows from the boundary conditions, in combination with the orthogonality of the VSHs and the fact that the incident field (2.19) does not possess those terms. Since I'd not just believe that either, a more detailed description may be found in App. A.1.

Find the Mie Coefficients. The Mie coefficients, $\{a_n, b_n, c_n, d_n\}$, are finally found by applying the BCs (2.10-2.13) to the field expansions (2.19-2.24). The derivation may be found in App. A.2. Four equations with four unknowns are then found, from which all four coefficients follow. In this

research, only the scattering coefficients are of interest:

$$a_n = \frac{\mu_2 m^2 j_n(y)[x j_n(x)]' - \mu_1 j_n(x)[y j_n(y)]'}{\mu_2 m^2 j_n(y)[x h_n^{(1)}(x)]' - \mu_1 h_n^{(1)}(x)[y j_n(y)]'}, \tag{2.26}$$

$$b_n = \frac{\mu_1 j_n(y)[x j_n(x)]' - \mu_2 j_n(x)[y j_n(y)]'}{\mu_1 j_n(y)[x h_n^{(1)}(x)]' - \mu_2 h_n^{(1)}(x)[y j_n(y)]'}, \tag{2.27}$$

where $x \equiv k_2 a$ is the size parameter, $y \equiv k_1 a = mx$, and m is the *relative* refractive index:

$$m \equiv \frac{k_1}{k_2} = \frac{N_1}{N_2} = \sqrt{\frac{\epsilon_1 \mu_1}{\epsilon_2 \mu_2}}. \tag{2.28}$$

It is now convenient to introduce the Ricatti-Bessel functions, which will simplify the Mie coefficients:

$$\psi_n(\rho) \equiv \rho j_n(\rho), \tag{2.29}$$

$$\xi_n(\rho) \equiv \rho h_n^{(1)}(\rho), \tag{2.30}$$

from which the Mie coeffients simplify to:

$$a_n = \frac{\mu_2 m \psi_n(y)\psi_n'(x) - \mu_1 \psi_n(x)\psi_n'(y)}{\mu_2 m \psi_n(y)\xi_n'(x) - \mu_1 \xi_n(x)\psi_n'(y)}, \tag{2.31}$$

$$b_n = \frac{\mu_1 \psi_n(y)\psi_n'(x) - \mu_2 m \psi_n(x)\psi_n'(y)}{\mu_1 \psi_n(y)\xi_n'(x) - \mu_2 m \xi_n(x)\psi_n'(y)}. \tag{2.32}$$

In the present research, $\mu_1 = \mu_2 = \mu_0 \equiv \mu$, and thus μ will drop out of the coefficients.

2.1.3. Amplitude Scattering Matrix

Substituting the definition of the VSHs (2.16 and 2.17) into the field expansion (2.23), yields:

$$\vec{E}_s \cdot \hat{r} = \frac{\cos\varphi}{\rho^2} \sum_{n=1}^{\infty} E_n i a_n n(n+1) \sin\theta \pi_n(\cos\theta)\xi_n(\rho), \tag{2.33}$$

$$\vec{E}_s \cdot \hat{\theta} = \frac{\cos\varphi}{\rho} \sum_{n=1}^{\infty} E_n \left(i a_n \tau_n(\cos\theta)\xi_n'(\rho) - b_n \pi_n(\cos\theta)\xi_n(\rho) \right), \tag{2.34}$$

$$\vec{E}_s \cdot \hat{\varphi} = \frac{-\sin\varphi}{\rho} \sum_{n=1}^{\infty} E_n \left(i a_n \pi_n(\cos\theta)\xi_n'(\rho) - b_n \tau_n(\cos\theta)\xi_n(\rho) \right), \tag{2.35}$$

where two angular functions were introduced to replace the associated Legendre polynomials:

$$\tau_n(\cos\theta) \equiv \frac{\mathrm{d}P_n^1(\cos\theta)}{\mathrm{d}\theta}, \qquad \pi_n(\cos\theta) \equiv \frac{P_n^1(\cos\theta)}{\sin\theta}. \tag{2.36}$$

As it turns out, these two angular functions possess nice recursive properties, useful for numerical algorithms [10, 11].

If we now apply the Far-Field (FF) approximation, ξ_n may be replaced by its asymptotic limit, cf. Eqs. (2.25 and 2.30). It then follows that the radial component of $\vec{E}_s$ falls off $1/\rho$ faster than the other components, making it negligible in the FF. We are left with (using $\mu \equiv \cos\theta$):

$$\vec{E}_s \cdot \hat{\theta} \sim \cos\varphi \frac{e^{i\rho}}{-i\rho} E_0 \sum_{n=1}^{\infty} \frac{2n+1}{n(n+1)} \Big(a_n \tau_n(\mu) + b_n \pi_n(\mu) \Big), \tag{2.37}$$

$$\vec{E}_s \cdot \hat{\varphi} \sim -\sin\varphi \frac{e^{i\rho}}{-i\rho} E_0 \sum_{n=1}^{\infty} \frac{2n+1}{n(n+1)} \Big(a_n \pi_n(\mu) + b_n \tau_n(\mu) \Big). \tag{2.38}$$

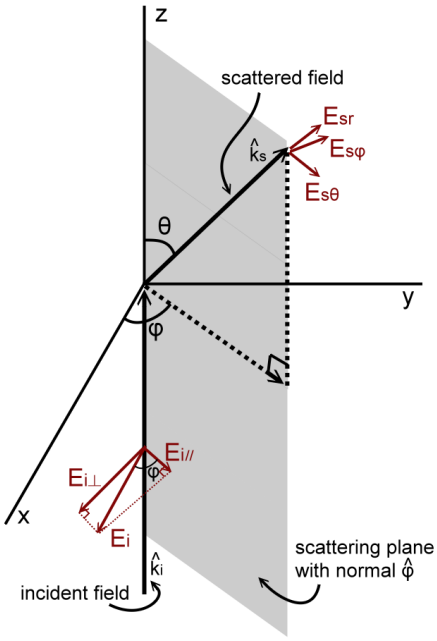


Figure 2.2: The standard scattering geometry is shown. An incoming PW is coming in from below with $\hat{k}_i = \hat{z}$ and is scattered radially outward from the origin $\hat{k}_s = \hat{r}$, which is the position of a scatterer.
The scattering plane is per definition the area spanned by these two propagation vectors, which is shown as a gray shaded area in the figure. Its unit normal is $\hat{\varphi}$.
Any incident field (here: polarised in the $\hat{x}$-direction) may be resolved into a parallel ($E_\parallel$) and perpendicular ($E_\perp$) component w.r.t. the scattering plane. The same applies to the scattered field. For an arbitrary coordinate system, the fields may then be expressed in terms of those parallel and perpendicular components.

Upon introduction of the 'Amplitude Scattering Matrix', $[S](\mu)$, this may be written in the form:

$$\begin{bmatrix} \vec{E}_s \cdot \hat{\theta} \\ \vec{E}_s \cdot \hat{\varphi} \end{bmatrix} = \frac{e^{i\rho}}{-i\rho} E_0 \begin{bmatrix} S_2 & 0 \\ 0 & S_1 \end{bmatrix} \begin{bmatrix} \cos\varphi \\ -\sin\varphi \end{bmatrix}, \tag{2.39}$$

where the components of the amplitude scattering matrix are given by

$$\begin{aligned} S_2(\mu) &\equiv \sum_{n=1}^{\infty} \frac{2n+1}{n(n+1)} \Big(a_n \tau_n(\mu) + b_n \pi_n(\mu) \Big), \\ S_1(\mu) &\equiv \sum_{n=1}^{\infty} \frac{2n+1}{n(n+1)} \Big(a_n \pi_n(\mu) + b_n \tau_n(\mu) \Big). \end{aligned} \tag{2.40}$$

Finally, recall that the incident field was chosen to travel in the $\hat{z}$-direction and polarised in the $\hat{x}$-direction. With this convention, $\theta = \theta_s$ is the scattering angle: the angle between the propagation vector of the incident field ($\vec{k}_i = k\hat{z}$) and the propagation vector of the scattered field ($\vec{k}_s = k\hat{r}$). The scattering plane, which is per definition the plane spanned by those two propagation vectors, is a vertical plane rotated with an angle φ w.r.t. the $\hat{x}$-axis. This is illustrated by Fig. 2.2.

Noting that a rotation of axis preserves orthogonality, it follows that for an arbitrary coordinate system we can use the component parallel and perpendicular to the scattering plane. In cylindrical coordinates (s, φ, z), the incident field may be written as

$$\vec{E}_i = E_0 e^{ikz} \hat{x} = E_0 e^{ikz} \left(\cos\varphi \hat{s} - \sin\varphi \hat{\varphi} \right), \tag{2.41}$$

where the $\hat{\varphi}$-component is orthogonal to the scattering plane and $\hat{s}$ is parallel to it (see Fig. 2.2).

Comparing this result for the incident field with Eq. (2.39), and noting that the $\hat{\varphi}$-component is orthogonal to the scattering plane and that the $\hat{\theta}$-component is parallel to it, we may write:

$$\begin{bmatrix} E_{s\parallel} \\ E_{s\perp} \end{bmatrix} = \frac{e^{ik(r-z)}}{-ikr} \begin{bmatrix} S_2 & 0 \\ 0 & S_1 \end{bmatrix} \begin{bmatrix} E_{i\parallel} \\ E_{i\perp} \end{bmatrix}, \tag{2.42}$$

where e^{-ikz} was merely introduced to cancel the exponential in (2.41), i.e., the left hand side is solely a function of r (and θ via $[S]$) and not of z, which is confusing in Mie theory literature.

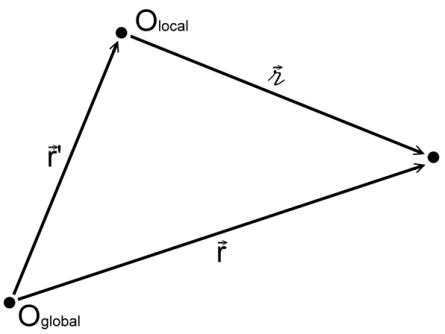


Figure 2.3: $\vec{\imath}$ is used to define a difference in position, which permits the local coordinates (which used $\vec{r}$ in previous sections) to be written in global coordinates (which now claims the symbol $\vec{r}$).

2.1.4. Multiple Scatterers

When multiple scatterers are considered, it is required to convert Mie theory to some global coordinate system, i.e., currently the origin was set in the center of the scatterer, which is ambiguous with multiple scatterers. Also, the different scatterers will very likely have a different initial phase, which must be considered.

Global Coordinates. In order to define Mie theory w.r.t. some global coordinate system, it is important to express everything relative to some common origin. In Fig. 2.3, we define

$$\vec{\imath} \equiv \vec{r} - \vec{r}', \tag{2.43}$$

where $\vec{r}$ is the position w.r.t. the global origin and $\vec{r}'$ is the position of the local origin w.r.t. the global origin. In Mie theory, $\vec{r}'$ is the position of the scatterer w.r.t. the global origin.

Using this symbol, it is possible to replace all $\vec{r}$ derived in local coordinates by $\vec{\imath}$ and the solution in global coordinates has been found. E.g., (2.42) becomes:

$$\begin{bmatrix} E_{s\parallel} \\ E_{s\perp} \end{bmatrix} = \frac{e^{\mathrm{i}k(\imath - z)}}{-\mathrm{i}k\imath} \begin{bmatrix} S_2 & 0 \\ 0 & S_1 \end{bmatrix} \begin{bmatrix} E_{i\parallel} \\ E_{i\perp} \end{bmatrix}. \tag{2.44}$$

Initial Phase. Since Mie theory was derived in a local coordinate system, the reference to any sort of initial phase is lost. However, it is still implicitly contained within the solution, i.e., the factor $e^{-\mathrm{i}\omega t}$ is omitted in the complex notation of the electric field, but the combination of $e^{-\mathrm{i}\omega t}$ and $e^{\mathrm{i}kz}$ determines a spacetime origin for the incoming PW, which cannot be the same for all scatterers.

Fig. 2.4 shows a test geometry which has an identical distance between each particle and the scattering target, such that only the initial phase may influence the relative phase of the two scatterers. Since we wish to measure the scattered field of the two scatterers simultaneously, it is required to sample the incoming PW at a different time (t_{-1} vs. t_0) at the same z or at a different z at the same time (t_0). So, with respect to some common global origin where $z = 0$, we may write for the field at the scatterer's local origin (using $\vec{k}_i = k\hat{z}$):

$$\vec{E}(\vec{r}_1, t) = \vec{E}(\vec{0}, t)e^{\mathrm{i}\vec{k}_i \cdot \vec{r}_1} = \vec{E}(\vec{0}, t)e^{\mathrm{i}kz_1}, \tag{2.45}$$

$$\vec{E}(\vec{r}_2, t) = \vec{E}(\vec{0}, t)e^{\mathrm{i}\vec{k}_i \cdot \vec{r}_2} = \vec{E}(\vec{0}, t)e^{\mathrm{i}kz_2}, \tag{2.46}$$

and since $\vec{E}(\vec{0}, t)$ may be unambiguously defined, the relative phase difference immediately follows. So, the incident field of particle j may be given by:

$$\vec{E}_j = E_0 e^{\mathrm{i}(kz_{\mathrm{local}} - \omega t)} e^{\mathrm{i}kz_j}\hat{x}, \tag{2.47}$$

where the first exponential is the exponential used in (local) Mie theory, whilst the second is a phase difference picked up due to the global coordinates, which must be taken into account explicitely, separately from the Mie solution.

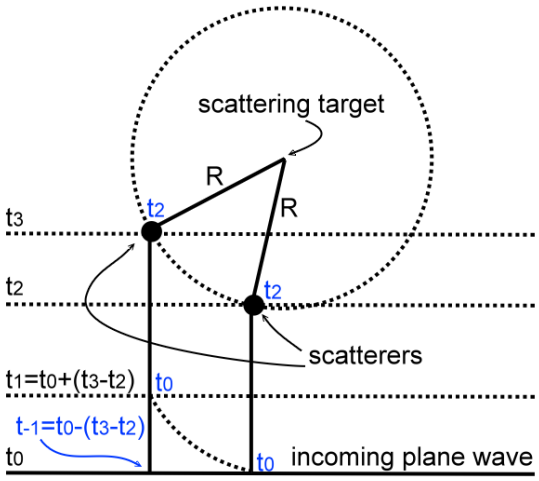


Figure 2.4: This figure helps in understanding the initial phase within Mie theory. The scattering target (e.g., a camera) is at a fixed identical distance from the two scatterers, such that the final phase need not be considered.
The (black) times on the very left show the times from the perspective of the incoming PW. As expected, the wave does not arrive at the two scatterers at the same time.
The (blue) times shown at the 'events' are reverse-engineered times: If we are to measure at the scattering target at a fixed time, we cannot consider the incoming PW at the same time *and* at the same position simultaneously.

2.2. Blood

For a human to survive, his organs need to be given useful energy. Blood is the means for the human body to transfer all kinds of proteins and oxygen to the organs. Via white blood cells, the immune system too uses the blood to fight infections.

Blood consists of the yellow liquid blood plasma in which particles are dispersed: Red Blood Cells (RBCs) (erythrocytes), white blood cells (leukocytes) and platelets (thrombocytes). The hematocrit (volume fraction of RBCs) is about 45% (for men) [12], blood plasma occupies about 54.3% and white blood cells about 0.7%. Blood plasma consists of 92% water and 8% dissolved proteins, e.g. glucose. At 37°C, the resulting blood density is 1060 kg/m^3 [13, 14].

For the present research, primarily the RBCs are of interest. These donut-shaped particles (see Fig. 2.5) are responsible for transporting oxygen through the body, which they acquire from the lungs.

Figure 2.5: This figure shows the shape of a RBC. It is a flexible oval biconcave disk, a donut-shape if you will. Its flexibility implies that it is able to change its shape as to be able to squeeze through smaller arteries.

The high volume fraction of particles causes blood to behave like a non-Newtonian fluid: it has a different rheology than simple water. In line with the theory of evolution, this adapted rheology increases the transport compared with what pure blood plasma would be able to do. The (dynamic) viscosity of blood is about $\mu_{\text{blood}} = 3$ to $4\ \text{mPa}\cdot\text{s}$ at 37°C. This is about a factor 4 more viscous than water ($\mu_{\text{water}} = 0.890\text{mPa}\cdot\text{s}$ at 25°C) [15].

In this flow, there is a radial distribution for the RBC-concentration, as is measured by Aarts et. al. [16] and shown in Fig. 2.6. This distribution may be converted into a radial distribution for the particle number density, which may serve as a probability density function when injecting particles in a flow simulation.

2.2.1. Converting a volume distribution to a number distribution

If ϕ denotes the hematocrit, then we may write that in an infinitesimal thin cylinder surface (between r and $r + \mathrm{d}r$):

$$\phi(r) = \frac{\mathrm{d}V_{\text{rbc}}(r)}{\mathrm{d}V(r)},$$
$$\mathrm{d}V = 2\pi r\,\mathrm{d}r L,$$
$$\mathrm{d}N(r) = \frac{\mathrm{d}V_{\text{rbc}}}{V_{1\ \text{rbc}}},$$

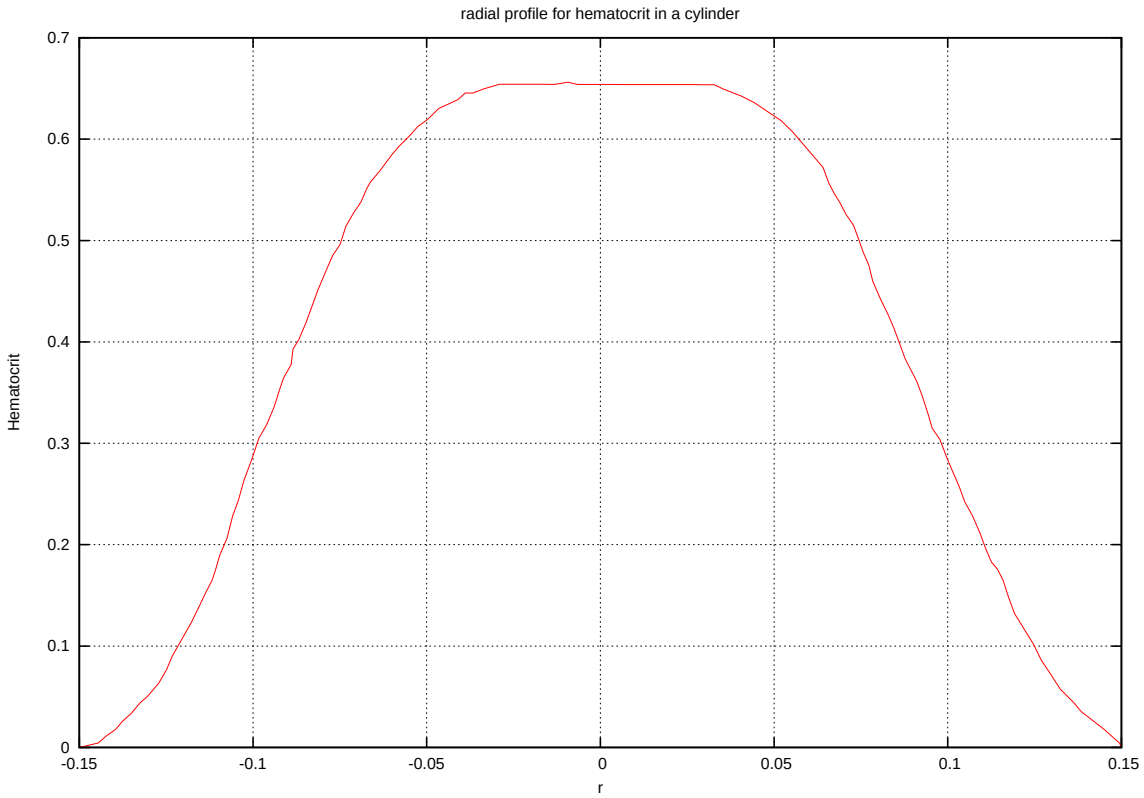


Figure 2.6: Radial hematocrit profile in a cylinder as measured by Aarts et. al. [16].

where V denotes a volume, $\mathrm{d}N$ is the number of particles within the volume $\mathrm{d}V$ and L is the length of the cylinder. And thus:

$$\frac{\mathrm{d}N(r)}{\mathrm{d}r} = [\phi(r)r]\,\frac{2\pi L}{V_{1\,\mathrm{rbc}}} \equiv NP(r), \tag{2.48}$$

where $P(r)$ is the probability density function for the particle number and N is the total number of particles within a cylinder of length L and radius R:

$$P(r) = \frac{\phi(r)r}{\int_0^R \phi(r)r\mathrm{d}r}, \tag{2.49}$$

$$N = \frac{2\pi L}{V_{1\,\mathrm{rbc}}} \int_0^R \phi(r)r\mathrm{d}r. \tag{2.50}$$

Here, N is required to obtain the correct particle concentration, whereas $P(r)$ is required to obtain the correct radial particle distribution. For the present research, N is strongly limited by the computational resources and thus the required N cannot be achieved.

The cumulative particle distribution is then given by:

$$P(r \leq R_1) = \int_0^{R_1} P(r)\mathrm{d}r. \tag{2.51}$$

which is useful for numerically converting a uniform distribution (as generated by the computer) to an arbitrary distribution, i.e., if $u \in U[0,1]$, then we can find the r_{inj} at which to inject the particle by inverting the equation $P(r_{\mathrm{inj}} \leq R_1) = u$. Fig. 2.7 illustrates this graphically.

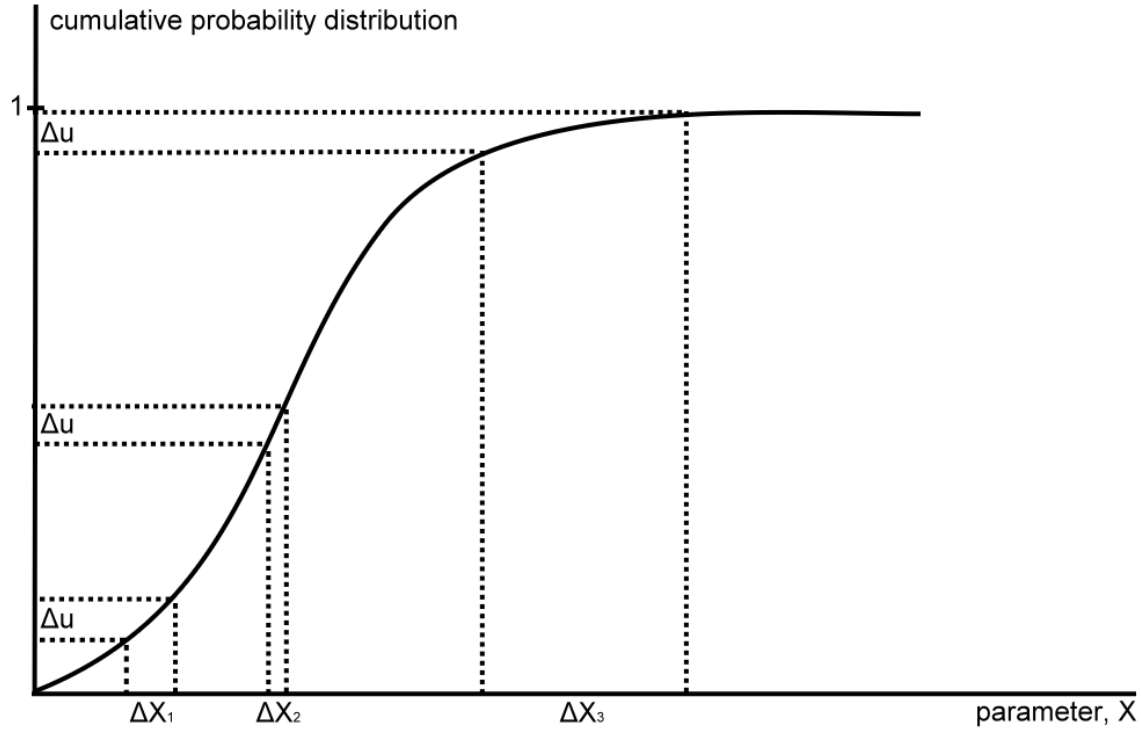


Figure 2.7: This figure shows a cumulative probability function, $P(x \leq X)$, for some parameter, x. The derivative of this function is the probability density function, i.e., the slope of this function is a measure for how probable x is: the steeper the function at some x_1, the more likely that x_1 is chosen.
A random number $u \in U[0,1]$ can then be mapped to the x-axis. It is shown that for some constant range Δu, the range Δx varies. Since all numbers in Δu are equally likely, this means that values for x are not equally likely.
In other words, $f(x) \equiv P(x \leq X)$ and since $f(x)$ is uniformly increasing, $P(f(x) \leq f(X)) = f(x)$. But if we take a random number $u \in U[0,1]$, then $P(f(x) \leq u) = u$. It then follows that $P(x \leq X) = u$ for $X = f^{-1}(u)$, i.e., $x = f^{-1}(u)$ should be taken, when sampling for x using uniform numbers, to satisfy the required probability density function.

2.2.2. The Navier-Stokes Equations and Rheology

The equations of fluid dynamics to be solved are the Navier-Stokes equations for *incompressible* flow, using the Newtonian model for the stress tensor [17]:

$$\frac{\partial u_i}{\partial x_i} = 0, \tag{2.52}$$

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = \frac{1}{\rho}\left(-\frac{\partial P}{\partial x_i} + \frac{\partial \sigma_{ji}}{\partial x_j} + f_i^{\text{ext}}\right), \tag{2.53}$$

$$\sigma_{ji} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \tag{2.54}$$

where u_i is the velocity in the x_i direction, x_i denotes the i'th coordinate: $x_i \in \{x, y, z\}$ for $i \in \{1, 2, 3\}$, ρ is the fluid density, μ the fluid viscosity, P is the pressure, σ_{ij} is the deviatoric stress tensor and f_i^{ext} represents the sum of all external forces per unit volume (e.g., gravity). The Einstein summation convention is employed.

Blood does, however, as mentioned, not behave like a Newtonian fluid. Consequently, (2.54) should be replaced by a different constitutive relationship when a non-Newtonian fluid model is assumed (a different rheology). For the scope of the present research, a Newtonian model will be assumed. For future research, non-Newtonian models for blood can be found in e.g. Merrill [18], who shows that blood behaves like a Newtonian fluid for high strain rate $\dot{\gamma} = \frac{\partial u_z}{\partial r} > 100\text{s}^{-1}$ (cylindrical coordinates), but has an offset in its shear stress (the yield stress) for small strain rates with a different accompanying viscosity.

2.3. Exact Solution to the Navier-Stokes Equations

Whilst the Navier-Stokes Equations (2.52-2.54) are non-linear and there exists no general solution (as of today), it may still be solved exact for some cases, typically for a well-defined simple geometry and a sufficiently low Reynolds number.

The Reynolds number is the ratio of inertial forces and viscous forces and thus provides a measure for whether the flow is *inertia-dominated* (which will eventually result in turbulence at a sufficiently high geometry-dependent Reynolds number, e.g. about $Re > 4000$ for pipe flow) or *viscous-dominated* (which results in laminar flow for a sufficiently low Reynolds number):

$$Re \equiv \frac{UD}{\nu} = \frac{\bar{u}2R}{\nu}, \tag{2.55}$$

where U is a typical velocity scale, D a typical length scale and ν is the kinematic viscosity of the fluid at hand. The second expression is in terms of parameters of pipe-flow, R is the radius of the pipe and $\bar{u}$ is the mean velocity:

$$\bar{u} \equiv \frac{\int u(r)da}{\int da} = \frac{\int_0^R u(r)rdr}{\int_0^R rdr}. \tag{2.56}$$

2.3.1. Hagen-Poiseuille

For the case of a cylinder of radius R, there exists an exact solution for the steady-state ($\frac{\mathrm{d}\Psi}{\mathrm{d}t} = 0$) fully-developped ($\vec{u} = u(r)\hat{z}$, $P = P(z)$) profile. The result is the following:

$$\vec{u}(r) = \left(-\frac{\mathrm{d}P}{\mathrm{d}z}\right)\frac{1}{4\mu}\left(R^2 - r^2\right)\hat{z}. \tag{2.57}$$

Applying Eq. (2.56) to Poiseuille flow allows us to express the mean velocity, $\bar{u}$, in terms of the maximum velocity, $u_{max} = u(0)$. The result is that $u_{max} = 2\bar{u}$.

2.3.2. Womersley

The Womersley exact solution is similar to the Hagen-Poiseuille solution, but with the $\frac{\mathrm{d}}{\mathrm{d}t}$-term included. For an arbitrary time-dependency of the pressure gradient, the solution $u(r,t)$ takes the form of an inverse temporal Fourier transform of the Fourier transform of $\frac{\partial P}{\partial z}$ multiplied by Bessel functions of the first kind and of zeroth order, J_0. For a cosine forcing of frequency ω_0, $\frac{\partial P}{\partial z} = \left|\frac{\partial P}{\partial z}\right|_{max}\cos\omega_0 t$, this inverse Fourier transform may be carried out, resulting in:

$$\begin{aligned} u(r,t) &= -\left|\frac{\partial P}{\partial z}\right|_{max}\frac{1}{\rho\omega_0}\left[A(r)\sin\omega_0 t + B(r)\cos\omega_0 t\right], \\ A(r) &\equiv \Re\{y(r/R)\}, \\ B(r) &\equiv \Im\{y(r/R)\}, \\ y(x) &= 1 - \frac{J_0(\alpha_0 \mathrm{i}^{3/2}x)}{J_0(\alpha_0 \mathrm{i}^{3/2})}, \end{aligned} \tag{2.58}$$

where $\alpha_0 \equiv \sqrt{\omega_0/\nu}R$ is the Womersley dimensionless number and ρ is the mass density of the fluid. This solution could be used to evolve Lagrangian particles in a time-dependent cylindrical fluid flow, without requiring a Computational Fluid Dynamics (CFD) code.

2.4. Lagrangian Particle Tracking (LPT)

To simulate particles with a subgrid size, LPT is used. Particles are point-particles subject to forces and are evolved using Newton's law and some integration scheme. Defining $\vec{Y}(t)$ as the particle position, $\vec{V}(t) = \frac{\mathrm{d}\vec{Y}(t)}{\mathrm{d}t}$ as the particle velocity and $\vec{u}(\vec{x},t)$ as the fluid velocity, and letting the subscript p refer to the particle and f to the fluid, the forces can then be written as [19, 20]:

- **Particle Inertia Force.** (left-hand side of Newton's equation) $\vec{F}_{in} = m_p \frac{\mathrm{d}\vec{V}}{\mathrm{d}t}$
- **Gravity and Buoyancy Force.** $\vec{F}_g = \left(m_p - m_f\right)\vec{g}$
- **Viscous Drag Force (Stokes).** $\vec{F}_d = -6\pi\mu R_p \left(\vec{V}(t) - \vec{u}(\vec{Y}(t), t)\right)$
- **Pressure Force.** $\vec{F}_{pr} = m_f \left(\frac{\mathrm{D}\vec{u}}{\mathrm{D}t} - \nu\nabla^2\vec{u}\right)\Big|_{\vec{Y}(t)}$
- **Added-mass Force.** $\vec{F}_{am} = -\frac{m_f}{2}\frac{\mathrm{d}(\vec{V}(t) - \vec{u}(\vec{Y}(t),t))}{\mathrm{d}t}$
- **Basset/History Force.** $\vec{F}_B = \frac{R_p}{\sqrt{\pi\nu}}\int_{-\infty}^{t} \frac{\mathrm{d}(\vec{V}(\xi) - \vec{u}(\vec{Y}(\xi),\xi))}{\mathrm{d}\xi}\frac{\mathrm{d}\xi}{\sqrt{t-\xi}}$

where $\frac{\mathrm{d}\Psi}{\mathrm{d}t} = \frac{\partial\Psi}{\partial t} + \vec{V}\cdot\nabla\Psi$ and $\frac{\mathrm{D}\Psi}{\mathrm{D}t} = \frac{\partial\Psi}{\partial t} + \vec{u}\cdot\nabla\Psi$ for any Ψ. Which results in the equation for particle motion:

$$\vec{F}_{in} = \vec{F}_{pr} + \vec{F}_{am} + \vec{F}_d + \vec{F}_g + \vec{F}_B. \tag{2.59}$$

3

Description of the algorithm/code

The purpose of the created codes is to study light scattering from an ensemble of scatterers: Red Blood Cells (RBCs). So, given an incoming Plane Wave (PW) and given a certain configuration of scatterers, the Optics code should compute the intensity profile as measured by a given distant camera. This computed intensity will be independent of time, unlike the instantaneous intensity, as will be discussed in Sec. 3.1.2.

In-vivo, these scatterers are moving as a function of time in the arteries, which implies that the above-mentioned configuration is not static. So, the Optics code must be executed repetitively for each timestep. Evolving the particle positions will be performed by the Fluids code. Since this process is one-way coupled, the Fluids and Optics codes will be two separate codes, rather than one monolithic code[1]. Provided that scripts handle the linking process, this provides a lot more convenience and flexibility: it permits virtually any existing CFD code to be used.

This chapter briefly describes the assumptions of and the functionality of the Optics code (Sec. 3.1), the Fluids code (Sec. 3.2) and the linking of the codes (Sec. 3.3). A detailed description of the codes is delegated to App. B.

Evidently, some approximations are to be made. Arguably the strongest two are that the interparticle distance is assumed to be in the Far-Field (FF) and that the scatterers (RBCs) may be approximated as spherical particles. It is difficult to determine how accurate these approximations are in a general way.

3.1. Optics

The Optics code is responsible for computing the intensity profile at a given camera for a given incoming Plane Wave (PW) and a given distribution of spherical scatterers. The computation will be performed in an interferometric manner, i.e., the phase will carefully be taken care of for each individual trajectory.

Currently, two codes have been created: Single-Scattering Far-Field (SSFF), which only takes single scattering into account, and Multi-Scattering Far-Field (MSFF), which takes multiscattering into account without any statistical averaging, which is an extention to the SSFF code. In either case, a FF assumption is made for the distance between the scatterers.

3.1.1. The Mie Algorithm

As a starting point, the Bohren & Huffman Mie code ('bhmie') has been used, which accompanies their book [10]. Their code has been written in F77 (Fortran). Their code computes the amplitude scattering matrix as a function of the scattering angle, $[S](\theta_S)$, for a single homogeneous sphere. The scattering angles necessarily are a sequence of evenly spaced scattering angles between 0° and 180°. Additionally, it does as well output some θ_S-averaged quantities, like the extinction efficiency.

[1] A monolithic code is per definition a single code which performs all tasks. This design principle disencourages resuability and flexibility and should be avoided if there is no reason to design a code that way. Since the Fluids code is independent of the Optics code, there is no reason for the combined physics to consist of a single monolithic code, because it is possible for them to be two separate codes without any additional effort.

The only input parameters required by bhmie are the size parameter, x, and the relative refractive index, m (see Eq. (2.28) and the text just above it).

This code has been adapted to take as an argument an array of arbitrary scattering angles, or rather, the cosine of the scattering angles, since the scattering angle only appears as a cosine in the entire algorithm. Then, any computed quantities, which are not required for the present study (like the extinction efficiency), have been removed from the code.

In bhmie, $[S]$ is computed cf. (2.40), which is an infinite sum. This sum is truncated at the index n_C, which has been found empirically:

$$n_C = \text{round}\left(x + 4x^{\frac{1}{3}} + 2\right), \tag{3.1}$$

where x is the size parameter of the sphere. This heuristic was proposed by Wiscombe in 1979 [21, 22] and is merely a given wisdom in Bohren & Huffman's book. The Bessel/Hankel functions and Legendre Polynomials are computed using recursion relationships, given the exact (or asymptotic) value for some index. The book of Barber & Hill nicely summarises all these, and more, computational techniques for easy reference [11].

3.1.2. The Extended Algorithm (Camera, Multiscattering)

Fig. 3.1 shows qualitatively what the algorithm is expected to do. Briefly, an incoming PW is scattered by each particle to each other particle. In the very-FF these scattered fields are again PWs, and thus the exact same equations may be used to scatter those fields again by each particle to each other particle. Ad infinitum. I will refer to these iterations as "scattering orders".

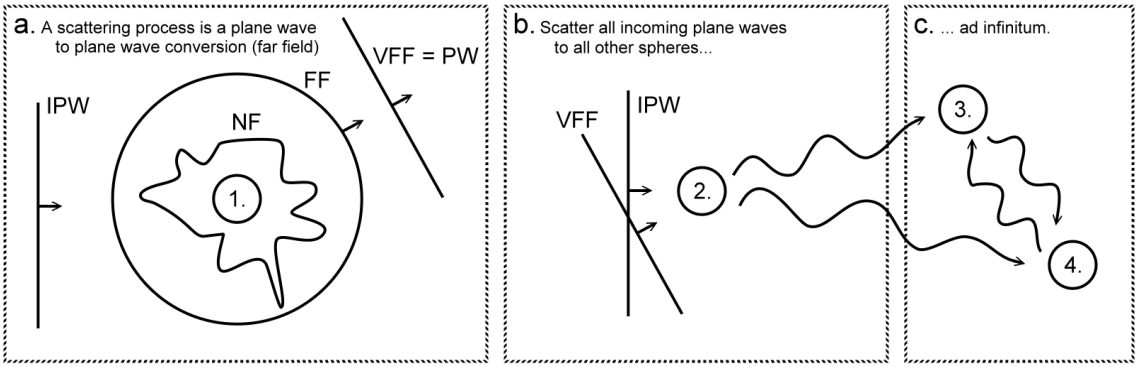


Figure 3.1: The figure shows the workflow of the algorithm in an animated manner. In (a), there is an incoming Plane Wave (PW) (IPW) incident on the spherical scatterer labelled '1', which will scatter. In the Near-Field (NF), this gives a very complex scattered field as described by Mie theory (2.40). In the Far-Field (FF), this may be approximated by a spherical wave (which is still a function of the scattering angle), which in the Very Far-Field (VFF) may be approximated as a PW.
In (b), a second scatterer is illuminated by both the scattered PW and the original incoming PW, which will again result in a PW in the VFF. In this manner, each scatterer can scatter each wave to each other scatterer, until convergence.
In (c), it is noted that this scattering process may continue ad infinitum when it is performed in the described iterative manner.

Fig. 3.2 illustrates what the scattering order means. It also discriminates between three distinct regions in the multiscattering-process, which are to be treated differently. In initialscatter, the incoming PW is involved, which can be interpreted as the start-up of the multiscattering part of the code. In multiscatter, the scattered field from each particle, l, is used as an incoming PW for each particle, i, and is to be scattered to each particle, j, where $l \neq i \cap i \neq j$. Note that the figure does not show the "from each particle, l, to each particle, j, via each other particle, i"-process, but rather shows exactly one l, one i and one j (Fig. 3.3 will help with that, as will be described below). Lastly, in scatter2cam, the scattered fields from each particle of all scattering orders are scattered towards *every pixel* of the camera, accumulated, and the intensity is computed. Finally, add some I/O around these three routines and the entire Optics code has been described.

From a mathematical point of view, the algorithm is structured as follows. Using that fields add up arithmetically and using the Mie algorithm with the FF approximation applied (cf. Eq. (2.44)),

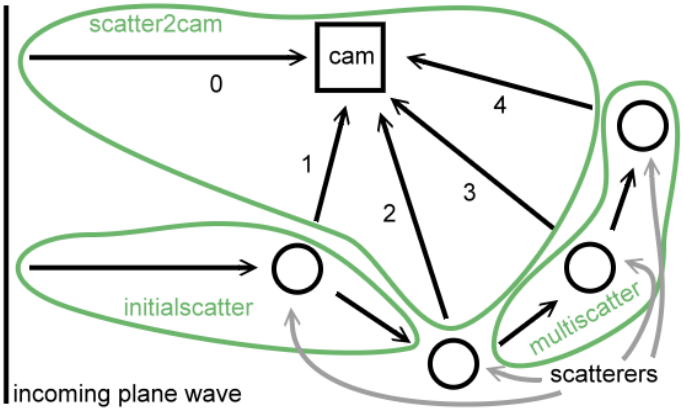


Figure 3.2: This figure illustrates what the scattering order, p, means. Zeroth order scattering is the direct arrival of the incoming PW on the camera. Any order above that is the number of scatterers the field has 'seen' before reaching the camera.
The figure also identifies three distinct regions in the multiscattering-process which each require a slightly different algorithm, as described in the text.
Evidently, every scattering order has many more contributions than is shown in the present figure.

which gives the scattered field given an incoming PW, we have:

$$\vec{E}_j^p = \sum_{i \neq j} \vec{E}_{ji}^p, \tag{3.2}$$

$$\vec{E}_{ji}^p = \sum_{l \neq i} \vec{E}_{jil}^p, \tag{3.3}$$

$$\vec{E}_{jil}^p = [S]_{jil} \vec{E}_{il}^{p-1} \frac{\exp \mathrm{i}\Delta\varphi_{ji}}{-\mathrm{i}k r_{ji}}, \tag{3.4}$$

where $\vec{E}$ is the (electric) field, p denotes the scattering order, $\{l, i, j\}$ are indices which denote the three scatterers involved, $\Delta\varphi_{ji} = k r_{ji}$ is the phase picked up from moving from particle i to j, r_{ji} is the distance between particles i and j. The sums over l and i are in words: "for each incoming plane wave (originating from particle l), via each other particle, i, scatter to particle j". $[S]$ is the amplitude scattering matrix, which is the output of the bhmie algorithm of Sec. 3.1.1 with as input the scattering angle. $[S]$ thus depends on all three indices, because it takes three points to span an angle. Combining the equations and rearranging terms, it follows that:

$$\vec{E}_j^p = \sum_{i \neq j} \frac{\exp \mathrm{i}\Delta\varphi_{ji}}{-\mathrm{i}k r_{ji}} \sum_{l \neq i} [S]_{jil} \vec{E}_{il}^{p-1}, \tag{3.5}$$

where it is noted that the spherical wave term could be taken out of the sum. This equation is to be interpreted as an iterative equation which takes us up one scattering order.

Eventually, every scattering order needs to be scattered to every pixel, c, of the camera, as depicted in Fig. 3.2. Since fields add up arithmetically, this becomes:

$$\vec{E}_c = \sum_p \vec{E}_c^p, \tag{3.6}$$

where $\vec{E}_c^p$ is found from (3.5) by setting $j = c$ (because a scattering target needs not be sphere j, but might as well be pixel c). Bringing everything together and rearranging terms, we then find:

$$\vec{E}_c = \vec{E}_c^0 + \vec{E}_c^{\text{scattered}} = \vec{E}_c^0 + \sum_i \frac{\exp \mathrm{i}\Delta\varphi_{ci}}{-\mathrm{i}k r_{ci}} \sum_{l \neq i} [S]_{cil} \sum_{p=1}^{\infty} \vec{E}_{il}^{p-1}, \tag{3.7}$$

where $\vec{E}_c^0$ denotes the incoming PW at the position of pixel c. Be careful not to omit the phase of the incoming PW which is hidden within $\vec{E}_c^0$, cf. (2.47). The term $\vec{E}_{il}^{p-1}$, is the result of the multiscattering-process, cf. (3.3) (after changing indices), with the incoming PW as its starting point: $\vec{E}_{il}^{1-1} \equiv \vec{E}_i^0$ (using the phase belonging to the position of the sphere). Note that when the equation is written in this manner, $[S]_{cil}$ is only needed exactly once. "This manner", implies that

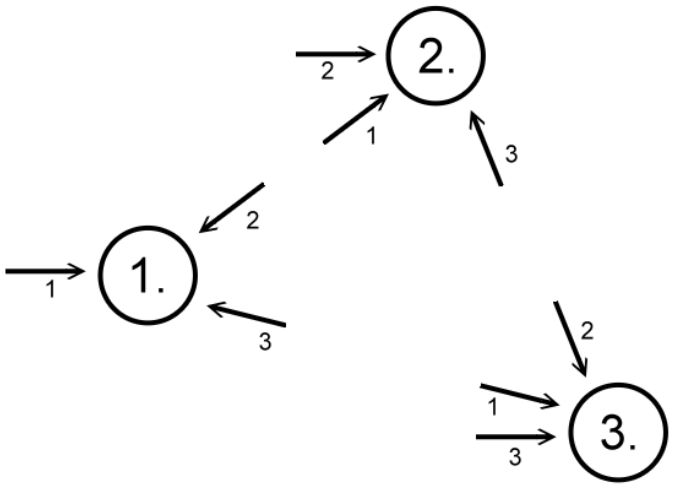


Figure 3.3: This figure illustrates all interactions between particles. The numbers denote the 'source' of the arrow, with the incident field referred to as the number of the sphere (since that index was unused anyway). These numbers may be used as indices to store $\vec{E}_{il}$, as used in the algorithm. So, in an OOP-manner, within each sphere i, N_l fields are stored. The result is a Lattice-Boltzmann-like memory storage, in analogy with fluid dynamics.

we can scatter an accumulated field from each sphere, i, to each pixel, c, where the accumulated field is:

$$\vec{E}_{il}^{\text{accum}} = \sum_{p=1}^{\infty} \vec{E}_{il}^{p-1}. \tag{3.8}$$

This will save both computational resources as RAM.

Fig. 3.3 shows three spheres with all their required interactions (excl. camera). Within the shown arrows, information may be stored. In Fig. 3.1b two fields incident on particle 2 were shown. Those incident fields were called $\vec{E}_{il}$ in e.g. (3.5). Now, for each sphere, i, we may store this field in arrow l. Given the equations, we need to store three fields within each arrow: $\vec{E}_{il}^{p-1}$, $\vec{E}_{ji}^{p}$ and $\vec{E}_{il}^{\text{accum}}$. The first is the result of the previous iteration, 'old', the second is what is currently being computed, 'new', and the third is the accumulator used to scatter to the camera once the multiscattering-process has converged.

Finally, the intensity on the camera has been computed as

$$I_c = |\vec{E}_c|^2 = \vec{E}_c^* \vec{E}_c, \tag{3.9}$$

where I_c is the intensity in pixel c, and $\vec{E}_c$ is the complex electric field, cf. (3.7). Strictly speaking, $I_{\text{true}} \propto I_{\text{computed}}$, depending on e.g. the calibration of the physical camera. Note that this is a time-averaged intensity. The instantaneous intensity would scale with $\cos^2(\phi_0 + \omega t)$, which averages out to a simple constant of proportionality.

3.1.3. Memory Requirement of the Algorithm

Let N denote the number of particles, then from (3.5) it follows that we require

$$8\text{bytes} \cdot \left(\left(\underbrace{1/2}_{\text{sym. in } l \,\&\, j} \right) \left(\underbrace{2}_{\text{S1, S2}} \right) \left(\underbrace{N-1}_{N_l} \right) \left(\underbrace{N}_{N_i} \right) \left(\underbrace{N-1}_{N_j} \right) \right) = O\left(8N^3\right) \text{bytes} \tag{3.10}$$

memory to store $[S]$, which may be reduced by $O(N^2)$ by noting that the backscattering matrix is a constant for a given sphere i, and

$$8\text{bytes} \cdot \left(\left(\underbrace{3}_{\text{3D}} \right) \left(\underbrace{3}_{\text{new\&old\&accum}} \right) \left(\underbrace{N}_{N_j} \right) \left(\underbrace{N}_{N_i \,\&\, \text{IPW}} \right) \right) = 72N^2 \text{bytes} \tag{3.11}$$

to store the fields (cf. Fig. 3.3). The interparticle distance, $\vec{r}_{ji}$, needs not be stored as it is computed with just one (vectorised) subtraction, which is negligible compared to bhmie. Its amplitude requires Pythagoras, but is still very much negligible and not worth complicating the code for.

Note that there is as well a scattering matrix to scatter from particle l, via particle i, to camera pixel c. Which would require require $O(16MN^2)$bytes, similar to (3.10) without the symmetry, and where M is the number of pixels. Since there are already 10^4 pixels to have a 1D resolution of 100 pixels, this is typically higher than N for realistic computations. Sadly, for $M = 100^2$ and $N = 1000$ this results in 160GB, which is too demanding for the RAM. However, as was noted using (3.8), it needs not be stored in RAM, since it is only required once. Instead, $8 \cdot O(3MN)$bytes will be stored during scatter2cam for computational efficiency of bhmie, i.e., bhmie is faster if performed in a batch-process, because it can reuse its Bessel functions and Legendre Polynomials. Calling bhmie requires an additional amount of RAM equal to $8 \cdot O(4MN)$bytes, resultig in a total of $O(56MN)$bytes.

As an example calculation, using 1000 particles and $300 \cdot 300$ pixels, we would require approximately 8GB of RAM during the multiscatter routine. And during the scatter2cam routine, we would require 2.2GB, which occasionally spikes to 5GB. Or, using 602 particles and $300 \cdot 300$ pixels, we require 1.8GB during multiscatter and 1.3GB spiking to 3GB during scatter2cam. Note that the memory requirement of the first example is particle-limited, whereas the second becomes pixel-limited. Since both cases are realistic, one should consider both routines when determining the memory requirement for a given case.

3.1.4. Complexity of the Algorithm

At the top level, the algorithm is structured in three phases, which were shown in Fig. 3.2.

In the initialscatter routine, bhmie first is called exactly once using a vector size of $N - 1$ to scatter the incident field from the first to the second sphere, which is negligible. Then it is called N times to prepare for the multiscatter routine using a vector size $N(N - 1)$. Additionally, there is a loop over j and i, $O(N(N - 1))$, which performs virtually the same calculations as the multiscatter routine.

In the multiscatter routine, at its top level, the algorithm consists of four loops, as shown in Alg. 3 of Sec. B.1. The inner loop is ran a total number of $pN(N-1)(N-1)$ times, and has a complexity of $O(1)$. This results in a complexity of $O(pN^3)$.

In the scatter2cam routine, bhmie, again, is called N times, but now with a vector size of $M(N - 1)$. Then, (3.7) is performed, requiring $O(N(N - 1))$ operations (recall that $\sum_p$ was accumulated in multiscatter, cf. (3.8)). This is to be done for each pixel, c, resulting in a complexity of $O(MN^2)$.

The complexity of bhmie is $O(n_C n_\theta)$, where n_C is the truncation index of the infinite sum from Mie theory and n_θ is the vector size as used above (which is the number of scattering angles).

The accumulated result is summarised in Tab. 3.1, neglecting lower order terms. Please note that complexities are frequently misinterpreted. They describe the behaviour of a code in the limit that the parameters of the complexity (N, M, ...) go to ∞. For practical cases, it may very well be that the neglected lower order terms do in fact contribute significantly more to the required computational resources, simply because they have a greater constant of proportionality than the higher order terms.

	Loop work	Call bhmie	Total
initialscatter	N^2	$N \times N^2 n_C$	$N^3 n_C$
multiscatter	pN^3	-	pN^3
scatter2cam	MN^2	$N \times MN n_C$	$MN^2 n_C$
Total	$N^2(M + pN)$	$N^2 n_C(M + N)$	$N^2(Mn_C + N(n_C + p))$

Table 3.1: The table shows a summary of all complexities involved in the individual routines of the algorithm.

3.1.5. Convergence of the Algorithm

In general, the study of convergence of an algorithm is a difficult one. Instead, a seemingly sufficient condition is relatively easily seen from Eq. (3.4). If that equation is not to diverge as a function of the scattering order, p, we require that

$$k\varkappa > \max_{\theta_s} \left([S](\theta_s) \right), \tag{3.12}$$

which right away gives a measure of how far the FF ought to be for convergence of the FF-approximation. Satisfying this condition will be a sufficient condition, provided that the sum in

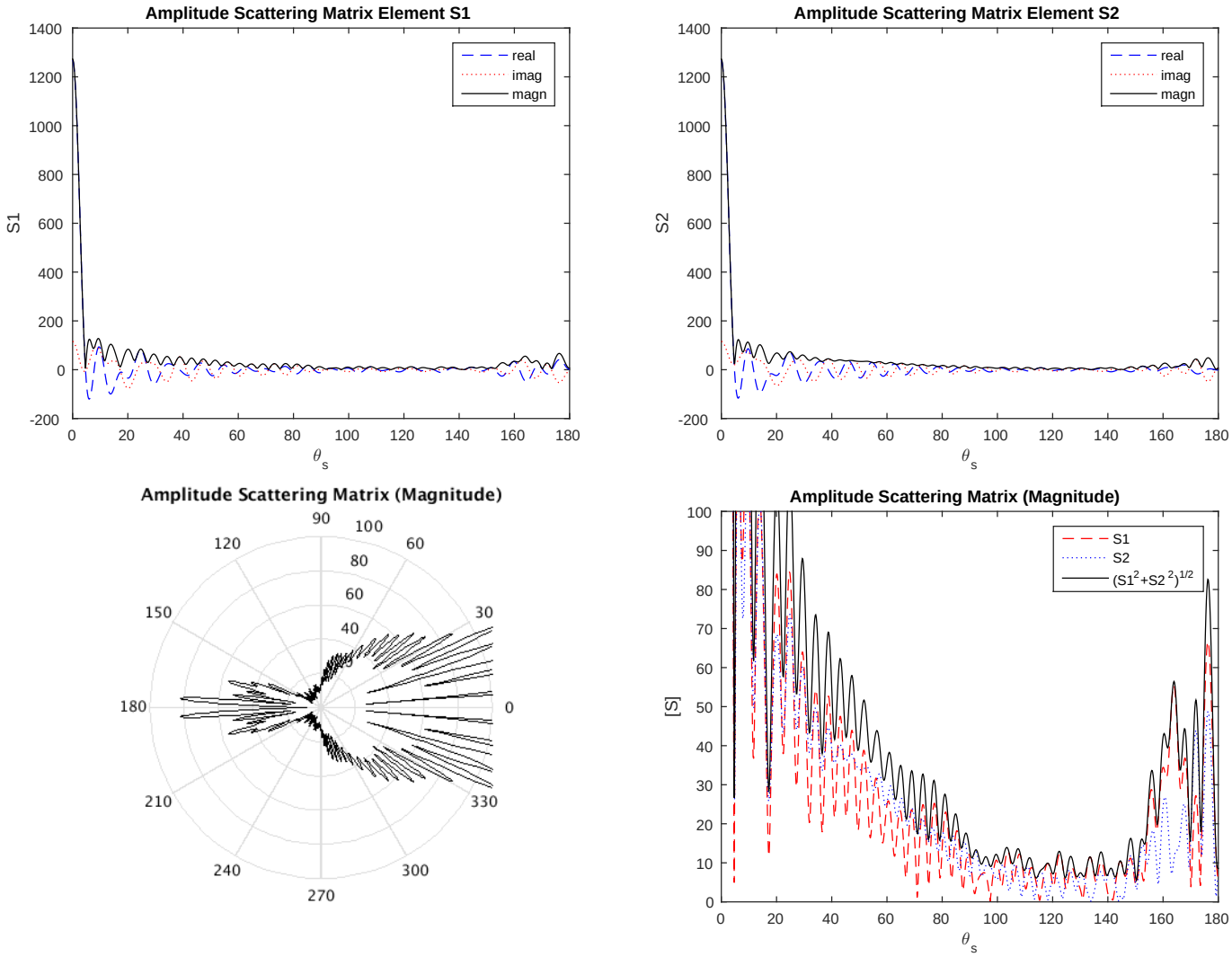


Figure 3.4: The amplitude scattering matrix elements S_1 and S_2 cf. Eq. (2.40) are shown as a function of the scattering angle. The result was generated using the Bohren & Huffman Mie algorithm as described in Sec. 3.1.1. The used parameters are: relative refractive index $m = 1.52$ (in Eq. (2.28)), wavelength $\lambda = 532\text{nm}$, particle radius $a = 4\mu\text{m}$, and size parameter $x \approx 47$.
The bottom-right figure provides a zoomed-in version of the magnitudes, leaving out the huge forward scattering peak. The bottom-left figure plots $\sqrt{S_1^2 + S_2^2}$ in a polar plot.

(3.3) does not grow out of bounds. This is guaranteedly satisfied if

$$k\imath > N\text{max}_{\theta_s}\left([S](\theta_s)\right) \tag{3.13}$$

holds, but for randomly distributed particles it is easily seen that this is an extremely strong condition, since this condition assumes that all particles are distributed such that their scattering angles give a maximum $[S]$ for all particles. Fig. 3.4 show what the amplitude scattering matrix looks like as a function of the scattering angle, θ_s. Clearly, for most values of θ_s, $[S]$ is a factor $O(10^2)$ lower than its maximum value.

With that being said, Eq. (3.13) is certainly too strong, but even Eq. (3.12) appears to be strong in terms of convergence, because if it is not satisfied for some spheres, the algorithm may still converge numerically, although it becomes more difficult. However, the algorithm numerically converging is not the same as the convergence of the algorithm to the true solution. If the electric field is to increase in strength in any scattering event, it becomes difficult to believe that conservation of energy is satisfied. This is however paradoxial, since it is allowed for the electric field to increase in strength in one direction, as long as it decreases in another (which is a direction which was not

of interest and was thus not considered).

Regardless, Eq. (3.12) should yield a practical sufficient condition, although its sufficiency cannot be guaranteed in general.

3.1.6. Approximations and their Consequences

Spherical Particles. The first approximation made, is the fact that we assume that RBCs may be approximated by spherical scatterers. Steinke and Shepherd [23] have shown that Mie theory can successfully be applied to randomly-oriented RBCs. Evidently, RBCs in blood flow will have a preferential alignment. Nilsson et. al. [1] observe a single red blood cell using the T-matrix method and found that the forward-scattering peak of a sphere is lower and broader than for oblate spheroids (which is not the precise shape of a RBC, but it is closer than a sphere). In other words, the amplitude scattering matrix, $[S]$, is affected by the shape of the particles. Also, an azimuthal-dependency is introduced, which is absent for the case of a sphere. Based on their numerical study of a single particle, they conclude that it 'seems inappropriate' to use the spherical Mie theory to study RBCs. Especially if fundamental hematological and morphological properties of RBCs are of interest. Nevertheless, at the same time one would expect that if the particles are dense enough (and RBCs in blood are dense with a typical hematocrit of 45%, see Sec. 2.2), multi-scattering will blur the precise shape of $[S]$, making the spherical assumption statistically more valid.

If the spherical assumption is not to be made, one could attempt implementing the T-Matrix method, as described in Barber & Hill [11]. For time-dependent simulations with $O(1000)$ particles, this will bring along a steep increase in the required computational resources.

Infinite Series Truncation. Now that we have spherical scatterers, the exact Mie solution may be used to solve the scattering of an incoming PW by a single scatterer. There are no further assumptions inside Mie theory: it is exact. However, it is exact using an infinite series representation, which enforces it to be approximated (truncated) if it is to be evaluated. The series is truncated after n_C terms, cf. (3.1). Regardless of how accurate this truncation is, the other approximations made will certainly be more questionable than truncating this series.

A similar problem lies within the computation of the Bessel/Hankel functions and Legendre Polynomials, required by Mie theory: They are computed using recursion, which has the consequence that any rounding errors will eventually blow up. For this reason, Bohren & Huffman [10] explicitely warn their reader not to 'simply' increase n_C, as you may unknowingly make your simulation less accurate.

Far-Field Assumption. Three FF-assumptions have been made: (1) the camera is in the FF, (2) the particles are in each other's FF and (3) particles are sufficiently small for the spherical wave to be a PW over the size of the particle. Given the second assumption, the first assumption is an obvious assumption. The second and third assumptions require elaboration.

Starting with (3), mathematically this is the question of whether a spherical wave has a constant amplitude over the size of a particle and is thus a simple PW. A stronger version yet would be to say that the field is constant altogether (no longer a wave) over the size of the particle, which we will observe first.

A spherical wave, moving outwards from the origin, is given by (the real part of):

$$\Psi(r) = \frac{e^{\mathrm{i}kr}}{kr}, \tag{3.14}$$

where the factor $e^{-\mathrm{i}\omega t}$ has been omitted. Now, if R denotes the *constant* distance between the source of Ψ and the particle under consideration (as before) and δ is the radius of the particle, then $\Psi(R+\delta)$ (at the particle's boundary) may be expressed in terms of $\Psi(R)$ (at its center). Applying a

Taylor expansion to $\Psi(R+\delta)$ around $\delta R/R = 0$, and deploying Newton's Binomial theorem, gives:

$$\Psi(R+\delta) = \Psi(R)\sum_{n=0}^{\infty}\frac{(\delta/R)^n}{n!}\sum_{j=0}^{n}\binom{n}{j}(\mathrm{i}kR)^{n-j}(-1)^j j! \tag{3.15}$$

$$\approx \Psi(R)\left(1+\frac{\delta R}{R}(\mathrm{i}kR-1)+\left(\frac{\delta}{R}\right)^2\left(\frac{1}{2}(\mathrm{i}kR)^2-\mathrm{i}kR+1\right)\right). \tag{3.16}$$

The physical property is the real part of this complex function, which is approximately:

$$\mathrm{Re}\{\Psi(R+\delta)\} \approx \mathrm{Re}\{\Psi(R)\}\left(1-\frac{\delta}{R}\right) - \mathrm{Im}\{\Psi(R)\}\,k\delta, \tag{3.17}$$

and consequently equality implies that $\frac{\delta}{R} \ll 1 \cap k\delta \ll 1$. Recall that these conditions describe the situation in which the particle is so small, that even the phase is constant on the length scale of the particle. Recalling (3.12), which gives a sufficient condition for convergence, the conditions may be rewritten as

$$\frac{\delta}{R} \ll 1 \quad \cap \quad \max([S])\,\frac{\delta}{R} < kR\frac{\delta}{R} = k\delta \ll 1. \tag{3.18}$$

Given a typical value of $\max_{\theta_S}([S](\theta_S)) \approx 10^3$ (size parameter $x = 42$ and refractive index $m = 1.52$), it follows that the second condition is *at least* a factor 10^3 stronger than the first for the present parameters.

Tracing back, we can say something about the weaker assumption made in (3), which assumed a PW rather than a constant field, as was studied above:

$$\Psi(R+\delta) \approx \frac{e^{\mathrm{i}k(R+\delta)}}{kR}\left(1-\frac{\delta}{R}+\left(\frac{\delta}{R}\right)^2\right), \tag{3.19}$$

where solely the reciprocal has been expanded. From this it is seen that a spherical wave may be written as a plane wave if $\frac{\delta}{R} \ll 1$, which interestingly was the weaker part of above condition for a constant field. From (3.19) it is easily seen that the relative error made *in the wave function* is approximately $\frac{\delta}{R}$. E.g., if $\frac{\delta}{R} = 0.1$, a 10% error is made, *in the wave function*.

Regarding (2), saying that the particles are in each other's FF is saying that the interparticle distance should be much greater than a characteristic length. Mathematically, this boils down to the question if it is justified to take the asymptotic value of the Hankel functions in Mie theory. The expansion of the spherical Hankel function for big $z \equiv k\wr$ is given by:

$$h_n^{(1)}(z) = e^{\mathrm{i}z}\left(-\frac{\mathrm{i}e^{-\frac{1}{2}\mathrm{i}n\pi}}{z}+\frac{e^{-\frac{1}{2}\mathrm{i}n\pi}n(n+1)}{2z^2}+O(z^{-3})\right), \tag{3.20}$$

$$= (-\mathrm{i})^n e^{\mathrm{i}z}\left(\frac{1}{\mathrm{i}z}+\frac{n(n+1)}{2z^2}+O(z^{-3})\right), \tag{3.21}$$

which immediately shows that $h_n^{(1)}(z)$ becomes independent of n for big z, and hence could be taken out of the sum in Mie theory. Sadly, the second term scales with n^2 and n is an index that runs until ∞, so z must be a greater ∞ than n.

This problem is resolved by observing the full Mie sum, cf. (2.40). Ignoring the coefficients a_n and b_n and the Legendre functions, it is proportional to $\frac{2n+1}{n(n+1)}$. Consequently, the importance of the terms drops as $\frac{2}{n}$ for big n. So, while indeed the Hankel functions do not converge to the first term in the expansion for all n as z grows big, the series no longer cares about those Hankel functions.

To make this FF-assumption more quantitative, we note that the series will be truncated at n_C, cf. (3.1). If it is again assumed that the terms in the Mie series are equally important for all n, then we obtain a very worst case estimate of what FF means:

$$z \gg \frac{n_C(n_C+1)}{2}. \tag{3.22}$$

For $\lambda = 6 \cdot 10^{-7}\text{m}$ and $r_{\text{sph}} = 4 \cdot 10^{-6}\text{m}$, this results in $z \gg 0.16\text{mm}$. Again, this is a very worst case requirement. A lower z may very well be feasible. If a_n and b_n would be taken into account, which is difficult to do generally, the requirement will soften.

The speed of light is much greater than the fluid speed. It is assumed that the speed of light is much greater than the velocity of the particles (and thus the fluid). The consequence of this assumption is that it is not required to consider the retarded time: the instantaneous time suffices. If this assumption is not made, the position of the scatterers would need to be backtracked to the time at which they scattered the light, rather than taking the current snapshot of all positions.

If this assumption is not satisfied, an error will be made in $[S]$ for each scattering event, because the used scattering angles ($\{\theta_s\}$) will be incorrect. Also, an error will be made in the phase, since the calculated pathlength of the light may differ from the true ones. The first error is difficult to analyse, since the fluctuations in $[S]$ as function of θ_s are a very complex function of the parameters of Mie theory (x, m). The second error may be analysed by noting that the maximum error in the phase occurs when the particle moves in the direction of $\vec{k}$. If the velocity of a particle is v, then in a time Δt, the phase changes according to (with $z = z(t)$):

$$\begin{aligned} \phi(t) &= kz - \omega t = k(z - ct), \\ \phi(t + \Delta t) &= k(z + v\Delta t - c(t + \Delta t)) \\ &= \phi(t) + k(v - c)\Delta t \\ &= \phi(t + \Delta t)|_{v \ll c} + kv\Delta t. \end{aligned} \tag{3.23}$$

From which it follows that $kv\Delta t \ll 2\pi$ is the criterion which justifies the assumption. Now, note that the Δt of interest is the time it takes the light to travel from the light source to the camera, as this is what was assumed to be instantaneous: $\Delta t_{\max} = z_{\max}/c$. The value for $z_{\max}$ is technically infinite, because of the multiscattering-process. However, since each successive scattering order contributes less than the previous one to the final result, the series is truncated. A typical measure for $z_{\max}$ would then be:

$$z_{\max} = O(|\vec{r}_{\text{cam}} - \vec{r}_{\text{CM}}|) + pO(\langle|\vec{z}|\rangle), \tag{3.24}$$

where $\vec{r}_{\text{cam}}$ is the position of the camera, $\vec{r}_{\text{CM}}$ is the center-of-mass position of the particles and $\langle|\vec{z}|\rangle$ is the mean interparticle distance.

Altogether, this results in the following criterion:

$$kv\Delta t = k\frac{v}{c}\left(O(|\vec{r}_{\text{cam}} - \vec{r}_{\text{CM}}|) + pO(\langle|\vec{z}|\rangle)\right) \ll 2\pi. \tag{3.25}$$

This may be made a little more concrete by noting the criterion for convergence (3.12). Using that for $\lambda = 600\text{nm}$ and $a = 4\mu\text{m}$, $[S]_{\max} \approx 10^3$ in the forward direction (as was used in the text just below Eq. (3.18)), taking the camera to be at a distance $f\langle|\vec{z}|\rangle$, and $\langle|\vec{z}|\rangle = mz_{\min}$, we can write:

$$\frac{v}{c}m(f + p)10^3 < k\frac{v}{c}z_{\max} \ll 2\pi, \tag{3.26}$$

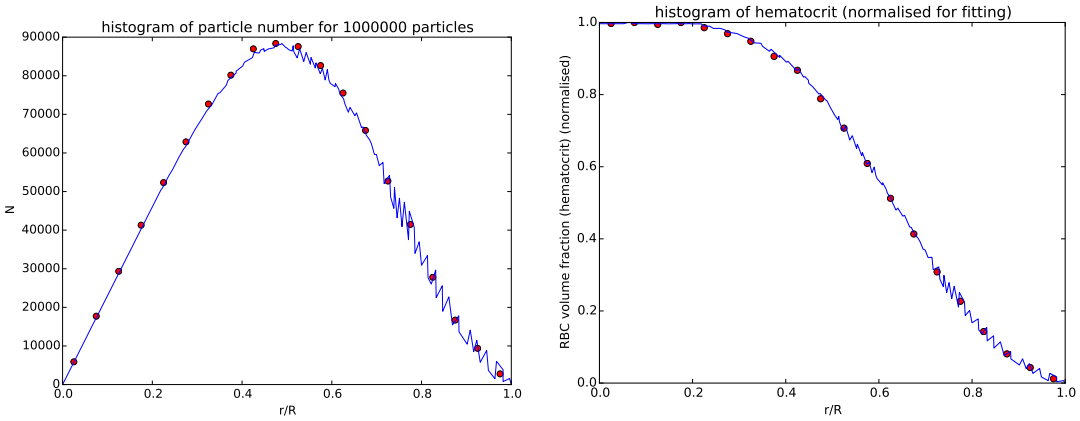
and consequently a worst-case criterion (by numerical convergence) would be (for say, $m = p = f = 10$):

$$\frac{v}{c} \ll \pi \cdot 10^{-5}, \qquad v \ll 10^3\text{m/s}, \tag{3.27}$$

which is extremely easily satisfied for any flow, let alone blood flow. A realistic case will not be too far off from the assumptions for $\{m, p, f\}$ made, and consequently the $v \ll c$ assumption is justified (for the tested parameters).

3.2. Fluids

For the fluids part of our multiphysics problem, OpenFOAM will be used, as described in App. C. This choice is rather arbitrary: any CFD code could be used (e.g. Fluent), provided that it can simulate blood. Initially, OpenFOAM was however chosen for two reasons: (1) my experience and (2) the open-source character of OpenFOAM, making it easy to couple the Optics code with the



(a) Particle number density function / histogram (b) Particle volume density function / histogram

Figure 3.5: Particles are injected using the radial hematocrit profile given in Fig. 2.6. This was converted to a number density profile according to Eq. (2.49). The histogram of how 10^6 generated particles are distributed is shown in these two figures (dots), while the solid line is simply Aarts et. al.'s result (Fig. 2.6). This shows that they do indeed follow the required distribution, which confirms the method of Sec. 2.2.1.
The reader may wonder why the solid line is noisy. The reason is that Aarts et. al.'s plot has a range from $-R < r < R$, which we converted to the range $0 < r < 1$ by simply flipping the data points. The noise is the consequence of the fact that their radial distribution profile was not perfectly symmetric around $r = 0$. For our radial distribution profile we then take the average (which is per definition the smoothed version of the shown noisy line) of their left ($r < 0$) and right ($r > 0$) side. Evidently, the noisiness of the solid line does not affect the histogram, since the solid line is merely a probability distribution and because of binning.

Fluids code. The latter turned out to be a needless requirement, as there is no reason to run/design the code as a monolithic code: the Optics code depends on the Fluids code's output, but the Fluids code does not need any feedback from the Optics code (unlike in e.g. Fluid-Structure Interaction). Nevertheless, there hasn't been any reason to deviate from this initial choice.

The purpose of the Fluids code is to compute the positions of the RBCs as they are convected along with the fluid. These positions will be written to a file (one file per writeTime), which will be converted to the input of the Optics code by the linker.

The ultimate goal is to have the Fluids code model blood flow in a complex artery geometry. But now, to begin with, a simple cylindrical geometry is used, which hopefully permits comparing with experimental data.

3.2.1. Blood

Blood, which was described in Sec. 2.2, is usually modelled as a non-Newtonian fluid, rather than a mixture between blood plasma and its many dispersed particles. Then, any particles of interest (e.g. in Drug Delivery) are modelled as Lagrangian particles. For the present research the RBCs are of interest, which differ from the above particles in two important ways:

Firstly, they are asymmetrical and thus rotation plays a role, most certainly for the light scattering. Currently, the Optics code does, however, scatter the light from particles as if they are spheres and thus rotation does not matter in the current stage, but will in the future. Hence in the present stage, taking this into account for the Fluids code does not have a high priority.

Secondly, they are extremely densely packed with a typical hematocrit of 45%. This will certainly mean that RBCs aren't simple tracer particles and that they are two-way coupled with the fluid, and possibly four-way coupled with particle-particle interactions. However, if blood is modelled as a non-Newtonian fluid, the effect the RBCs have on the fluid has already been taken into account by means of empircal models for the rheology of blood. Then, particle-particle interactions may be forgotten about as well for the used cylindrical geometry (see Sec. 3.2.4), since all particles will be "keeping their lane" without collisions, while any other effects (e.g. drag applied to adjacent "lanes") are in the rheology. This is only true, however, if we use an empircal result for the radial particle distribution, as was shown in Sec. 2.2. Applying Fig. 2.6 and Eq. (2.49) gives the particle number distribution (histogram) resp. the particle volume distribution shown in Figs. 3.5.

Although non-Newtonian effects will affect the (radial) velocity profile in the cylinder (and hence

the time-dependency of the scattering profile), they do not affect the *instantaneous* scattering profile, since this solely depends on the radial particle distribution, which has been taken care of. It will, however, hypothetically affect how the scattering profile evolves over time. Hence at the present stage, non-Newtonian effects have not yet been taken into account, although they should in the near future. So, for now (2.54) will be used to describe the rheology of the fluid.

3.2.2. Lagrangian Particle Tracking (LPT)

Now that RBCs are being modelled using LPT, it is important to determine what forces are relevant for the present problem of blood flow in the arteries. The possibly relevant forces have been described in Sec. 2.4. Using dimensional analysis, we can estimate the order of magnitude of each of those forces.

The forces scale with particle mass $m_p = \rho_p \frac{4}{3}\pi a^3$, fluid mass $m_f = \rho_f \frac{4}{3}\pi a^3$, characteristic velocity scales V (particle) and U (fluid), characteristic length scales H (in the pipe thickness direction) and L (in the stream direction) and characteristic timescales τ_p and τ_f.

- **Particle Inertia Force.** $\sim \frac{m_p V}{\tau_p}$
- **Gravity and Buoyancy Force.** $\sim (m_p - m_f)g$ (in direction of $\vec{g}$)
- **Viscous Drag Force (Stokes).** $\sim 6\pi a\mu(V - U)$
- **Pressure Force.** $\sim m_f U\left(\frac{1}{\tau_f} - \frac{\nu}{H^2}\right)$
- **Added-mass Force.** $\sim m_f \frac{V-U}{\tau_p+\tau_f}$
- **Basset/History Force.** Vojir and Michaelides [24] show that the Basset term is relevant (rigid sphere, viscous fluid) when there are high-frequency velocity variations. And then, it is most pronounced for fluid to particle density ratios of 0.002 to 0.700. For the present case the density ratio is about 1, which shows virtually no influence of the Basset force. Certainly for the simple cylindrical geometry, there are no high-frequency velocity fluctations: it has a steady state solution. Hence the Basset force will be neglected. This conclusion is in line with the findings of De Gruttola et. al. [25].

The typical timescale of the fluid is $\tau_f \sim \frac{L}{U}$. The typical timescale of the particles is $\tau_p = \frac{2\rho_p a^2}{9\nu\rho_f}$ [26, 27]. Defining the Stokes number as the ratio $St \equiv \frac{\tau_p}{\tau_f}$, we can rewrite τ_p in terms of τ_f. Anticipating on the typical numbers below, the Stokes number will be $St \sim 2 \cdot 10^{-7}$, which implies that $\tau_p \ll \tau_f$. This is sufficiently low to assume that the particles are merely tracing particles and thus that the velocity scales U and V will be of the same order of magnitude, $U \sim V$.

Upon applying the above, substituting the relation of the masses, and letting $\Delta U = (V - U)$, we then find:

- **Particle Inertia Force.** $\sim \frac{4}{3}\pi \frac{\rho_p a^3 U^2}{L} St^{-1}$
- **Gravity and Buoyancy Force.** $\sim \frac{4}{3}\pi a^3(\rho_p - \rho_f)g$ (in direction of $\vec{g}$)
- **Viscous Drag Force (Stokes).** $\sim 6\pi a\nu\rho_f \Delta U$
- **Pressure Force.** $\sim \frac{4}{3}\pi\rho_f a^3 U\left(\frac{U}{L} - \frac{\nu}{H^2}\right)$
- **Added-mass Force.** $\sim \frac{4}{3}\pi\rho_f a^3 \frac{U\Delta U}{L}$

Now, typical numbers for the given quantities are (see Sec. 5.1) $\rho_f = 1.16 \cdot 10^3 \mathrm{kg/m^3}$, $\rho_p = 1.1 \cdot 10^3 \mathrm{kg/m^3}$, $\nu = 8.28 \cdot 10^{-6} \mathrm{m^2/s}$, $a = 4\mu\mathrm{m}$ and $H \sim R = 8\mathrm{mm}$. The typical Reynolds number is $Re = 50$, from which we can take $U \sim u_{max} = 5.2\mathrm{cm/s}$. A useful value for L would be the entrance/evolution length for laminar pipe flow, since that is the length over which streamwise variations happen and L was introduced only in relation to τ. A formula for the evolution length fitted using numerical results is given by Durst et. al. [28], which is valid for the given Reynolds number (unlike most engineering formulas out there). The result is that $L \sim 9.2\mathrm{cm}$. Using these numbers, we can estimate the order of magnitude of all forces:

- **Particle Inertia Force.** $\sim \frac{4}{3}\pi \frac{\rho_p a^3 U^2}{L} St^{-1} \sim 4 \cdot 10^{-8}\text{N}$
- **Gravity and Buoyancy Force.** $\sim \frac{4}{3}\pi a^3(\rho_p - \rho_f)g$ (in direction of $\vec{g}$) $\sim 2 \cdot 10^{-13}\text{N}$
- **Viscous Drag Force (Stokes).** $\sim 6\pi a \nu \rho_f \Delta U \sim 7 \cdot 10^{-7}\text{kg/s} \cdot \Delta V$
- **Pressure Force.** $\sim \frac{4}{3}\pi \rho_f a^3 U \left(\frac{U}{L} - \frac{\nu}{H^2}\right) \sim 7 \cdot 10^{-15}\text{N}$
- **Added-mass Force.** $\sim \frac{4}{3}\pi \rho_f a^3 \frac{U \Delta U}{L} \sim 2 \cdot 10^{-13}\text{kg/s} \cdot \Delta V$

From these estimates, it immediately follows that the added-mass is negligible compared to Stokes drag. Pressure is negligible compared to buoyancy. And buoyancy is only important compared to Stokes drag if $\Delta V < 3 \cdot 10^{-7}$. This number is sufficiently low to state that, within the characteristic timescales of interest, Stokes drag is the only important force. This implies that it is to be expected that $V = U$ (with negligible error). The particle inertia force is in fact not really a force: It is the left-hand side of Newton's equation. Therefore, its order of magnitude is merely a measure for the particle response time.

Regarding gravity, having it turned on would not be desireable, as it would slowly destroy the neat radial particle distribution profile, which was obtained from experimental results[2]. At the present stage, the cases should remain simple to be able to draw conclusions more easily.

The particles will be modelled as one-way coupled. This is justified, because at the present stage we will only simulate $N = 1000$ particles. For the parameters of Chap. 5, these have a resulting volume fraction of $O(10^{-7})$, which is sufficiently low to assume one-way coupling, according to Peirano et al. [29]. In actual blood, or even in dilute experiments, particles are much more densely packed. To simulate this effect, the best way is to adopt a non-Newtonian fluid model, rather than two-way coupling the particles. Two-way coupling the particles would only have an effect if one would actually simulate densely packed particles. There is however no reason for the Fluids code to simulate so many particles, because the Optics code currently cannot handle that many particles anyway.

3.2.3. Numerically solving the Navier-Stokes Equations

In OpenFOAM, the Navier-Stokes Equations are discretised using the Finite-Volume Method. This method converts partial differential equations to a set of coupled equations. For a set of coupled linear equations, a single matrix inversion can solve the entire problem. Alternatively, the problem can be solved iteratively without inverting a single matrix. The Navier-Stokes Equations do, however, result in a set of coupled non-linear equations, which are to be solved iteratively. OpenFOAM's pimpleFoam solver is able to do so.

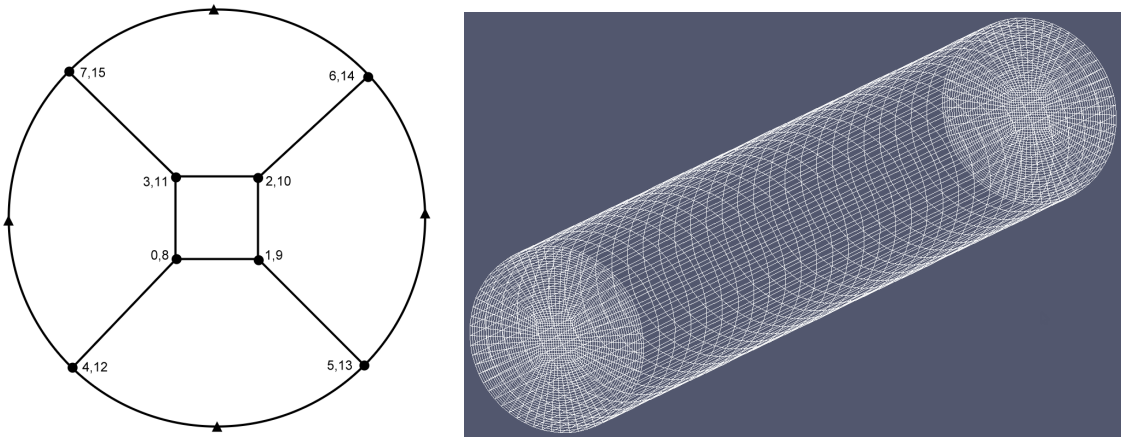
The PIMPLE Algorithm

pimpleFoam uses the iterative "PIMPLE" algorithm, which is a combination of the "Pressure Implicit with Splitting of Operator (PISO)" algorithm [30] and the "Semi-Implicit Method for Pressure Linked Equations (SIMPLE)" algorithm [31]. PIMPLE essentially takes PISO as an internal algorithm for an outer loop as seen in the SIMPLE algorithm. The resulting PIMPLE algorithm is as follows[3].

The inner loop (PISO) consists of solving the momentum and pressure equation in succession using a predictor-corrector approach for a fixed amount of corrector steps. The outer loop (SIMPLE) iteratively repeats the inner loop until a certain convergence criterion is satisfied, or the maximum number of iterations has been reached. In this loop the velocity and pressure (for incompressible flow) are updated using under-relaxation. Once the outer loop has converged, PIMPLE continues with the next timestep. I'd like to think of the loops as finding a "steady state" solution for the next timestep.

[2] Note that from Fig. 3.5 it followed that the experimental radial distribution profile was not symmetric. This might be due to gravity, since the author has not mentioned the direction of gravity in their experiments.

[3] The PIMPLE algorithm appears to be unique to OpenFOAM (correct me if I'm wrong). Its documentation consists merely of the source code for the solver: "$FOAM_APP/solvers/incompressible/pimpleFoam".



(a) Sketch of the used mesh. (b) The resulting mesh.

Figure 3.7: Geometry for the simple cylindrical case. (a) Sketch of the multiblock mesh, to be used by OpenFOAM. The numbers indicate vertex numbers in a tuple of the front ($z = 0$) and back ($z = L$) index. The triangles are anchors to be used to create a circular shape. (b) The resulting mesh, visualised in Paraview.

3.2.4. Geometry / Mesh

In CFD, the flow is solved on a grid, conforming some geometry. While it is possible to simulate virtually any geometry with the Fluids code, in the present research arteries are of interest. This could be a complex artery geometry, or a simple cylinder. As of present, the complex artery geometry has not yet been implemented. The cylindrical geometry allows for comparison with experiments. The mesh for a cylinder is shown in Figs. 3.7.

3.3. Combined Physics

To simulate the whole of physics, the Fluids code and Optics code must be able to communicate, i.e., the output of the Fluids code needs to be translated to the input of the Optics code. To emphasise, the task of the Fluids code is to compute the particle positions as a function of time. The Optics code takes these particle positions (amongst other input parameters) and computes the light scattering intensity at some predefined positions (the camera's pixels). Since this is a one-way coupled problem, the Fluids code may be executed independently of the Optics code, i.e., it does not matter which Fluids code is used (any existing CFD package can be used), provided that there is some script to convert its output to the input of the Optics code. Therefore, the linking process of the two codes will be uncoupled into three separate workflows, as is shown in Fig. 3.9.

The first workflow is related to the Fluids code. In the preprocessing step, the flow geometry with its accompanying mesh are generated, and the particle positions are generated (if the case under consideration uses manual injection, like in Fig. 3.5). Some more complicated cases might require more preprocessing steps, depending on the CFD package used. Immediately thereafter, the Fluids code is executed, writing particle positions as a function of time to a series of files. First relatively big timesteps may be taken to reach the desired state. Then small timesteps (on the scale of fluid dynamics) may be taken to advance the particles on the timescale of interest for optics[4].

The second workflow, which is not coupled to the first workflow and thus seen completely separately, is related to the Optics code. First, the output of the Fluids code is converted to an appropriate format. Then, an Optics input file is generated, which specifies all parameters required for the Optics code (e.g. the refractive indices). Finally, an external looper script loops over all particle position files (different timesteps of the Fluids code) and will repetitively call the Optics code. The result is an intensity file for each timestep.

The third workflow is related to post-processing, as is required for virtually every code in scientific computing. The first step is, however, to convert the output of the Optics code to a more pleasant format for post-processing, i.e., the Optics code can, in theory, compute the intensity in arbitrary

[4]The precise timescale of interest differs with the optical properties of the system, but typically you can think of $O(1\mu s)$, which is significantly shorter than $O(1ms)$, as seen in CFD.

points in space. In practice, it is convenient to define those arbitrary points as a 2D grid which represents a physical camera. The output of the Optics code can then be converted to the format corresponding to this 2D grid to ease the post-processing process.

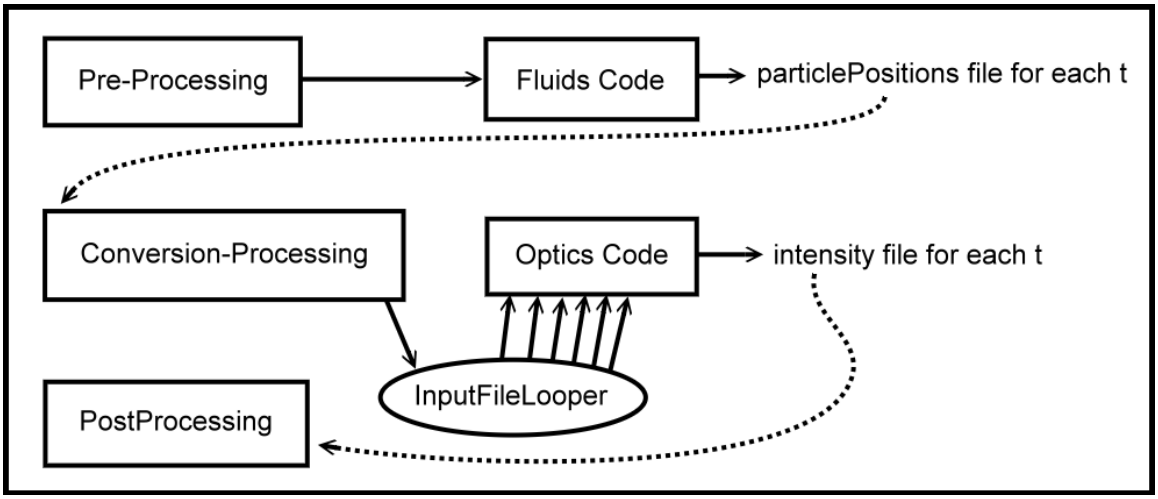


Figure 3.9: This figure shows the workflow of executing the combined code. The workflow consist of three independent branches, which are, in principle, not connected. The first is related to the Fluids code, the second to the Optics code, whilst the third is related to any post-processing activities. See the accompanying text for details.

3.4. Summary

This chapter described the developped Optics code, the Fluids code and how they are linked. These codes were developped to study the interferometric light scattering by flowing spherical particles.

The Optics code uses Mie theory, which describes the scattering of a plane wave by a single spherical particle. It then assumes that the interparticle distance is sufficiently big that the scattered wave is a spherical wave by the time it reaches the next particle: a far-field assumption. Then it assumes that the particles are sufficiently small (or far away) for the spherical wave to be approximately a plane wave over the size of the particle. Now, since the scattered wave is again a plane wave incident on a single spherical particle, Mie theory may be applied to describe the scattered wave caused by the first scattered wave. By continuing this process iteratively, all scattered waves may be collected at a camera, in which multiscattering has been taken into account.

For the fluid dynamics part of the problem, any existing computational fluid dynamics code may be used. OpenFOAM was used for the present research. Its task it to evolve the particles as a function of time within a flow. In the present research, the modelled fluid is a simplification of blood. It is modelled as a Newtonian fluid in which one-way coupled particles are suspended, which are spherical particles that represent red blood cells. The particles will be given a special radial number distribution, which represents the radial distribution of red blood cells in an artery.

The codes are one-way coupled, because the Optics code depends on the Fluids code, but not the other way around. Hence, the Fluids code will be used to output particle positions at all times of interest, and then the Optics code will be executed for each time to compute the scattered intensity profile.

4

Optics Code Validation using Fraunhofer

Writing a code consists of several phases. (1) One starts by digging into the relevant theory, which is mostly relevant for scientific computing, rather than for coding in general. (2) Then one tries to convert the acquired knowledge into pseudocode, i.e., an abstraction of the actual algorithm used to solve the problem at hand. That step is probably the most difficult one if done properly, since it requires a lot of anticipation and insight, although in practice there is a gray boundary between that phase and the next: (3) implementing the algorithm. Sadly, no programmer is perfect from which one may logically deduce that neither am I. (4) Consequently, the next phase in line is the process of debugging, which is the most time consuming phase. Code may behave unexpectedly, because of the well-known '+1' mistakes, pointer-issues, or even logical oversights in the algorithm itself, which implies that phase 2 was not executed properly. So, debugging as examplified here, is the process of convincing yourself that your code does what you designed it for - that it does what you'd be doing by hand had computers not existed. In the case of scientific computing, this does not guarantee that your code correctly predicts nature, so an additional phase follows: (5) validation, in which we compare the output of the algorithm with other results (typically with experiments).

4.1. The Fraunhofer Approximation

One way to validate a code "to some extent" is to compare its output with known exact theoretical solutions. In the case of light scattering, we may use the Fraunhofer diffraction equation (which is a Far-Field (FF) approximation) on some well-known cases: a rectangular aperture and a double slit. In this section, the formalism of Goodman [32] is followed.

The Huygens-Fresnel Principle may be used to describe the response of an aperture to an incoming Plane Wave (PW) (Fig. 4.1). It states that the scattered amplitude, $U(P_0)$, at some distant point P_0 behind the aperture is given by:

$$U(P_0) = \frac{1}{i\lambda} \iint_{\Sigma} U(P_1) \frac{\exp(ikr_{01})}{r_{01}} \cos(\theta) \mathrm{d}\sigma, \tag{4.1}$$

where r_{01} is the magnitude of $\vec{r}_{01}$, which points from P_1 to P_0, θ is the angle between the aperture's outward normal, $\hat{n}$, and $\vec{r}_{01}$. λ is the wavelength and $k = \frac{2\pi}{\lambda}$ is the wave number. Σ is the aperture and $\mathrm{d}\sigma$ is an infinitesimal surface element of the aperture. The Huygens-Fresnel Principle is arrived upon after having made two assumptions: (1) scalar wave theory must hold and (2) $r_{01} \gg \lambda$.

Fresnel approximated the Huygens-Fresnel Principle by assuming that $r_{01} \approx z$, where $z = r_{01}\cos(\theta)$ is the normal distance between P_0 and P_1 (Fig. 4.1) (so $\theta \ll 1$, or near-axis), maintaining only the zeroth order term for $1/r_{01} \approx 1/z$ and up to second order for the phase factor. The resulting Fresnel Diffraction Integral is said to be valid in the 'near-field' of the aperture[1].

[1] I consider the term 'near-field' for this purpose as very paradoxial, given the approximation made: near-axis, i.e., Fresnel is completely valid in the FF as well.

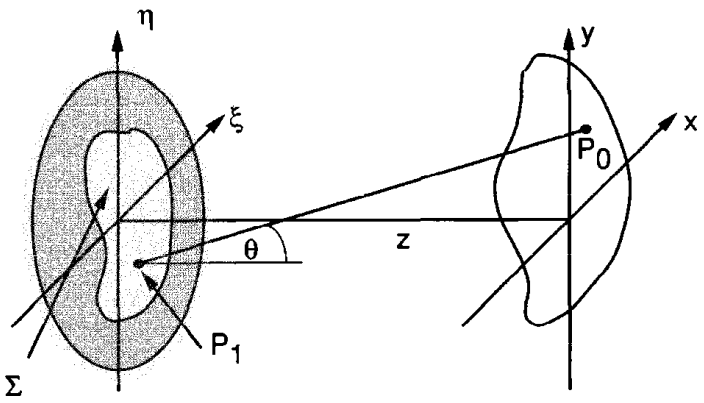


Figure 4.1: Diffraction geometry, cf. Goodman Fig. 4.1 (p65) [32].

The Fraunhofer approximation further approximates the Fresnel integral by only retaining the terms up to first order in the phase, i.e.,

$$z \gg \frac{k}{2}\max(\xi^2+\eta^2), \tag{4.2}$$

where ξ resp. η are the local coordinates of the aperture $\vec{P}_1 = \xi\hat{x} + \eta\hat{y}$, while $\vec{P}_0 = x\hat{x} + y\hat{y}$. The resulting Fraunhofer Diffraction Integral then becomes,

$$U(x,y) = \frac{e^{ikz}e^{i\frac{k}{2z}(x^2+y^2)}}{i\lambda z}\iint_\Sigma U(\xi,\eta)\exp\left[-i\frac{2\pi}{\lambda z}(x\xi+y\eta)\right]\mathrm{d}\xi\mathrm{d}\eta. \tag{4.3}$$

The Fraunhofer approximation is said to be valid in the 'FF', cf. (4.2).

4.1.1. Solution 1: Rectangular Aperture

The Fraunhofer Diffraction Integral (4.3) predicts an intensity profile of the form

$$I(x,y) = \frac{A^2}{\lambda^2 z^2}\mathrm{sinc}^2\left(\frac{2w_X x}{\lambda z}\right)\mathrm{sinc}^2\left(\frac{2w_Y y}{\lambda z}\right), \tag{4.4}$$

where w_{X_i} is the half-width of the aperture in the $\hat{x}_i$ direction, where x_i denotes either x or y, and $A = 4w_X w_Y$ is the area of the aperture. This is a periodic, decaying function for any given z.

The distance between two successive minima in one direction is given by

$$\Delta x_i = \frac{\lambda z}{w_{X_i}}. \tag{4.5}$$

4.1.2. Solution 2: Double Slit

Equation (4.5) for the rectangular slit applies to the double slit as well, provided that we interpret w_{X_i} as the distance between the two slits.

4.1.3. Solution 3: Circular Aperture

A circular aperture will result in an Airy pattern of the form

$$I(r) = \left(\frac{A}{\lambda z}\right)^2\left[2\frac{J_1(kwr/z)}{kwr/z}\right]^2, \tag{4.6}$$

where $r = \sqrt{x^2+y^2}$ is the radius in the observation plane, w is the radius of the aperture and J_1 is the first Bessel function of the first kind.

Tab. 4.1 of Goodman (p.78) [32] gives the location of the minima and maxima for the first few oscillations. E.g., the distance between the first maximum (at $r = 0$) and the second is $\Delta r = x\lambda z/2w$, where $x = 1.635$, which describes the argument of the Bessel function ($J_1(\pi x)$) that gives an extremum, and is *not* the size parameter for which too the symbol 'x' was used earlier in this thesis. The relative intensity of the two maxima is 0.0175. The distance between the first maximum and the first minimum has $x = 1.220$.

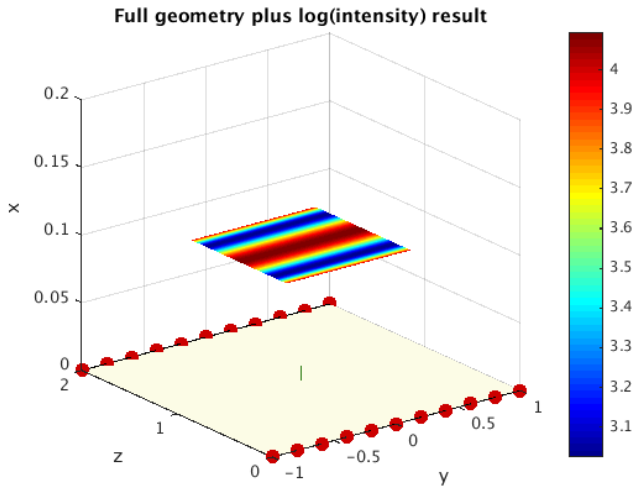


Figure 4.2: The used scattering geometry. An incoming PW is coming in from below with $\vec{k} = k\hat{x}$. The wave scatters on each individual scatterer (red spheres) and forms an interference pattern on the camera, which has $L = 0.1$ in this figure. Note that this figure does not illustrate the FF, as FF figures cannot be used to show the individual elements of the geometry.
In the tested cases, more spheres than shown have been used. The spheres shown are enlarged.

4.2. Cases studied

To compare the Fraunhofer approximation with the output of the code, it is convenient to observe the distance between fringes (maxima/minima). To make this even more convenient, the camera's virtual image is resized to 1x1 (non-dimensionalised). Now, if the distance between fringes is chosen to be $\Delta x_i = 0.1$ (non-dimensional), one may choose the physical camera size such that we expect to see 10 fringes in the direction $\hat{x}_i$ on the camera's virtual image, according to:

$$\frac{\lambda L}{\Delta x_i d} = 2\,|\vec{r}_i|\,, \tag{4.7}$$

where $|\vec{r}_i|$ is the halfwidth of the camera (which may be different for $\hat{x}$ and $\hat{y}$, since the camera needs not be a square), $d = w_{X_i}$ is the 'characteristic size' of the aperture being studied and L is the distance between the aperture and the camera. Here, the camera's normal is taken to coincide with the aperture's normal. So, by choosing the camera's position and size in a specific way, the same distance between fringes is observed for each case studied.

To satisfy the FF requirement of the Fraunhofer approximation (cf. (4.2)), it must hold that:

$$L \gg d^2/\lambda. \tag{4.8}$$

Furthermore, the code is not designed to work with apertures. Rather, it computes the field scattered by an ensemble of spherical scatterers. We may, however, arrange many spheres in a line and state that for an observer sufficiently far away this looks identical to a single slit (two lines for a double slit). The scattering geometry then becomes as in Fig. 4.2. In contrast with the Fraunhofer approximation, it is then reasonable to assume that an Airy pattern is superimposed on the exact Fraunhofer solution.

For the Airy pattern, a similar equation to (4.7) holds true:

$$\frac{x\lambda L}{2w\Delta r} = 2\,|\vec{r}_i|\,, \tag{4.9}$$

using the parameters of Sec. 4.1.3. In the present study, w was chosen such that $w \equiv r_{\text{sph}} = \lambda$. If we now demand $\Delta r = 0.3$ when $x = 1.635$, the hypothesis is that on the camera's virtual image the second maximum lies at a radius $r = 0.3$, in analogy with $\Delta x_i = 0.1$ above.

The cases studied are shown in Tab. 4.1. Note that (4.8) is not strictly satisfied in all cases (chosen such because of resolution reasons), but that does not impose a problem, since it describes when the whole Fraunhofer solution is valid: The distance between fringes might hold at closer distances as well. In addition to the table, different values for λ have been tested, cf. Tab. 4.2.

ID	d or w	i	$\lvert\vec{r}_i\rvert$	
(a)	60	2	1.667	Length of each individual slit
(b)	2	3	50	Distance between the two slits
(c)	0.2	2	500	Distance between two scatterers within a slit (a.k.a. $\imath$)
(d)	0.02	Airy	1362.5	Radius of a scatterer (equals λ)

Table 4.1: Within the present geometry, there are four distinct characteristic sizes, d (or w for Airy). Taking $\lambda = r_{\text{sph}} = 0.02$ and $L = 10^3$, a different $\lvert\vec{r}_i\rvert$ follows for each case. The camera is taken to be a square. i gives the direction in which d exists, which determines the direction in which one may expect an interfringe-distance of 0.1 (a circular 0.3 for (d)). The incoming PW travels in the $\hat{x}$-direction ($i = 1$). There are 301 equidistant spheres within each slit.

ID	$\lambda = r_{\text{sph}}$	d	i	L	$\lvert\vec{r}_i\rvert$
(A)	0.02	60	2	10^3	$1.667 \cdot 10^0$
(B)	0.2	60	2	10^5	$1.667 \cdot 10^3$
(C)	2	60	2	10^6	$1.667 \cdot 10^5$

Table 4.2: These are the cases in analogy with Tab. 4.1.(a), but for different λ. This makes it easier to satisfy the FF assumption. L is as well increased, which was not possible for (A) because of resolution reasons.

4.3. Results

4.3.1. Single-Scattering Far-Field (SSFF) code

The results of the SSFF code validation are shown in Fig. 4.3. Note that, since L is kept a constant, the only difference between the figures is the physical size of the camera's aperture. So (a) is contained within (b) within (c) within (d). E.g., (b) is the middle 1% of (c) (area-wise).

Ignoring (a) for now, (b) and (c) show exactly what was hypothesised: there are bright fringes with a separation of 0.1 in the $\hat{z}$ ($i = 3$) resp. $\hat{y}$ ($i = 2$) direction. The fringes in (c) appear to be further apart as we move off-axis, but that does not invalidate the result: Fraunhofer only says something about the near-axis (i.e., around the camera point $(a, b) = (0.5, 0.5)$), so it is perfectly fine if the distance between fringes changes as we move off-axis.

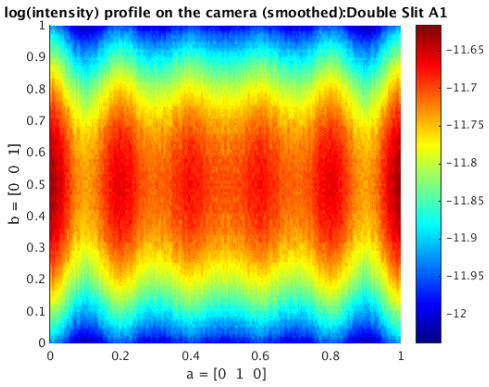
Additionally, the fringes appear to be bending, seen most clearly in (d). Before looking at this bending phenomenon in detail, let us observe the Airy pattern which is superimposed upon the fringes.

The circles in (d) show the hypothesised radii of the extrema, which are of the correct order of magnitude, but do not quite match exactly: they are about 25% off, not a factor 10. One could argue that the pretty accurate match in (b) and (c) was too good to be true, since a double slit is represented by a string of spheres. But at least, there one could argue that in the FF the observer does not see that the slits consist of spheres. That argument does, however, fail for the Airy pattern: no matter how far the observer is, he will not see a circular aperture with a radius equal to the radius of an individual scatterer (here: $w \equiv r_{\text{sph}} = 0.02$). In other words, it would have been surprising to see an exact match here. The only thing that is nice to see here is that the order of magnitude is the same as was hypothesised. In fact, the measured 'Airy pattern' is not an Airy pattern at all. The circles seem to be elongated (i.e., ellipses) in the $\hat{a}$-direction, which happens to be parallel to the slits. The measured pattern is, evidently, the superposition of the influence of each individual scatterer.

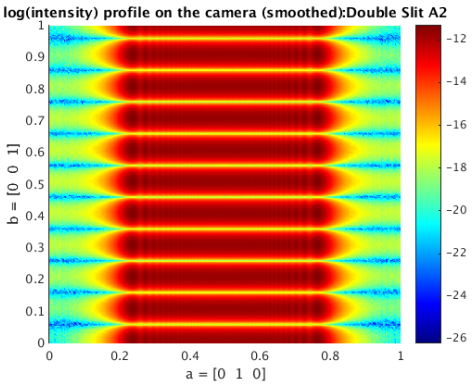
It was also mentioned in Sec. 4.1.3 that the relative intensity between the first maximum and the second maximum should be 0.0175 (for a true circular aperture). This is difficult to see from the figure, but if the global maximum is taken, minus the global maximum when the center circular disk with radius 0.2 is excluded, a value of -1.7345 is found. $\log 0.0175 = -1.7570$, which is a close match. Given that no exact match is expected, this is sufficiently close.

Regarding the bending, this seems to occur under influence of the Airy pattern. At first glance, I'd even go as far as to speculate that there exists a coordinate transformation to convert Cartesian coordinates to elliptical coordinates in such a manner that it matches the observed pattern, i.e., the 'vertical' fringes appear to be orthogonal to the ellipses which describe the extrema. Regardless, the bending is the consequence of the fact that the slits are represented using a string of spheres.

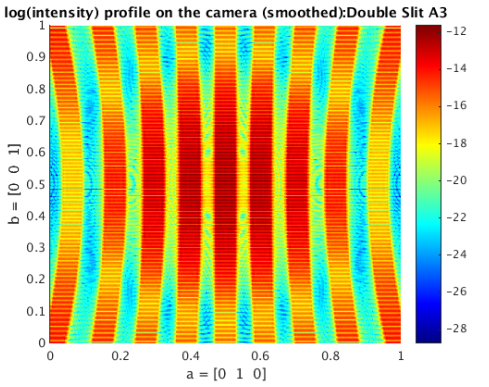
For the previously-ignored figure, Fig. 4.3a, the one and only conclusion is that this figure does not show the FF solution. Fig. 4.4 shows the same figures, but for different values of $\lambda = r_{\text{sph}}$ and



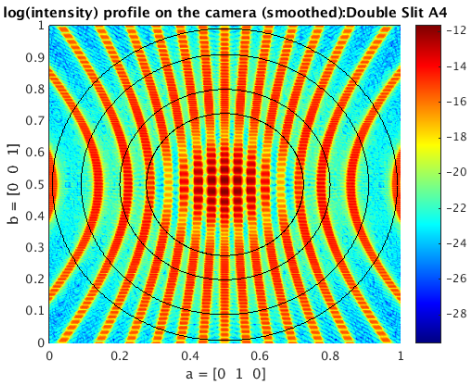
(a) $d = 60$, $i = 2$



(b) $d = 2$, $i = 3$

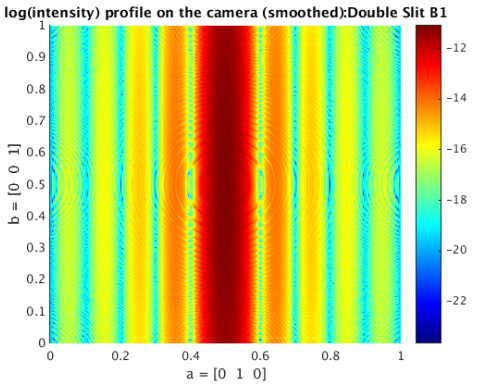


(c) $d = 0.2$, $i = 2$

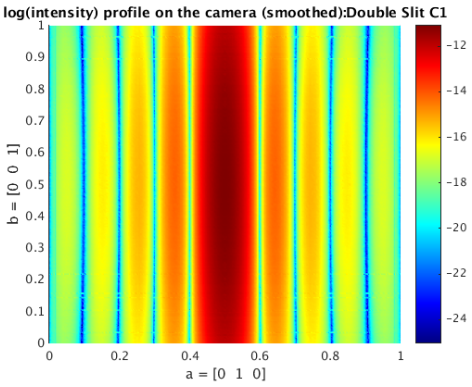


(d) $d = 0.02$, Airy

Figure 4.3: Results for the SSFF code, cf. Tab. 4.1 ($\lambda = 0.02$). In (a)-(c), the hypothesised distance between fringes was 0.1. In (d), circles are shown to illustrate the hypothesised extrema. Outwards: min, max with $\Delta r = 0.3$, min, max, with another maximum at $r = 0$.



(a) $\lambda = r_{\text{sph}} = 0.2$



(b) $\lambda = r_{\text{sph}} = 2$

Figure 4.4: Results for the SSFF code, cf. Tab. 4.2. Compare with Fig. 4.3a.

L, which made it easier to satisfy the FF condition. In these figures, the separation distance of 0.1 is clearly shown, as was hypothesised. Note that there is a wider central peak of size $2 \cdot 0.1$. This is as well included in the Fraunhofer solution: the sinc function in (4.4) is not equal to 0 at zero argument.

It is nice to note that in Fig. 4.4a the Airy pattern is already appearing. Because $r_{\mathrm{sph}} = \lambda$, as λ is increased by a factor 10, the Airy circles become a factor 10 smaller relative to the fringes.

4.3.2. Multi-Scattering Far-Field (MSFF) code

The same analysis as for the SSFF code is performed for the MSFF code. Fig. 4.5 shows the resulting interference patterns. It is quickly noted that the same conclusion will follow.

It is, however, interesting to compare the SSFF results with the MSFF results. Note that the intensity differs consistently by a constant of about 1.5 on the shown logarithmic scale, which is a relative intensity of about a factor 30 in favor of multi-scattering. The only way to explain that difference is if convergence was not satisfied for the present case, i.e., a fixed number of 6 scattering orders have been used, but apparently the spheres are too close together on the length scale of Mie scattering such that after 6 scattering orders the field did not fall off. Or possibly worse, this validation case might be in the divergence regime in which the field strength increases with each scattering iteration, as was discussed in Sec. 3.1.5. For the present case, $k r_{\mathrm{min}} \approx 60$, while $[S]_{\mathrm{max}} \approx 20$ in the forward direction. This satisfies (3.12), but not (3.13)[2] Consequently, the present case is at the border of convergence and divergence according to the theoretical criterion, and appears to be diverging in practice.

Evidently, this implies that the simulation is not physical, which is understandable, because a FF assumption has been used in the code which was not satisfied. Despite the simulation being unphysical in amplitude, from this case it does follow that multi-scattering does not ruin the interferometric properties of the geometry, which is a positive result for as far as this validation is concerned.

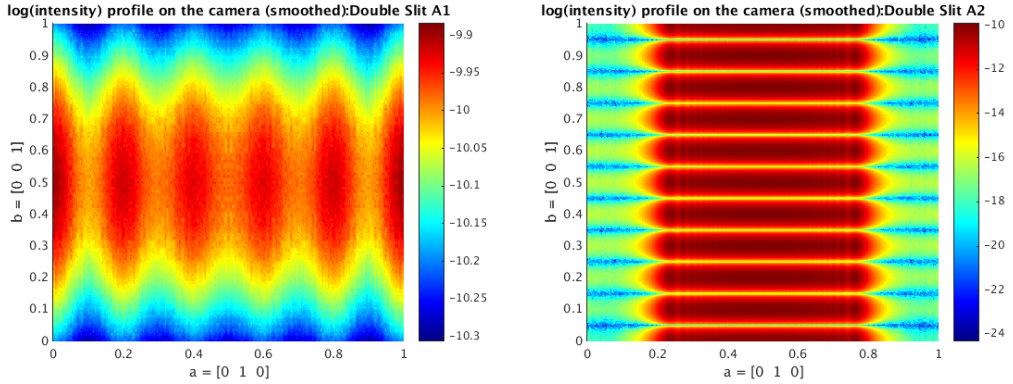
4.4. Conclusions

It has been shown that both the SSFF and MSFF codes have the same interferometric properties as the Fraunhofer solution for the case of a double slit.

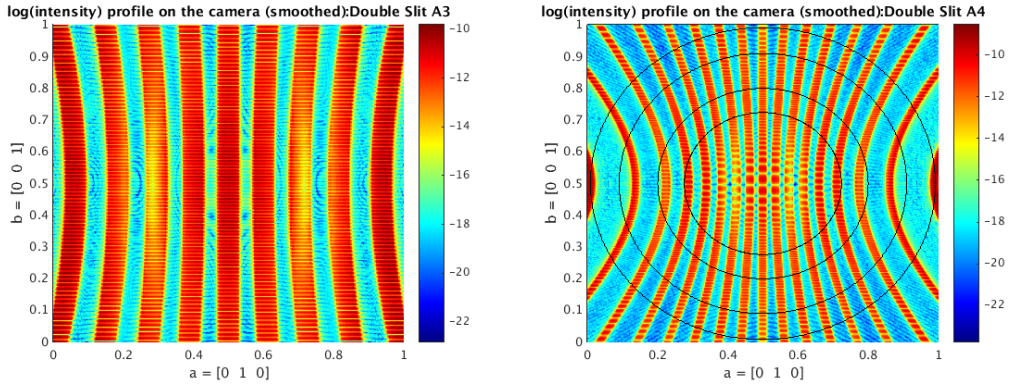
Note that this does not verify that the code correctly describes nature, let the theory, for what it was designed to do: light scattering by an ensemble of spherical scatterers. All that is certain is that the code correctly implements the interferometric part of the theory. It does, however, make it likely that it correctly implements the theory, given that it shows a certain limit behaviour.

Additionally, an Airy-like pattern was observed with rings of the order of magnitude one would expect from a spherical aperture with a radius equal to the radius of the used scatterers. This does as well make it more likely that the code correctly implements scattering by spherical particles.

[2] Note that many spheres scatter in the forward direction, since the spheres are static and distributed over two linear lines.



(a) $d = 60$, $i = 2$ (b) $d = 2$, $i = 3$



(c) $d = 0.2$, $i = 2$ (d) $d = 0.02$, Airy

Figure 4.5: Like Fig. 4.3, but now for the MSFF code, up to a scattering order of $p = 6$ (inclusive) ($\lambda = 0.02$).

5

A Proof-of-Principle Case Study

Now that an Optics code has been created, and coupled to an existing Fluids code, we can take the validation one step further than in Chap. 4, by comparing with experiments. G. Loozen [33] performed experiments for his MSc Thesis, in which light from a laser was sent onto a collection of flowing spheres and the scattered intensity was measured at some distant camera. In this chapter, we will try to reproduce his results numerically.

5.1. Simulation Definition

A simplified description of Loozen's set-up is shown in Fig. 5.1. Briefly, an incoming Plane Wave (PW) (from a laser) is sent onto a cylinder filled with flowing spherical glass beads, after which the scattered intensity is measured with a camera.

5.1.1. Fluids

As for the fluid dynamics part of the experiment, we have multiphase flow in a cylindrical geometry. The main phase is an index-matched water-glycerol mixture and the dispersed phase consists of spherical glass beads. It may be assumed that the flow is fully-developed. The flow is driven using a pump, which is a simplified representation of a human heart. The pump induces a time-dependent pressure gradient over the cylinder with a frequency of $O(1\text{Hz})$. Tab. 5.1 shows all relevant parameters, including the Optics parameters.

To represent this in the Fluids code, the cylindrical geometry as described in Sec. 3.2.4 has been used. Since the flow is fully-developed, cyclic boundary conditions will be used, and wall boundary conditions for the cylinder surface. When, by advancing the time, a steady-state flow solution has been reached (which should conform to the Hagen-Poiseuille solution from Sec. 2.3.1), the particle positions as a function of time may be used by the Optics code. The particles will be given a predetermined radial distribution, as was discussed in Sec. 3.2.1, with the other two cylinder coordinates, $z \in [0, L\rangle$ and $\phi \in [0, 2\pi\rangle$, uniformly generated. Fig. 5.2 shows the resulting flow profile and particle distribution.

Currently, a constant pressure gradient is applied, rather than a pulsating pressure gradient. This will result in a steady-state flow, unlike in a human body or in Loozen's experiments. As will be mentioned in the next (Optics) section, the typical timescale of the speckles[1] is of $O(1\mu\text{s})$, which is a factor $O(10^5 - 10^6)$ smaller than the pulsating flow. Over the timescale of the speckles, the pulsatile flow is effectively constant. In order to extract a heartbeat from the results, pulsatile flow must be considered. Due to the computational times involved, this is out-of-scope for the present research, as will become clear in the next section.

5.1.2. Optics

Concerning the Optics part of the experiment, a laser sends out a plane wave. This plane wave will then scatter on the glass spheres, and the intensity of the electromagnetic field will be measured

[1] The scattered intensity will result in a noise-like pattern with a lot of small-scale features. These features, being the consequence of interferometric scattering, are called speckles.

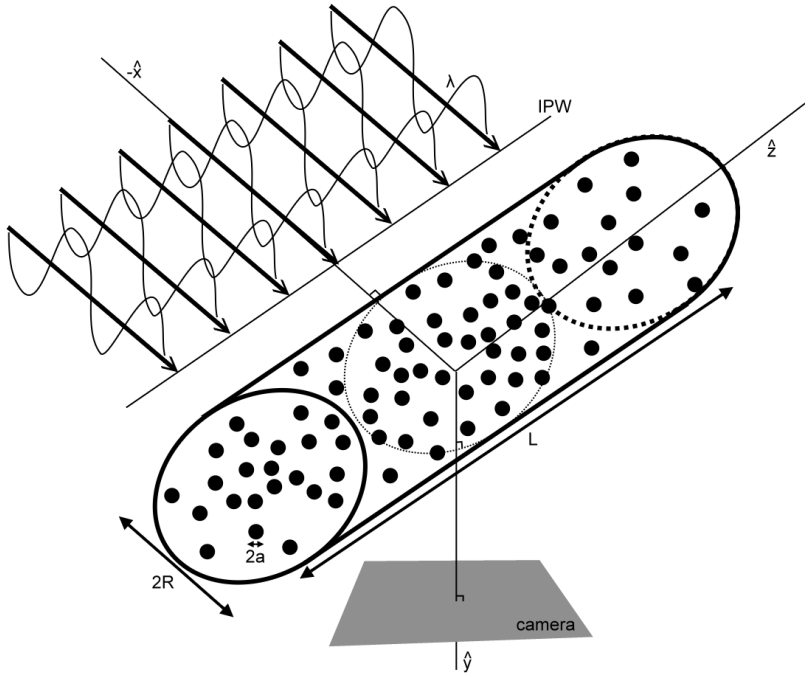


Figure 5.1: The experimental set-up as used by Gyllion Loozen is shown [33], but in a simplified manner. Only the core essence of the set-up is shown, omitting all other optical devices (like lenses) of his set-up.
We have an incoming plane wave, travelling in the $\hat{x}$ direction and polarised in the $\hat{y}$ direction, incident on a cylinder whose axis is aligned with the $\hat{z}$ direction. Within the cylinder there is a glycerol-water mixture in which glass beads are dispersed. In the $\hat{y}$ direction there is a camera, whose normal is orthogonal to the cylinder.

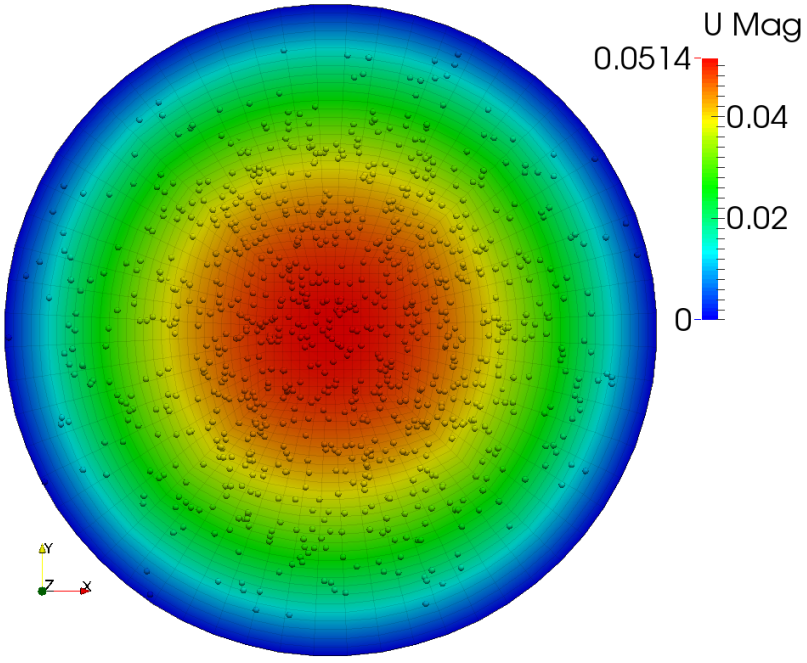


Figure 5.2: Velocity profile and particle distribution when looking along the cylindrical axis ($\hat{z}$). Note that there are more particles near the center of the cylinder. The velocity profile corresponds with the expected profile from the Hagen-Poiseuille exact solution (see Sec. 2.3.1) with an error of 0.7% as measured from the (maximum) value at $r = 0$.

L	1cm	Length of the simulated cylinder
R	8mm	Radius of the cylinder
a	$4\mu\text{m}$	Radius of the simulated particles
Re	50	Reynolds number, cf. Eq. (2.55)
u_{max}	5.14cm/s	Centerline velocity
$\bar{u}$	2.57cm/s	Mean velocity, cf. Eq. (2.56)
dP/dz	30.8Pa/m	Applied pressure gradient
ρ	1157.2kg/m^3	Density of the fluid (water-glycerol mixture)
ρ_p	$1.1 \cdot 10^3\text{kg/m}^3$	Density of the particles
μ	$9.58 \cdot 10^{-3}\text{Pa} \cdot \text{s}$	Viscosity of the fluid
ν	$8.28 \cdot 10^{-6}\text{m}^2/\text{s}$	Kinematic viscosity of the fluid
N_{sim}	1000	Simulated number of particles
N_{exp}/L	$6.5 \cdot 10^8\text{m}^{-1}$	Number of particles in Loozen's experiment
n_{medium}	1	Refractive index of the surrounding medium
n_{rbc}	1.52	Refractive index of the particles
λ	532nm	Wavelength
y_{cam}	25cm	Distance between the camera and the cylinder axis
$\lvert\vec{r}_i\rvert$	1.25cm	Halfwidth of the camera
t_{int}	$100\mu\text{s}$	Integration time of the camera

Table 5.1: This table lists all parameters of the problem at hand for easy reference.

by the camera. The incident field does not hit the camera, i.e., the measured intensity is only due to scattering: $\vec{E}_s$ in Eq. (2.9). The intensity as measured by the camera is given by Eq. (3.9).

Unfortunately, the experimental set-up contains a lot of lenses, which the present Optics code does not support. Hence a way is required to determine at what distance the numerical camera should be at, and what size it should have. Recalling the Fraunhofer approximation of Chap. 4, and assuming that the Airy pattern of a single sphere provides a good measure for the characteristic size of the speckles, it becomes possible to make an educated guess for the camera properties. Using Eq. (4.9) with $\Delta r = 0.1$, and using the parameters of Tab. 5.1, the viewing angle of the camera should be $\frac{|\vec{r}_i|}{y_{\text{cam}}} \approx \frac{1}{2}$, where y_{cam} is the distance to the camera as measured from the cylinder axis. The FF-condition (4.2) requires that $y_{\text{cam}} \gg 0.1\text{mm}$, for example $y_{\text{cam}} = 2.1\text{cm}$. This does, however, correspond to a viewing angle of over $50°$, which is huge. Instead, $y_{\text{cam}} = 25\text{cm}$ and $|\vec{r}_i| = 1.25\text{cm}$ will be used, for a viewing angle of $\alpha = 2\arctan(|\vec{r}_i|/y_{\text{cam}}) \approx 5.7°$, which is more in line with Loozen's experiments, although still large.

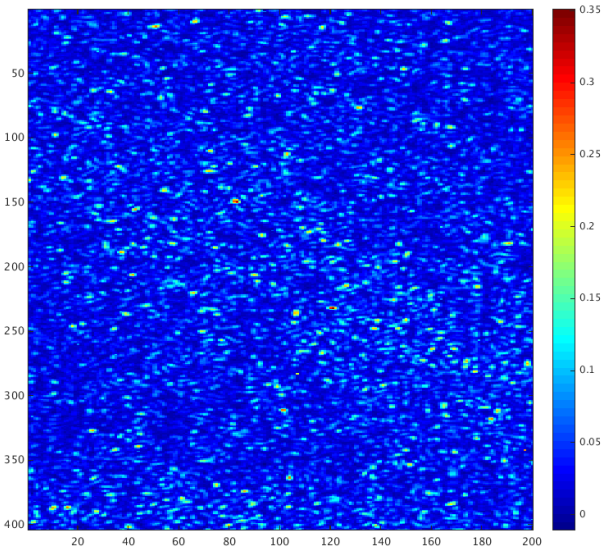
In a physical experiment, it is not possible to measure instantaneously. The camera has a finite integration time, over which it integrates the measured intensity. In Loozen's experiments, this integration time is longer than the typical timescale of the speckles. This causes blurring of the speckles, because the camera integrates over independent intensities, which causes averaging and thus blurring. Loozen has observed several different integration times. As a first comparison, we will for now only look at the results for an integration time of $t_{\text{int}} = 100\mu\text{s}$.

In the developed code, a simulation outputs an instantaneous speckle pattern, i.e., the 'virtual camera' has a zero integration time. Consequently, to mimic the camera, simulations must be ran at different times in rapid succession and then be integrated[2]. If what a camera sees for a full pulsatile flow cycle is to be considered, this process must be repeated several times at different points in the flow cycle. Due to time constraints, the present research will focus on producing results for a single point in the flow cycle.

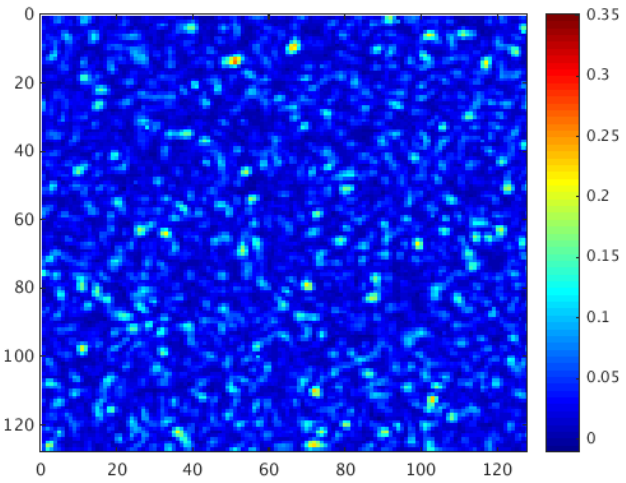
5.2. A Qualitative Comparison

Figs. 5.3a and 5.3b show a speckle pattern as measured by Loozen. Fig. 5.3c shows an instantaneous speckle pattern as acquired from the simulations, and Fig. 5.3d shows a speckle pattern

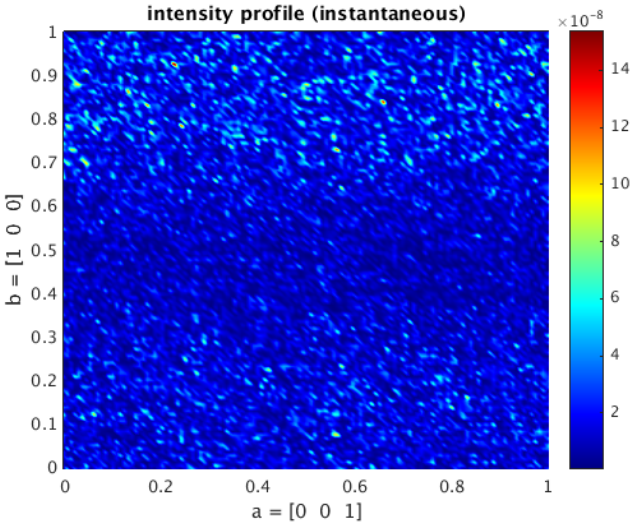
[2]Whereas a physical camera integrates the intensity, we will choose to average the result instead. This is valid, since intensity is defined 'times an arbitrary constant' either way. The advantage of averaging is that any number of integration points will give approximately the same $\langle I \rangle$, which is convenient.



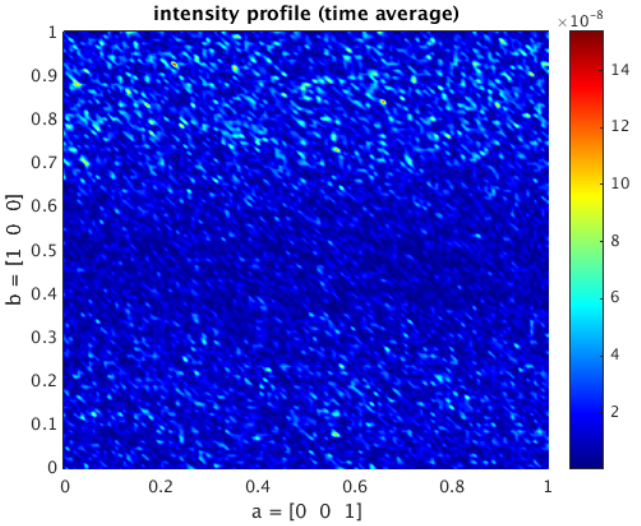
(a) One of Loozen's results.



(b) One of Loozen's results (zoomed in).



(c) Simulation result, instantaneous.



(d) Simulation result, $100\mu s$-averaged.

Figure 5.3: Some intensity profiles / speckle patterns are shown. Note that intensity is defined 'times an arbitrary constant'. There is no reason for the experimental intensities and the simulated intensities to have the same order of magnitude. Only the speckle size and some statistics should be similar. The simulations use 128x128 pixels.

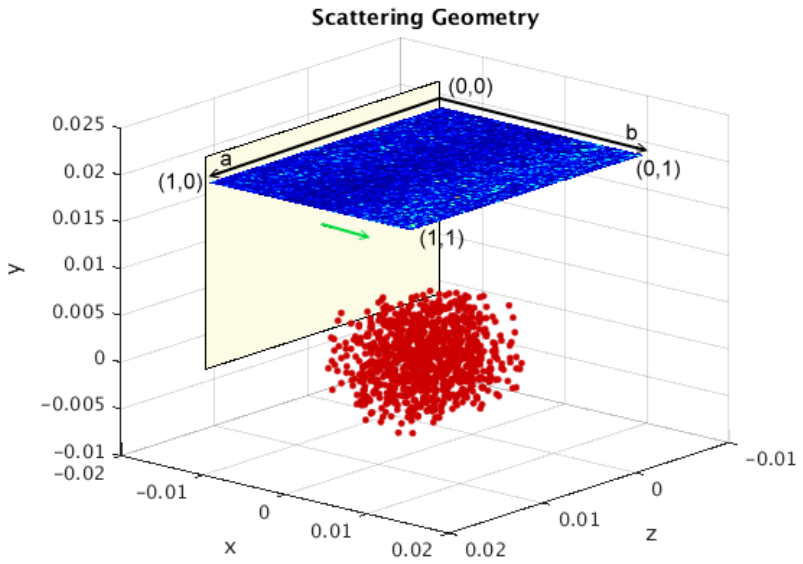


Figure 5.4: The scattering geometry is shown. Fig. 5.1 provided a sketch of the geometry. The pale plane with the green arrow on the left, illustrates the incoming plane wave. The red dots represent the particles, which are distributed over a cylindrical geometry.
This figure further clarifies the definition of the a and b direction, which represent the camera's local coordinate system. (a, b)-tuples are shown at the corner of the camera. Note that a higher value for b corresponds with a lower value for the scattering angle, θ_s.

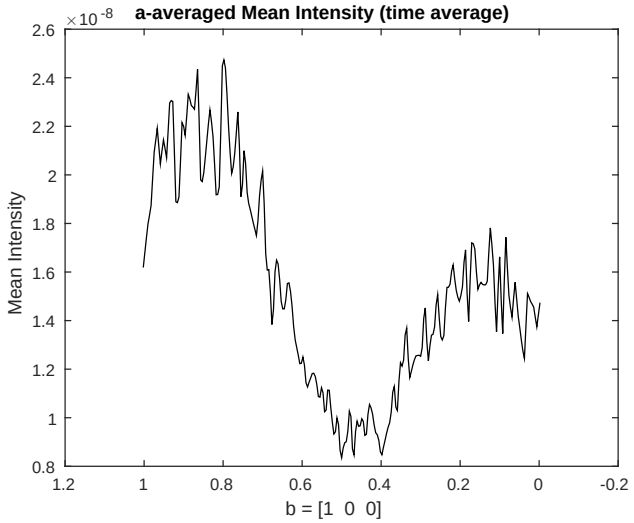
integrated over $100\mu s$. In these figures, the b-direction is in the direction of the incoming wave (and thus a varying (zenith) scattering angle, θ_s), whereas the a-direction is aligned with the cylindrical axis (and is thus aligned with the flow direction, and it corresponds with the azimuthal angle, ϕ). This is illustrated in Fig. 5.4.

Very qualitatively, a similar result is found: the simulated intensity profile on the camera as scattered by the 1000 spheres does indeed become a speckle pattern. Comparing Figs. 5.3c and 5.3d, it should be noted that the speckles are blurred as a consequence of the time averaging. Given the low velocities involved and the very brief integration time, this is difficult to see by eye, and can be seen more easily if the frames are stacked on top of each other and then switched between. Nevertheless, it is most notably seen near maxima, where the dark red of the instantaneous figure becomes lighter in the time-averaged figure, and at points where the green of the instantaneous figure becomes blue in the time-averaged figure, which is a change of color the human eye is sensitive for. This blurring will be made quantifiable in Sec. 5.4.

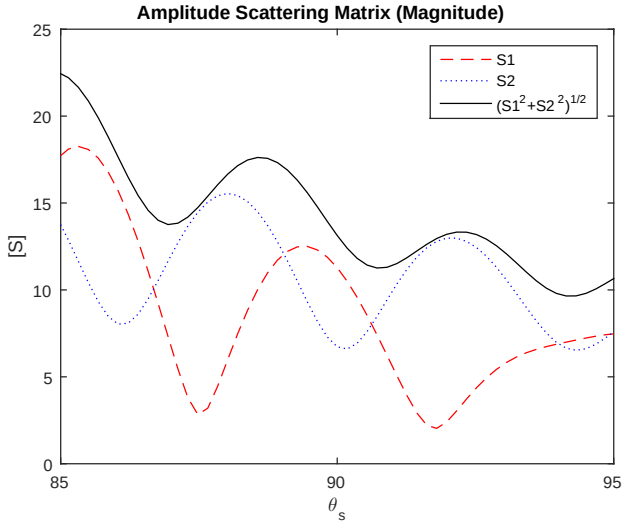
There is however a major difference between the experimental results and the simulations: there appears to be a gradient in the mean intensity (in the b-direction: $\langle I \rangle_a = \langle I \rangle_a (b)$) of the simulation, which is not observed in the experiment. Recall that this corresponds with a varying θ_s. In both cases, no gradient is observed in the a-direction. The cause of this gradient is subject for the next section.

5.3. What Causes the Gradient in the Mean Intensity?

If the average intensity is taken in the a-direction (at constant b), the observed gradient in the mean intensity, $\langle I \rangle_a$, is nicely visualised, as is shown in Fig. 5.5a. If we recall Fig. 3.4, which shows the dependence of the amplitude scattering matrix (2.40) on the scattering angle, θ_s, and then zoom in to small angles (see Fig. 5.5b), the similarity between this graph and the simulated gradient is immediately seen. Approximately the middle 6° are the scattering angles seen by the camera. Comparing these graphs quantitatively, it is observed that the peak value divided by the bottom value equals 2.44 in both cases, when $\sqrt{S_1^2 + S_2^2}$ is taken as the representative value for



(a) Gradient observed in the simulations



(b) Amplitude Scattering Matrix near $\theta_s = 90°$

Figure 5.5: The left figure shows that the mean intensity depends on b, $\langle I \rangle_a = \langle I \rangle_a (b)$, which is equivalent to saying that the intensity depends on θ_s. The right figure shows how the amplitude scattering matrix elements, S_1 and S_2 (2.40), depends on θ_s. It should be noted that the left graph resembles the $\sqrt{S_1^2 + S_2^2}$ line of the right figure[3].

the amplitude scattering matrix[3]. Hence, the form of the amplitude scattering matrix is clearly the cause of the observed gradient in the mean intensity.

Let us, for a minute, pretend that we have no gradient in the mean intensity. We divide the intensity profile in Fig. 5.3d for each a by the $\langle I \rangle_a$ (see Fig. 5.5a) and multiply by the global mean intensity, $\langle I \rangle \equiv \langle \langle I \rangle_a \rangle_b$ (to maintain the same order of magnitude as the previous figures). That way, the resulting intensity profile will have a zero gradient in its mean intensity. This result is shown in Fig. 5.6. Suddenly, the speckle pattern looks a lot more like Loozen's result than before. The simulated speckles appear to be denser than the experimental ones, but given that they are brighter too this is probably merely an optical illusion caused by the color scale (which was dictated by the extreme values present in the data), or as will be seen in Sec. 5.5.2, the simulated grid is rather coarse, which smears the speckles in a smoothed color plot.

This begs the question: How come Loozen does not measure the gradient predicted by Mie theory? Or, how come we measure a gradient, whereas we should not? The definitive answer to these questions is left open for now (Chap. 6 will continue on this), but two important possibilities are offered.

Firstly, the particles in the simulation have a uniform radius of $4\mu\text{m}$, whereas the particles in Loozen's experiment have a size distribution of radii between 1 and $10\mu\text{m}$ with a mean of $5\mu\text{m}$. By noting that the amplitude scattering matrix depends on particle size, it is quite plausible that having a huge size distribution strongly reduces the fluctuations of the amplitude scattering matrix (as seen in Fig. 3.4), up to the point that it is virtually constant over a small viewing angle. If this is the case, the gradient in the mean intensity will become negligible.

Secondly, the simulated particle concentration is about a factor 1000 more dilute than the particle concentration in the experiments (which in turn is more dilute than RBCs in blood). Evidently, that will not change the scattering angles w.r.t. the incoming PW. However, a denser particle distribution will make the effect of multi-scattering stronger ($p \geq 2$ in Fig. 3.2), and multi-scattering happens to be a process in which the scattering angle to the camera is virtually arbitrary. Clearly, as multi-scattering becomes more significant, especially to the point that the $p = 1$ term becomes negligible, it is as if a uniformly random scattering angle is chosen and thus a uniformly random value from the amplitude scattering matrix is chosen. As a consequence, the angular dependence of the speckle pattern will be lost due to the random sampling, removing the observed gradient in the mean

[3]Noting that $I \propto \left|\vec{E}\right|^2$ and that S_1 and S_2 affect orthogonal directions of $\vec{E}$ (cf. Eq. (2.39)) during a scattering process, it is to be expected that $\sqrt{S_1^2 + S_2^2}$ is representative for the intensity.

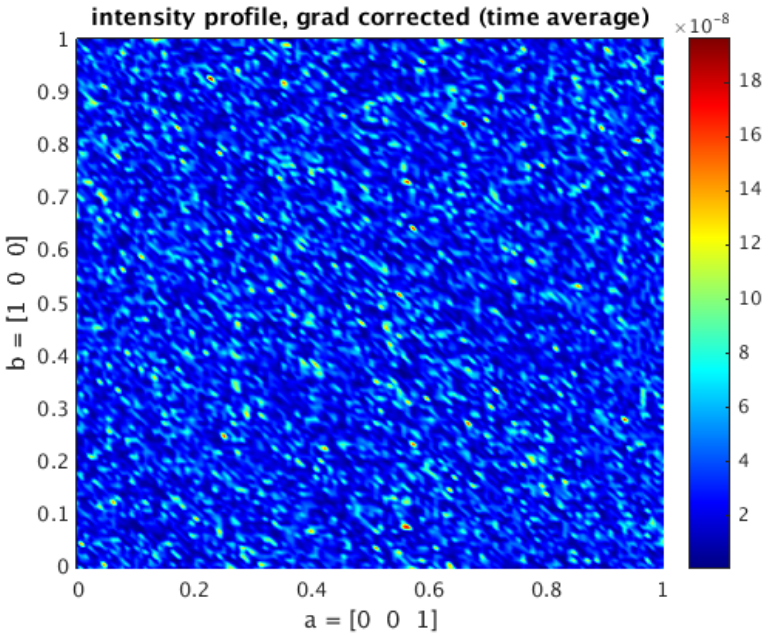


Figure 5.6: Like Fig. 5.3d, but now corrected for the observed gradient in the mean intensity. The speckle pattern now has a zero gradient in the mean intensity and shows a similar speckle size and intensity for all values of b.
The gradient was removed by dividing the original data for each a by the mean intensity, $\langle I\rangle_a(b)$ (i.e., dividing by Fig. 5.5a), and multiplying by the global mean.

intensity.

The metric for quantitative comparison introduced in Sec. 5.4 will be shown to be strongly affected by the gradient in the mean intensity (up to a factor ≈ 2 was observed, depending on the strength of the gradient) in Sec. 5.5. Hence, to make quantitative comparisons, the gradient-corrected results will be used.

5.4. A Metric for Comparison

A scalar measure characteristic for the measured speckle pattern is the so-called 'speckle contrast', which is defined as [34]:

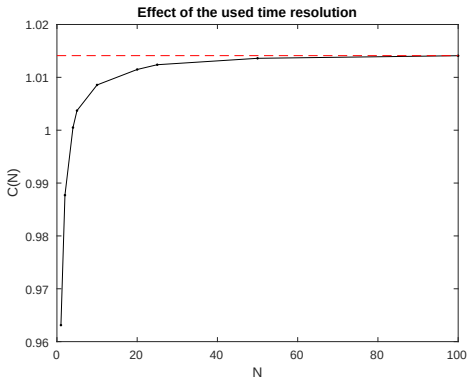
$$C \equiv \frac{\sigma_I}{\langle I\rangle} = \frac{\sqrt{\left\langle (I-\langle I\rangle)^2\right\rangle}}{\langle I\rangle} = \frac{\sqrt{\langle I^2\rangle - \langle I\rangle^2}}{\langle I\rangle}, \tag{5.1}$$

where I is the intensity, σ the standard deviation and $\langle\Upsilon\rangle$ denotes the spatial average of Υ. The speckle contrast describes the degree of blurring.

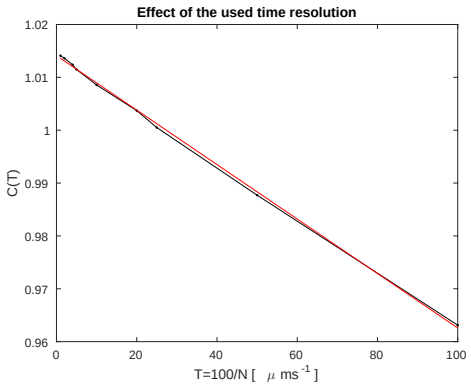
It should be noted that the speckle contrast is grid-dependent [35]. For a physical camera, in the limit of a coarse mesh consisting of an infinite number of speckles per pixel, $C = 0$. That is, all fluctuations average out to $\sigma_I = 0$. The opposite limit of a fine mesh consisting of an infinite number of pixels per speckle is not as trivial. According to Goodman and Parry [36], a static speckle pattern has under ideal conditions ("fully developed") a value $C = 1$, which they proved to be the maximum value for C. In the limit of an infinite resolution, this is the value obtained. Hence a fine mesh will give $C = 1$.

Moving scatterers, like in a flow, will give rise to a dynamic speckle pattern. Recall that a physical camera has an integration time, which causes blurring of these dynamic speckles. This blurring will reduce σ, and therefore C. Hence this metric has an expected value less than 1 when flow is studied, decreasing further as the velocity increases (which decreases the speckle correlation time), and decreasing yet further as the camera integration time increases (which causes integration over more independent speckle patterns). This property permits using the speckle contrast to find the velocity.

The speckle contrast is as well a scalar quantity that can be used to extract some information about the flow. For example, if it is tracked over time, its velocity-dependence for a finite integration time implies that it can be used to study a heartbeat. This is performed experimentally by Loozen, and is a future goal for the developed code. For now, it provides a means of comparison.

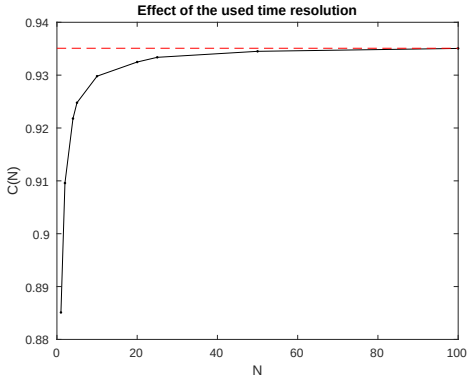


(a) Dependence on the Resolution

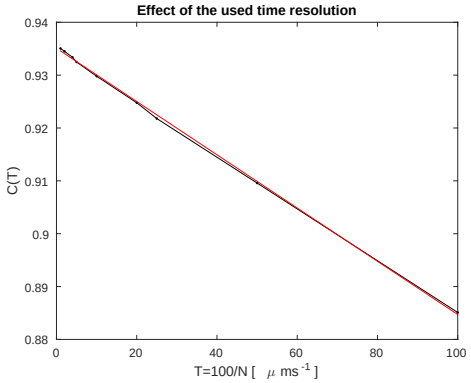


(b) Dependence on the Integration Interval

Figure 5.7: This figure shows how the speckle contrast, C, depends on the used temporal integration resolution. Note that C converges to a limit value in the limit of an infinite resolution. By the very fortunate linear shape of (b), it is easy to predict this value. The extrapolated limit value is $C = 1.014$.



(a) Dependence on the Resolution



(b) Dependence on the Integration Interval

Figure 5.8: Like Figs. 5.7, but now using the 'gradient in the mean intensity'-corrected speckle pattern. The extrapolated limit value is $C = 0.935$.

5.5. A Quantitative Comparison

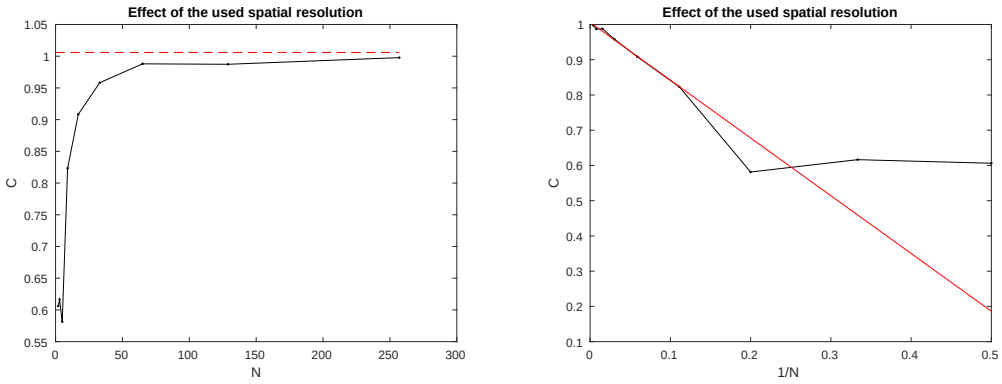
In the developed code, a simulation outputs an instantaneous speckle pattern, i.e., the 'virtual camera' has a zero integration time. Consequently, to mimic the camera, simulations must be carried out at different times in rapid succession and then be integrated[4]. In this section, we will investigate the required temporal and spatial accuracy to yield a convergent result for the speckle contrast, C.

5.5.1. Temporal Resolution

Using the intensity acquired from a 128x128 pixels simulation, we vary the number of temporal points used in the time integration, and observe the required accuracy to obtain the true value for C. Fig. 5.7a shows that as more integration points are used, C converges to a limit value: the true value. When about 50 integration points are used, a result with an acceptable error is obtained.

Interestingly, from Fig. 5.7b it appears as if C depends linearly on the inverse of the number of points taken. (Note that this is proportional to the used timestep for the integration: $T = t_{int}/N$.) Within the scope of the present work, only this case has been examined, and no theoretical explanation has been sought for this relation. If this relation is however true in general, it becomes very easy to extrapolate from a coarse temporal resolution of, say, only 10 points, to obtain the true

[4]Whereas a physical camera integrates the intensity, we will choose to average the result instead. This is valid, since intensity is defined 'times an arbitrary constant' either way. The advantage of averaging is that any number of integration points will give approximately the same $\langle I \rangle$, which is convenient.



(a) Dependence on the Resolution (b) Dependence on the inverse Resolution

Figure 5.9: This figure shows how the speckle contrast, C, depends on the used spatial resolution (the number of pixels of the virtual camera). N is the number of pixels in one direction, and N^2 is the total number of pixels. It is not as clear as in Figs. 5.7 and 5.8 that a linear line in (b) can be used to extrapolate. In fact, it extrapolates to a value greater than 1, whereas all data points are less than 1.

value for C with little uncertainty.

Whereas it is a good thing that C converges to a limit value, according to what was noted in Sec. 5.4, this limit value is higher than the theoretical maximum value ($C = 1$). This would imply that the observed intensity profile is not a speckle pattern at all. Fortunately, given the definition of C (5.1), it is not hard to imagine that the gradient in the mean intensity causes C to increase, as it causes σ to increase. The maximum value is valid for speckles as measured in an experiment, which, judging from Loozen's results, do not have such a gradient.

If we instead use the gradient-corrected speckle pattern from Sec. 5.3, C will change accordingly to more realistic values. Figs. 5.8 show the same procedure applied to the gradient-corrected speckle pattern. The limit value of $C = 0.935$ is now indeed below $C = 1$.

Finally, note from Tab. 5.2 that the time-integrated value for the speckle contrast is $C_{\text{integrated}} = 0.935 < 0.979 = C_{\text{instantaneous}}$. This is exactly the behaviour we'd expect of the speckle contrast: Measuring dynamic scatterers over a finite integration time decreases the speckle contrast.

5.5.2. Spatial Resolution

Apart from a time integration resolution, there is as well a spatial resolution involved in the grid used by the virtual camera. Due to the relatively long computational times involved at the finest level, we will only look at an instantaneous intensity profile, with the finest level being 257x257 pixels. By throwing away $\frac{N_i - 1}{2}$ data repetitively[5] in both a and b, the number of pixels is reduced approximately by a factor of 4.

The result of varying the resolution is shown in Fig. 5.9. The numbers for a coarse ($C = 0$) and fine ($C = 1$) mesh discussed in Sec. 5.4 are indeed observed. The qualitative shape of this figure does too agree with the theory[6].

Fig. 5.10 shows the data the virtual camera measured, without any smoothing applied to the figure. Judging from this figure, the speckles are still subgrid. In comparison, Loozen used 2 to 6 pixels per speckle as measured in 1 direction. For this speckle size, an instantaneous speckle contrast of approximately $C \in [0.85, 0.97]$ may be expected[6].

Judging from Loozen's speckle size, a finer mesh should be considered: the speckles aren't properly resolved. Judging from the values in Fig. 5.9 (which have a speckle contrast ≥ 0.98 for the two finest resolutions), a coarser mesh should be considered, because a coarser mesh would give a lower value for C, which is what Loozen's measurements find. This is in apparent conflict.

There is, however, one crucial difference when comparing the simulation results with experimental results. What a physical camera measures in a pixel is in fact a spatial average of the

[5] The '-1' choice was chosen such that it is possible to throw away data, without the total size of the image shrinking, i.e., this choice implies throwing away all even numbered columns and rows, which maintains the boundary of the figure.

[6] This follows from the theory in Goodman [35]. In Sec. 5.4 the limit values for the spatial resolution were given. Goodman derives a relation for all intermediate values as well.

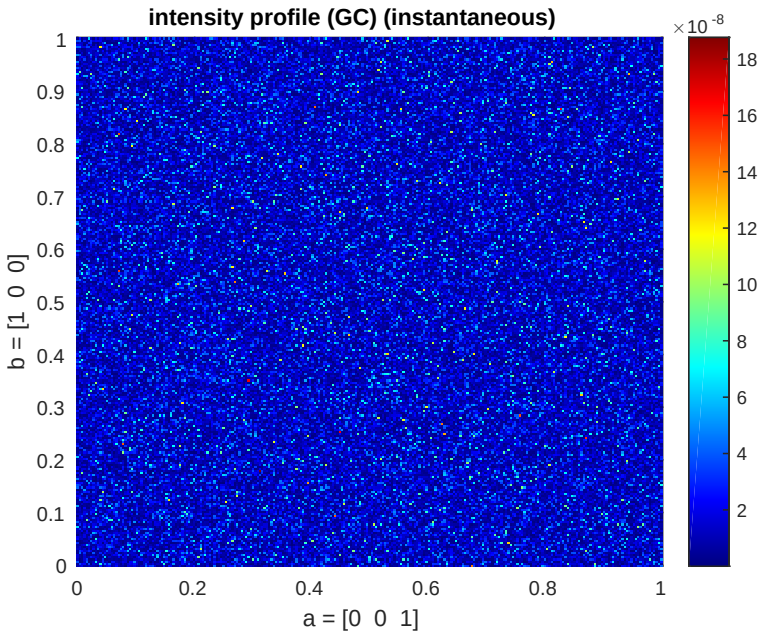


Figure 5.10: The speckle pattern is shown for the 257x257 pixels grid in a pixelised figure (i.e., no smoothing). The data has gradient correction applied to it.

intensity of the light that hits the pixel. That is exactly why a coarse mesh gives a lower speckle contrast: you are averaging away non-uniformities (speckles), such that σ goes to 0. The grid for the virtual camera consists of a collection of points (which are the virtual pixels), with a virtual pixel measuring the intensity at that very point in space. There is no spatial averaging involved. E.g., if there are 500 speckles in a given pixel, the contrast over the virtual camera may still give a value near 1 (since the center value of the physical pixel is taken for the virtual pixel, which is effectively a random number), whereas a physical camera will give approximately 0.

This resolves the apparent conflict: it is not allowed to use Goodman's relation for speckle contrast versus resolution for the virtual camera. With the simulated speckle size of Fig. 5.10, Goodman's relation would already predict a value $C < 0.5$, whereas the simulation gives a value of $C \approx 0.98$. Instead, the spatial resolution of the simulated camera must be finer than a physical camera, since a point in space rather than an area in space is sampled. This will per definition give a poorer solution at the same resolution.

It is however difficult to simply increase the spatial resolution, due to the computational costs involved (some possibilities are offered in Sec. 7.2). For a single instant in time, $M = 257\text{x}257$ pixels with $N = 1000$ particles would last 21 hours[7]. For this number of particles, a RAM of 7.7GB is required in the $\mathrm{multiscatter}$ phase. In the $\mathrm{scatter2cam}$ phase, about 3.7GB is required. From the results of Sec. 3.1.3, it may be found that for $M = 375\text{x}375$ pixels with $N = 1000$ particles a RAM of 7.9GB would be required. Extrapolating the runtime, the total simulation time would be 47h for the $M = 375\text{x}375$ pixels camera. This extrapolation should be interpreted with care: it is valid (and even then an underestimate) as long as all data resides in RAM. As soon as a little data must be stored in swap memory, the computational time will dramatically increase. Instead of increasing the spatial resolution, the viewing angle could be reduced to improve spatial resolution.

Smoothing Data

The fact that the virtual camera samples one point in space could as well be seen as an advantage over a physical camera. All speckle information is lost when a physical camera is used if its resolution is poor. This is not the case for the virtual camera. Evidently, if the speckles are subgrid, the information on the speckle size is lost. However, the statistical fluctuations in the amplitude of the intensity remains present, which is exactly the information required to compute the speckle contrast. If the resolution is poor, the statistics will be poor, but it might still yield reasonable results. A more thorough investigation is required to determine how accurate this can be.

[7] This value is only true if the simulation can use the computer's RAM without interference. Also note that to simulate what a physical camera sees, we found in Sec. 5.5.1 that at least 10 of those simulations are required.

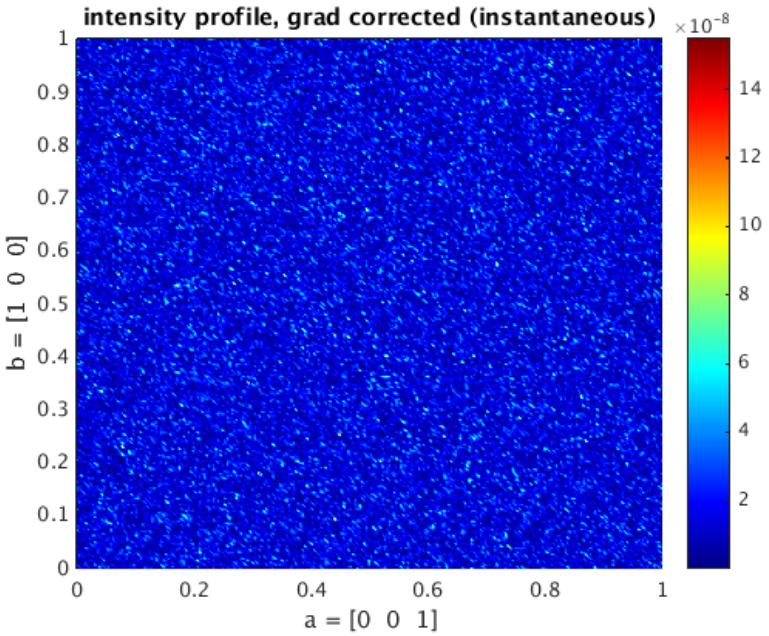


Figure 5.11: Like Fig. 5.10, but now smoothed. A linear interpolation has been applied, and the number of pixels increased by a factor 2.5 in both directions.

Integrated	1.014
Integrated, GC	0.935
Interpolated, Integrated	0.745
Interpolated, Integrated, GC	0.662
Integrated, Interpolated	0.745
Integrated, Interpolated, GC	0.662
Instantaneous	1.057
Instantaneous, GC	0.979
Instantaneous, Interpolated,	0.775
Instantaneous, Interpolated, GC	0.693

Table 5.2: This table shows the speckle contrast, C, when it is computed in different ways, i.e., when the intensity data is processed before computing C. Integrated means a temporal integration over $100\mu s$, see Sec. 5.5.1. Interpolated means interpolating the pixelised data from a 128x128 pixels grid to a 320x320 pixels grid. GC stands for "Gradient in the mean intensity'-Corrected', which was described in Sec. 5.3. Instantaneous means a single snapshot, taken at $t = 50\mu s$.

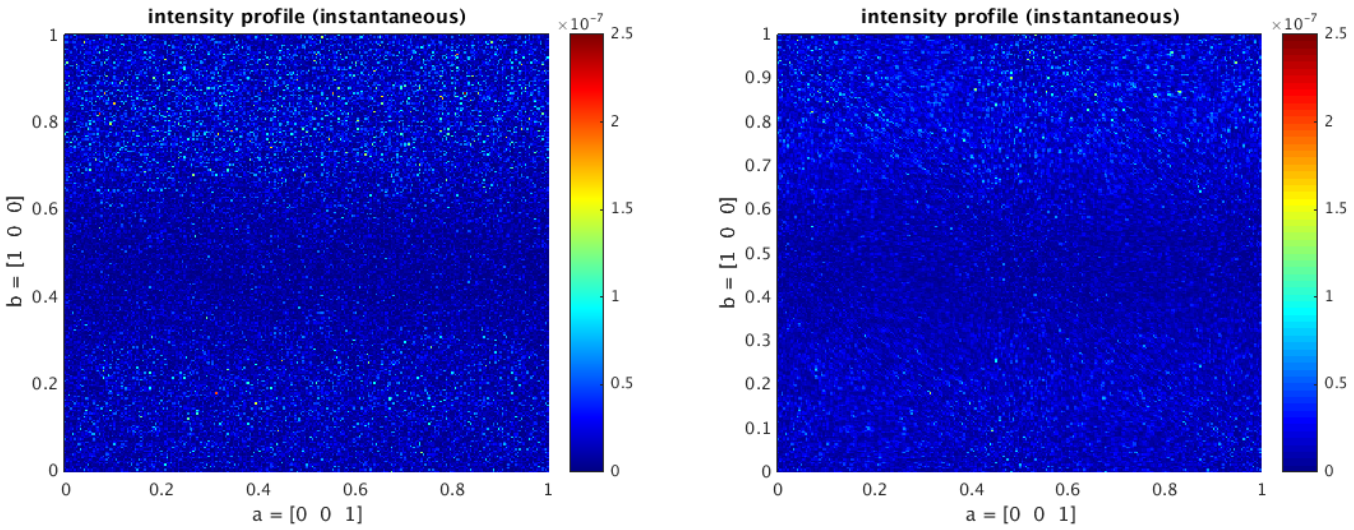
To convert what the virtual camera sees to what the physical camera sees, spatial averaging would be required. Since this was found to be computationally expensive, let us consider spatial interpolation instead: smoothing the data. This yields figures like is shown in Fig. 5.11. This method will increase the measured speckle size to unphysical values, but it is nonetheless interesting to see what effect it has on the speckle contrast. The result is shown in Tab. 5.2. It is noted that the speckle contrast decreases as a consequence of the smoothing.

To illustrate whether this smoothing process yields phyiscal information, one can study how the speckle contrast changes as a function of the smoothing. From this it is found that the speckle contrast drops to effectively the same value, $C = 0.706 \pm 0.2\%$, no matter how many grid points are added. Even if only one pixel in either direction is added. This is a strong indication that smoothing data is unphysical and therefore a bad idea. Fig. 5.12 illustrates why this occurs. It is seen that all peaks are immediately smoothed. Thereafter, judging from the found speckle contrasts, smoothing does no longer affect the speckle contrast.

The Required Resolution

Now let us return to Fig. 5.9a. We have concluded that the speckles are subgrid, and thus that the used resolution was not appropriate. We also concluded that despite the resolution not being appropriate, for the virtual camera the amplitude fluctuations in the intensity were not lost[8]. Therefore, Fig. 5.9a still provides useful information as to whether the spatial resolution of the used grid is appropriate. Or, whether there is enough data points for the statistics to yield accurate results. Whereas it should be checked for a simulation with a smaller viewing angle, it is likely appropriate

[8]This is the one and only reason that a speckle contrast $C = 1$ is found, whereas the speckles are in fact subgrid.



(a) Original data (b) Smoothed data

Figure 5.12: This figure illustrates the effect of smoothing on the original data (not gradient corrected). (a) shows the original data of 257x257 pixels. (b) shows interpolated data using 259x259 pixels.

to take about 150x150 pixels to yield an accurate result.

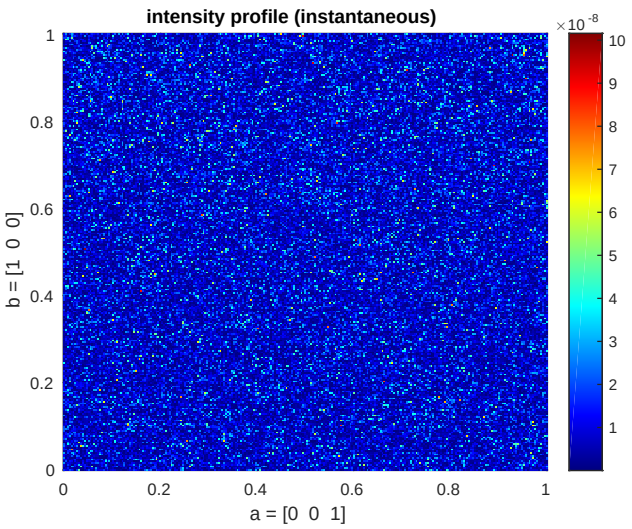
Lastly, two short simulations are performed using the parameters of Tab. 5.1, except for the value of $|\vec{r}_i|$, which will be reduced to 1.25mm and 0.219mm to achieve a viewing angle, α, of $0.57°$ and $0.10°$ respectively. With these reduced viewing angles, the speckle patterns shown in Figs. 5.13 are found. To return to Sec. 5.3, it should be noted that there is no longer a visual gradient in the mean intensity present. This is because the viewing angle became so small that the amplitude scattering matrix is effectively a constant over the observed area (compare with Fig. 5.5). Actually, in Fig. 5.13a the gradient in the mean intensity is of $O(10\%)$ of the global mean intensity. The fluctuations (similar to those observed in Fig. 5.5a) are of the same order of magnitude. In Fig. 5.13b, the gradient in the mean intensity is a factor 10 smaller than the fluctuations. In other words, the gradient did not disappear: it only became smaller and therefore will have a lesser effect on the speckle contrast. From Fig. 5.13b it may be concluded that the typical speckle size of the simulated system corresponds to an angle of about $\alpha = 0.003°$, where $\tan\alpha \equiv d_{\text{speckle}}/L$, with d_{speckle} the speckle size in units of length, and L the distance between the center of the sample and the camera.

Fig. 5.14a shows how the speckle contrast depends on the spatial resolution for the case in which the speckles are fully resolved (fine mesh). Since the speckles are about 8 pixels in Fig. 5.13b, the speckles become subgrid at about $N = 33$. It is noted that for $N \geq 33$, an expected behaviour is observed, whereas for $N < 33$ the physics breaks down. This is purely caused by how a virtual camera works, as was discussed for the coarse mesh: a physical camera performs averaging (causing C to go to 0), but a virtual camera samples (causing C to behave unpredictably due to a too small sample size).

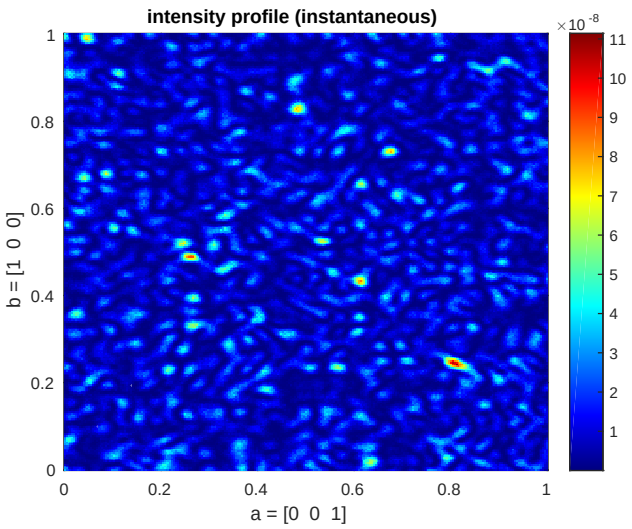
5.5.3. A Final Comparison

If the knowledge acquired in the previous sections is now gathered, it becomes possible to make a final comparison with the experimental results of Loozen. Comparing the simulation result in which the speckles were fully resolved (Fig. 5.13b) with the experimental result (Fig. 5.3b), a similar pattern is observed.

The speckle contrast can, however, not be directly compared. Loozen found a speckle contrast of about $C = 0.75$ (see Fig. 4.10d of his report), whereas the speckle contrast in Fig. 5.13b is $C = 0.997$. The latter is not equal to the first, because the latter is measured instantaneously (which should give $C = 1$ for fully-developed speckles at an infinite resolution according to the theory), whereas the first is measured using an integration time of $100\mu\text{s}$. For a poor resolution study of the effect of the integration time (with $\approx 5 \cdot 10^{-4}$ pixels per speckle), it was found that time integration decreases the speckle contrast, as is to be expected. In that case it would decrease to

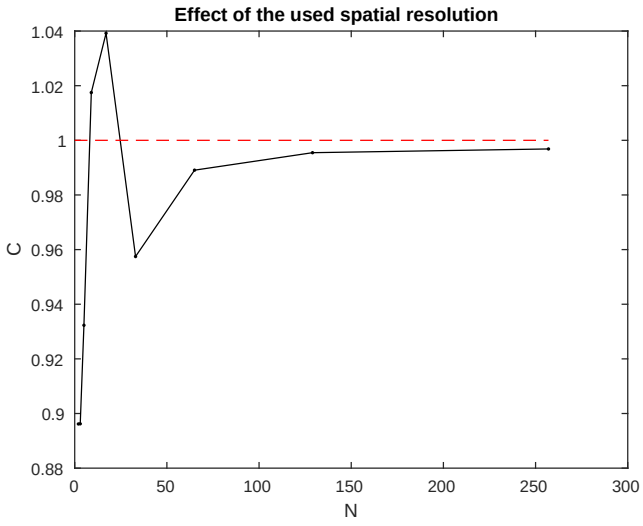


(a) $\alpha = 0.57^\circ$

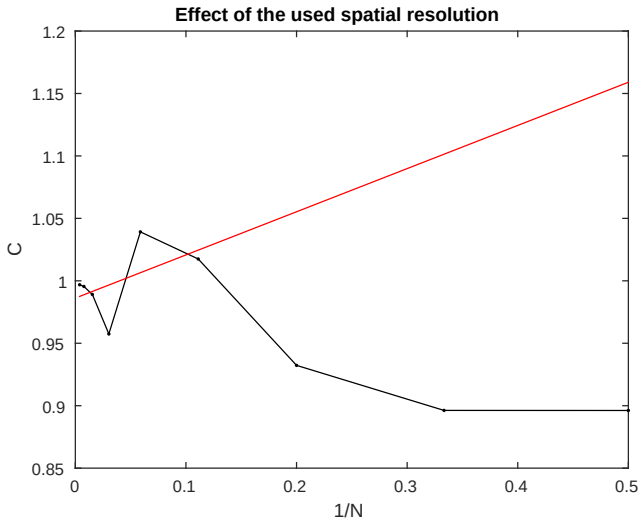


(b) $\alpha = 0.10^\circ$

Figure 5.13: The speckle patterns for much smaller viewing angles than in Fig. 5.10 are shown. Area-wise, (b) is the middle 3% of (a). In either case, 257x257 pixels have been used, and no smoothing was applied to the plot. Note that these figures are *not* 'gradient in the mean intensity'-corrected.



(a) Dependence on the Resolution



(b) Dependence on the inverse Resolution

Figure 5.14: Like Fig. 5.9, but now for a viewing angle of 0.10°. It is shown how the speckle contrast depends on the number of pixels used. The camera consists of N^2 pixels. From (b) it is seen that the linear extrapolation, which seemed accurate before, is nonsensical.

$C = 0.935$.

Due to time constraints, in the present research the time integration with the correct spatial resolution could not be performed, which is required to make a more quantitive comparison. For now, the conclusion is that the findings are at least not in conflict.

5.6. Conclusions

In this chapter, light scattering from dynamic spherical scatterers was studied. These scatterers are effectively tracer particles, moving in a steady-state flow in a cylindrical geometry. A virtual camera samples the reflected light intensity in a finite number of pixels. The resulting intensity profile is a speckle pattern.

It was found that the simulated speckle pattern has one major difference with the experiments: it has a significant gradient in the mean intensity in the direction in which the scattering angle, θ_s, changes. See specifically Figs. 5.3. The cause of this gradient was found to be the amplitude scattering matrix in Mie theory. Its angular dependency directly translates to the angular dependency in the simulations. Why the simulations show this dependency whereas the experiments do not is subject for Chap. 6.

In order to compare results quantitatively, the concept of the speckle contrast, C, was introduced. The speckle contrast is a scalar metric which describes the amplitude of intensity fluctuations within the speckle pattern.

It was found that, for the 'gradient in the mean intensity'-corrected data, the speckle contrast is within the bounds one would expect from the theory on speckles. When simulating the finite integration time of the camera, an expected behaviour was observed, i.e., the speckle contrast would decrease as a consequence of the temporal integration. A reasonable temporal accuracy could be achieved by extrapolating from as little as 10 integration points for the present parameters. The spatial resolution encountered a problem. Currently, the used spatial resolution was too coarse, resulting in the speckles not being properly resolved. This resulted in finding a higher value for the speckle contrast than in experiments. This is the opposite behaviour as one would expect from the theory, which is because a virtual camera does not measure the same way as a physical camera does. A virtual camera samples one point in space and takes it as the value for the pixel. A physical camera integrates over all light that is incident on a given pixel. The latter causes averaging, whereas the first does not, resulting in the opposite behaviour.

Future work should consider much smaller viewing angles for an increased spatial resolution. A small study showed that the speckles have a size of about $\alpha = 0.003°$ (see Sec. 5.5.2). Thereafter the acquired knowledge of the time resolution may be used to compare the speckle contrast more quantitatively with the experiments. If its behaviour is then studied as a function of the Reynolds number, the speckle contrast can be validated relative to the velocity. A validation for a single velocity is difficult, especially at a low velocity (which was considered here), since it is sensitive to the viewing angle[9]. Nonetheless, judging from Loozen's results, regardless of the viewing angle, a similar shape is found when C is plotted against the Reynolds number.

[9] Loozen observed a range of viewing angles, which differed by less than a factor 3, but resulted in a $O(40\%)$ difference for the speckle contrast at a given Reynolds number and a given integration time of $100\mu s$.

6

Smearing of the Scattering Matrix

In the simulated scattered intensity profiles, a gradient in the mean intensity was observed, which is absent in the experimental data with which we compared our results. In Sec. 5.3 this gradient was analysed and the cause was found to be of the form of the amplitude scattering matrix. Two plausible explanations are offered to explain why this feature is present in the simulations, whereas it is absent in the experiments.

The first option was that the gradient smears out due to the range in particle sizes present in the experiments[1]. Since all these different particle sizes have their own amplitude scattering matrix, any fluctuations in the amplitude scattering matrix might cancel out to obtain an effectively constant gradient in the mean intensity.

The second option was due to multi-scattering. If multi-scattering is significant, then any scattering event has a statistically random scattering angle. If so, the amplitude scattering matrix is sampled at random points, which will result in a constant value[2]. That will too cause a negligible gradient in the mean intensity.

6.1. Particle Size Distribution

To see whether the particle size distribution present in the experiment is the cause of the gradient in the mean intensity, we could study light scattering from an ensemble of non-uniformly sized particles. Currently, the Optics code does not support this feature. In order to still study different particle radii, the Optics code may be executed for each radius separately. Although these results should be interpreted with great care, they may offer insights as to whether the particle size distribution could be responsible.

From Fig. 6.1 it can be seen that different particle radii give rise to a different gradient in the mean intensity. From Fig. 6.1a it is particularly noticeable that different sized particles have a different order of magnitude for the mean intensity. This can be interpreted as that small particles scatter much less light in the orthogonal direction[3] than big particles do. The result of this is that big particles are to be expected to contribute most to the scattering process *per particle*[4].

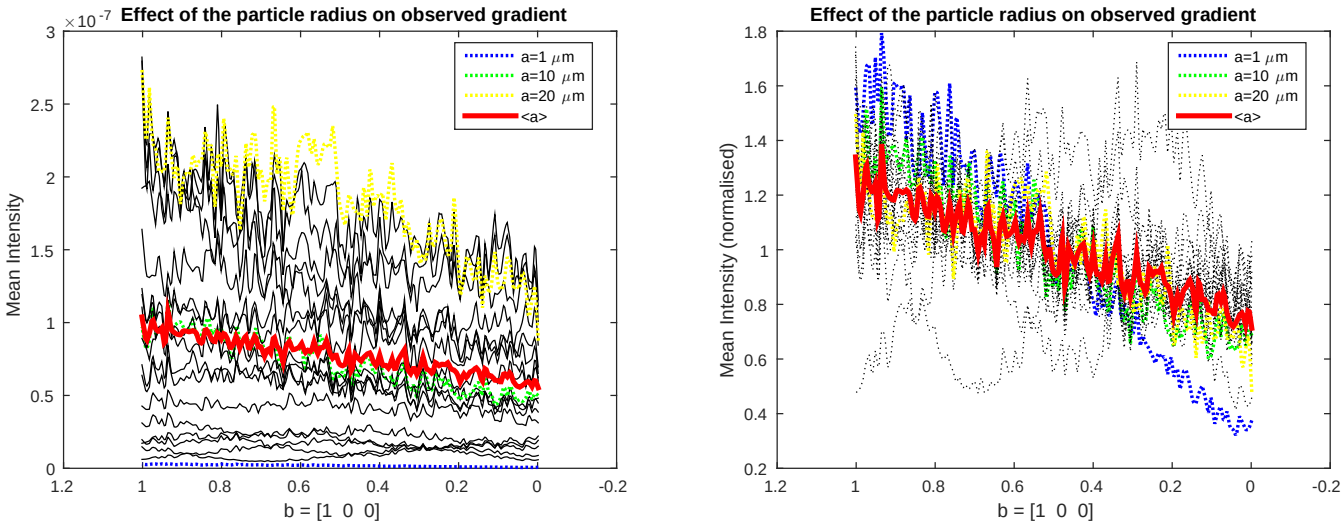
In Fig. 6.1b all particle radii have been given an equal weight by normalising them. Here it is seen that, with only one exception, all particle radii have a similar negative gradient. The precise slopes differ, and so do the small-scale features, but since the slopes are all negative, the slope will never average out to a zero slope. Therefore, the red line showing the slope as obtained from the averaged intensity field, does as well show a gradient.

[1] The simulations were performed using a constant particle radius of $4\mu m$. The experiments used particle radii between $1\mu m$ and $10\mu m$ with a mean of $5\mu m$

[2] This value will be the expectation value of the amplitude scattering matrix, which is simply the arithmetic mean for a uniformly random distribution.

[3] Actually, if one would observe the full amplitude scattering matrix for a small particle and compare it to a big particle, it is found that a small particle in general scatters less light than a big particle. For a particle of $1\mu m$, $\sqrt{S_1^2 + S_2^2} \approx 123$ in the forward direction. For a big particle of $20\mu m$ this value is $40.1 \cdot 10^3$. This is a factor 300 difference.

[4] If per volume would be considered, small particles will in fact contribute more to the scattering than big particles do, *if* we may take the numbers found in Footnt. 3 to be representative.



(a) As measured.

(b) Normalised.

Figure 6.1: This figure shows the observed gradient in the mean intensity for different particle radii. Particle radii between $1\mu m$ and $20\mu m$ are shown in steps of $1\mu m$, with four lines emphasised in a different color. $\langle a \rangle$ gives the gradient of the weighted arithmetic mean of all 20 intensity profiles. Weighted means taking the arithmetic mean after normalising the figures.

As mentioned, these results should be interpreted with great care. Intensities were averaged arithmetically to obtain the mean intensity gradient. Adding intensities is only allowed if the intensities are incoherent (independent), but when you have a size distribution of particles, then particles of different radii will have their scattered fields interact via multiscattering. However, if multiscattering is not responsible for the absence of the gradient in the experiments, then this interaction is negligible and the results obtained in this section are valid. From this it would immediately follow that the particle size distribution is **not** responsible for the absence of the gradient in the experiments, *if* multiscattering is not responsible either. Now if multiscattering would be responsible, then it does not matter what effect the particle distribution has: multiscattering already explains it.

6.1.1. Speckle Contrast

Finally, now that results for different particle sizes have already been generated, it is interesting to see what effect the particle size has on the speckle contrast. This result is shown in Fig. 6.2. From this figure it is seen that the gradient in the mean intensity has a big influence on the speckle contrast. If the gradient is removed, the speckle contrast appears to be independent of particle size. However, this result should again be interpreted with great care, since the values for the speckle contrast are very closely to the maximum value. To truly conclude that the speckle contrast is independent of particle size, simulations must be executed for a case which results in a lower speckle contrast (smaller viewing angle). For now, the conclusion would be indeterminate.

6.2. Multiscattering

In order to investigate whether multiscattering could be responsible for the gradient in the mean intensity, two simulations are performed using two different fixed number of scattering orders. The first will use a scattering order $p = 1$, which corresponds to single scattering[5]. The second uses a very big value, $p = 300$.

Fig. 6.3 shows the result of these simulations. Little difference is observed between these two greatly differing scattering orders. Specifically Fig. 6.3b shows that the maximum error between these figures is of $O(1\%)$. That is to say, multiscattering does not affect the solution.

The reason multiscattering did not affect the solution for the present simulation is that the particles are too dilute. In line with the reasoning of Sec. 3.1.5, this means that the typical

[5]Recall that the scattering order, p, was defined such that incident field scatters via a maximum of p spheres to the camera.

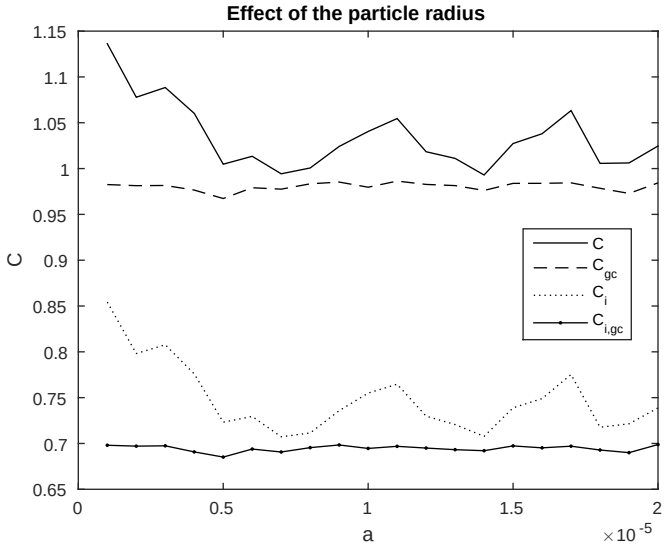
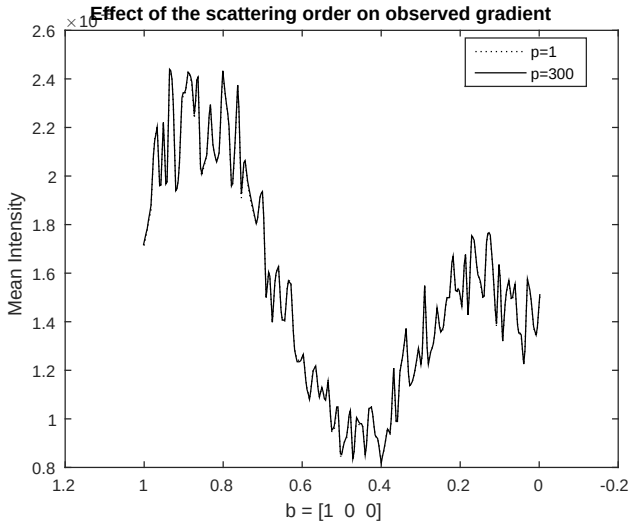
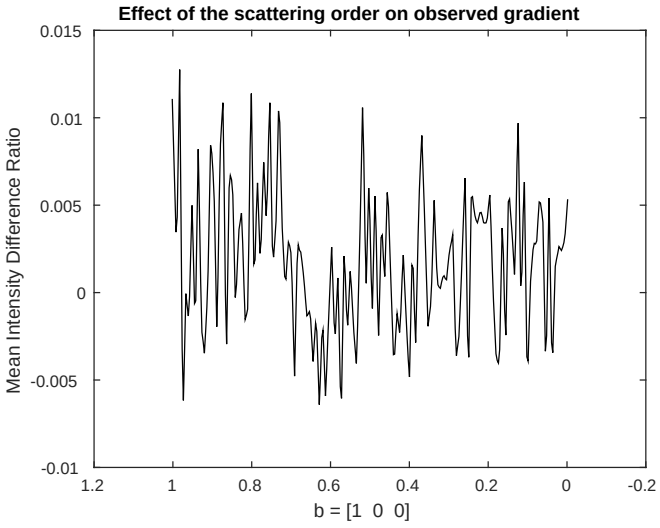


Figure 6.2: This figure shows the speckle contrast, C, versus the particle radius, a. Four lines are shown, where 'gc' stands for "Gradient in the mean intensity'-Corrected' and 'i' stands for 'Interpolated'.



(a) Gradient in the mean intensity



(b) $(I_{300} - I_1)\,/\mathrm{mean}\,(I_{300})$

Figure 6.3: The gradient in the mean intensity is shown for two different scattering orders. $p = 1$ corresponds to single scattering, whereas $p = 300$ means that the light scattered between up to 300 particles before hitting the camera. Little difference is observed.

interparticle distance times the wave number, kr, is much greater than the maximum value of the amplitude scattering matrix, i.e., scattered waves die out faster than they can find a new particle to interact with.

Since multiscattering did not affect the solution, it is not possible to say whether multiscattering is responsible or not. However, given that it did not affect the simulation at all, it means that the gradient in the mean intensity certainly was not surpressed due to multiscattering. That implies that multiscattering may very well be the reason for the absence of the gradient in the experiments, although the results of this section cannot prove it.

6.3. Conclusions

In this chapter, it was investigated why the simulation shows a rather strong dependence on the scattering angle, θ_s, resulting in a gradient in the mean intensity across the speckle pattern. This gradient is absent in the experiments. The two offered explanations were the particle size distribution present in the experiments, while the simulations use a uniform particle size, or the degree to which multiscattering is important, because the particles in the simulations are a factor 1000 more dilute than in the experiments.

Regarding the particle size distribution, it was found that *if* multiscattering is not relevant, then neither is the particle size distribution. From this it can safely be said that the particle size distribution is not the cause of the gradient in the mean intensity.

Concerning multiscattering, it was shown that multiscattering was negligible in the simulations. Therefore, multiscattering certainly wasn't able to surpress the gradient in the simulations. This does not prove that multiscattering is responsible for the absence of the gradient in the experiments, but it does not disprove it either. Therefore, while it wasn't proven in the present research, the hypothesis is that the degree to which multiscattering is important is the most likely explanation for the presence or absence of the gradient in the mean intensity.

7

Conclusions and Recommendations

7.1. Conclusions

The interferometric scattering of light by multiple dynamic spherical particles was studied numerically. The ultimate goal of such a study is to study the light scattering by red blood cells in complex artery geometry flows. Ultimately, we wish to invert an in-vivo measured speckle pattern and thereby extract cardiac and hematological parameters of a patient.

In the present work, a code was developed to study the speckle pattern resulting from the scattering of an incoming plane wave by an ensemble of spherical particles (see Chap. 3). This code was developed from scratch in the Fortran programming language, and then linked together with an existing computational fluid dynamics solver, OpenFOAM, using the Python programming language. Afterwards some proof-of-principle simulations were performed, but follow-up is required to generate more quantitative results. Another student, G. Loozen [33], used an experimental approach to study the same phenomenon, which allows for comparison.

The Fluids code evolves the spherical particles as a function of time. The Optics code takes instantaneous snapshots (the speed of light is infinite compared to the fluid velocity) and computes the resulting speckle pattern. By doing so in rapid succession, the finite integration time of a camera could be mimicked (here, $100\mu s$), resulting in blurred speckle patterns. The degree of blurring is directly correlated with the velocity of the particles, and can be made quantifiable using the "speckle contrast". By repeating this process using a frequency of about $50Hz$, a pulsatile flow can be reconstructed (e.g., a heartbeat). That is the idea, but the results generated in the present work did not yet perform this last step.

The Optics code makes use of an exact solution for the scattering by a single sphere, Mie theory, and then scatters the light from each particle to each camera pixel. Interferometric scattering has been achieved by careful bookkeeping of the phases. Multiscattering has been implemented by not only letting particles scatter the incoming wave to the camera, but as well to each, other particle. By continuing this process iteratively until convergence, multiscattering has been taken into account.

The first validation performed is the double slit experiment in Chap. 4. Many spherical particles were placed in two lines, which essentially formed two apertures. Then the intensity profile at a distant camera was observed. A diffraction pattern was found. By comparing the distance between the extrema of the diffraction pattern with the analytical values from the Fraunhofer approximation, a close match was found. Interestingly, an Airy pattern was found superimposed on the diffraction pattern, which had a size of the order of magnitude one would expect from a spherical aperture of the size of our particles. This validation showed that the interferometric part of our code was behaving as expected.

As a second validation, the experimental set-up of Loozen was mimicked in Chap. 5. The main difference between the simulations and the experiments is the fact that a gradient in the mean intensity was observed in the simulations, which was absent in the experiments. This gradient turned out to be caused by the dependence of the amplitude scattering matrix (as is seen in Mie

theory) on the scattering angle. In Chap. 6 it was studied why this gradient is absent in the experiments. Two plausible causes have been investigated: a non-uniform particle size distribution, and the importance of multiscattering. It was concluded that the particle size distribution alone could not be used to explain the absence of the gradient. Multiscattering turned out to be negligible with the simulated particle concentration, which was a factor 1000 lower than in the experiments. Therefore, it wasn't proven that multiscattering is the cause, but it became very likely that it is indeed the cause.

In Chap. 5 the required temporal and spatial resolution have as well been studied. Unfortunately, the case studied had a far too coarse mesh, which was taken this way based on an estimation of the required size. Regardless, qualitatively an expected behaviour was found. If the speckle pattern was corrected for the above-mentioned gradient in the mean intensity, a speckle contrast close to $C = 1$ was found, which is the value one would expect for an instantaneous fully-developed speckle pattern. As a consequence of applying the time integration, the speckles would blur, reducing the speckle contrast. This too is the expected behaviour. The value of $C = 0.93$ obtained does, however, not agree with the experiments, which found $C = 0.75$ (although this value did depend strongly on the precise viewing angle). But given the fact that the mesh was far too coarse, this does not matter.

Finally, a small study was performed to properly determine the required viewing angle to fully resolve the speckles. It was found that the typical speckle size is about $\alpha = 0.003°$ (see Sec. 5.5.2). This implies that the coarse mesh used in fact $\approx 5 \cdot 10^{-4}$ pixels per speckle. Evidently, this was a poor resolution.

Using the developed code, the described methodology, and the knowledge acquired regarding the required spatial and temporal resolution, it has now become possible to start producing real results. We are fully confident that the methodology of applying Mie theory to the scattering problem is possible, at least for dilute systems.

7.2. Recommendations

7.2.1. Optics Code Improvement

The Optics code may be improved in a dozen of ways. These may roughly be subdivided into 'optimisation' and 'accuracy'. Optimisation improves the performance of the Optics code, e.g. by reducing the runtime. Accuracy is related to the simplifying assumptions made while developing the code.

Optimisation

Firstly, some changes to the code are required to improve the performance of the program.

- **Convergence of Multiscattering.** I must admit that, while multiscattering has been implemented, no convergence criterion has been implemented. The only parameter the user can currently specify is the maximum number of iterations, which is then automatically the number of iterations. In the present work this was no problem, since multiscattering turned out to have a negligible effect (see Sec. 6.2) for the case studied in Chap. 5. Regardless, it has a very high priority to implement this.

- **Fortran column-major, etc.** By the way memory is accessed on a computer, it matters for multidimensional arrays in which order its elements are being accessed. For the Fortran programming language, this is 'column-major' ordering. I did take care of this feature very carefully, but there are many others which I did not carefully take care of. These optimisations may still be performed, but only when it is certain that that specific part of the code behaves as expected. To quote Knuth [37]: "We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil.". In other words, the Optics code is currently still being developed, and optimisations shouldn't be done before the development stage has passed.

- **Profiling.** In order to determine the slowest part of the algorithm, a profiler may be used. A profiler is able to pinpoint how much time the program spends in any routine. Without performing a detailed study, it was found that for a quick test case the program spent over

75% of its time in the bhmie subroutine, and 13% of its time in the SMatrix2EField subroutine. About 3% of its time was spent in modules relating to printing debug messages. Whereas this was performed on a simulation that lasted only one minute, and thus may not be generalised to simulations that last 24 hours, it provides a first insight in what subroutine needs to be optimised. The bhmie subroutine is in fact the subroutine taken from the book of Bohren & Huffman [10], of which they said it was written for enhanced readability, not for optimal performance.

Secondly, by neglecting negligible terms or making small numerical approximations by means of linearisation, the program may be sped up considerably.

- **A Connection Matrix.** Currently all particles scatter via all particles to all particles, irrespective of how close they are. Now clearly, if the via-particle is very far away compared with the other two particles, the scattered field will be negligible compared to a via-particle which is much closer. If this is the case, there is no reason to compute this term, especially not for consecutive multiscattering iterations. It is then quickly seen that the time the program spends in the SMatrix2EField subroutine (which was 13% in the item 'profiling' above) can be greatly reduced. Better yet, if we know that the interaction of three given particles is negligible, then there is no reason to compute the amplitude scattering matrix, $[S]$, either. If so, the time the program spends in the bhmie subroutine may be greatly reduced as well (which was over 77% above).
  However, it should be noted that this is a very simplistic argument. In reality the distance between the particles alone does not determine whether a term is negligible. It is in fact the value of
  $$\eta = \frac{[S]_{ijl}}{k^2 r_{ij} r_{jl}}, \tag{7.1}$$
  that matters. In this equation, the light scatters from particle i via particle j to particle l, r is the interparticle distance and $[S]_{ijl}$ is the amplitude scattering matrix with as its argument the angle from i via j to l. Now it can be seen that the earlier discussion was indeed simplistic since, judging from Fig. 3.4, the value of $[S]$ may differ by almost a factor 100 between its minimum and maximum value for the case studied in this thesis.
  Since this criterion depends on both the scattering angle (in a very complex manner, via $[S]$) and the interparticle distance, arguably the best method to neglect negligible terms is by introducing a boolean connection matrix. As a first step towards improving the code, it could be filled with only 1's, and if any computed scattered field is, say, a factor 10^2 lower than the mean of the scattered fields, the element of the connection matrix connecting the three spheres could be set to '0'. Then, the code may use this connection matrix in combination with an if-condition to determine whether or not to compute a certain term. The detailed implementation needs to be given further thought.
- **Interpolate the Amplitude Scattering Matrix.** If the bhmie subroutine is indeed the bottleneck for the speed limit, and it can no longer be further optimised, then we should be clever about computing the amplitude scattering matrix, $[S]$. From Fig. 3.4 it is seen that, at least for the parameters at hand, it is a rapidly fluctuating function. If we were to compute $[S]$ with a sufficient accuracy, then interpolation to angles in which $[S]$ is not known, may yield a reasonably accurate result. Fig. 3.4 was created using 1001 different scattering angles. In the present code, $[S]$ is computed for $O\left(N^2(M+N)\right)$ different angles (see Sec. 3.1.4), which for $N = 1000$ particles and $M = 128\text{x}128$ pixels is of $O\left(10^{10}\right)$. At least for scattering by spherical particles, and especially given the other approximations made (e.g. the far-field assumptions), I am sure no person will debate that with much less than 10^{10} points, the amplitude scattering matrix may be accurately interpolated. Then, it is not hard to imagine that bhmie needs not be optimised at all, since the number of calls to it will become negligible to the total computational time.
  Better yet, provided that less than $O(N \cdot \max(N^2, M))$ terms are used for the interpolation, this may potentially greatly reduce the RAM requirement of the program as well, which would allow for simulating more than $N = 1000$ particles. Memory-wise, 10^6 particles should then be easily achievable. In terms of computational power, this may however still be troublesome.

- **Approximate the Amplitude Scattering Matrix for the camera.** The most time consuming part of computing $[S]$ is for $M > N^2$ related to scattering to every camera pixel. Now, if the camera is very far away compared to the spheres, then perhaps the same value for $[S]_{ijc}$ may be taken for each (i,j)-tuple[1], which reduces the cost by $O(N^2)$. However, taking Fig. 3.4 as the representative shape of $[S]$ for the physical parameters which are of interest to us, it follows that the viewing angle of the camera must be $\ll 1°$. This condition is likely too limiting to yield true. Nevertheless, interpolation using $\ll N^2$ values could still make this method feasible.

Accuracy

Relating to the accuracy of the computed intensity field, we must return to the assumptions made, which were all discussed in Sec. 3.1.6.

- **Far-Field.** A Far-Field (FF) assumption was made, in which it was assumed that when an incoming Plane Wave (PW) scatters by a particle, by the time the scattered light reaches a second particle it may be considered as a spherical wave. This allowed for defining the amplitude scattering matrix, $[S]$, as a function of solely the zenith scattering angle, θ_s. If we wish not to make this assumption, $[S]$ will become dependent on the travelled distance, $\imath$, as well.
  So far, it is not elegant, but still feasible. Then it would however be required to scatter that scattered light by a second sphere, for which a generalised Mie theory would be required, in which the incoming wave needs not be a PW.
  For now, this should have a very low priority compared with the other recommendations, unless evidence is found that this is crucial for the solution.

- **Very-Far-Field I.** On top of the FF assumption, a very-FF assumption was made. It was assumed that the interparticle distance is big enough to assume that when a PW scatters on a spherical particle, the scattered field may again be considered as a PW. Using the derivations of Sec. 3.1.6, and using typical interparticle distances in actual blood (which are of the order of the size of a RBC), this is not satisfied.
  It is possible to drop this assumption by expanding the Vector Spherical Harmonics (VSHs) as seen in Mie theory (2.16 and 2.17) in terms of an infinite sum of PWs, and then truncating the series. This is mathematically difficult to do, but a derivation is given in e.g. Stratton [9]. Evidently, the computational cost will go up by a constant factor proportional to the cut-off index.
  It should be noted that, even if it is important for the real problem, this may still give a negligible difference for the simulation, i.e., if a far more dilute system is being simulated due to the maximum number of particles being restricted, then these simulated particles may still be in each other's FF. Hence, the FF assumption would be completely valid for these simulated particles, despite the physical particles they represent being too dense for this assumption to be valid.

- **Very-Far-Field II.** Instead of dropping the very-FF assumption, we should first study its importance. This may be achieved by considering a single sphere and the full Mie solution, i.e., without taking the asymptotic limit for the Hankel functions (2.25). Then it may be studied at what distance this asymptotic limit becomes sufficiently accurate, because the current criterion (3.22) is to be expected to be too strong.

- **Multiscattering.** Currently it was assumed that the interaction between particles is sufficiently weak for the light waves to be described as independent waves. These waves 'bounce' between particles (which is the multiscattering-process), regardless of whether there is a third particle between the source particle and the target particle.
  As of present, it is not known how valid this method is. We can however reason that there is no problem, i.e., in Mie theory the field outside a given particle is written as the sum of the original field and the scattered field (2.9) for any given incident field. Therefore, the 'error' made by ignoring intermediate particles for the p'th scattering order field, is in fact no error.

[1] Note that this will also remove the observed gradient in the mean intensity of Sec. 5.3.

That error is simply the scattered wave of the $(p+1)$'th scattering order and is thus taken into account.

- **Particle Shape.** One of the goals we eventually want to achieve, is to study cardiac and morphologic parameters of blood. Evidently, it won't be possible to study the morphologic properties of RBCs with a code that only considers a spherical shape. Therefore, non-spherical particles should be looked in to. See for example Steinke and Shepherd [23]. This is most easily performed by comparing the scattered profile of a single sphere computed by our code, and compare it with other (expensive) methods, like the FDTD method, which were applied to the biconcave shape of a RBC, in e.g. He et. al. [3].
  This currently has a low priority. It is more important to get the code to work somewhat better for spherical particles than it currently does, before moving on to non-spherical particles.

7.2.2. Fluids Code Improvement

The Fluids code may be improved specifically relating to modelling blood accurately. This should be looked at in terms of the combined physics: What effect does a certain Fluid simplification have on the resulting speckle pattern? Because even if a certain fluid simplification is relevant to e.g. the local velocity in the bend of a bending artery, it may have a negligible effect on the resulting speckle pattern.

- **Pulsatile Flow.** Whereas at the present stage the improvement of the Optics code has the highest priority, one part of the Fluid Dynamics part of the problem has a serious advantage if it is performed in the near future: pulsatile/transient flow. At the present stage, we suspect that the precise value of the speckle contrast does not matter to study certain cardiac parameters, i.e., to study a heartbeat its time-dependency matters, but not its precise value. This way, with relatively little effort, a temporal validation may be performed.
- **Non-Newtonian.** Currently blood (and the water-glycerol mixture of the experiment) is being modelled as a Newtonian fluid in which one-way coupled spheres are suspended. For actual blood, this is wrong. It results in a different velocity profile than would be the case for a non-Newtonian model of blood. It should be investigated what kind of effect this has on the resulting speckle pattern. Or rather, how the speckles depend on time.
- **Radial Particle Distribution.** A measured radial particle distribution was obtained from the literature (see Secs. 2.2 and 3.2.1) and imposed on the particles. It will be interesting to see what kind of effect this radial particle distribution has on the resulting speckle pattern, e.g. by comparing it to a uniform radial distribution in which more particles are moving nearby the walls at a lower velocity than those particles near the centerline. This will immediately provide some insights in what one may expect near bends in arteries, where the radial profile does too change.
- **Particle Shape.** Despite the Optics code still looking at spherical particles, the shape of the particles does as well affect the Fluid Dynamics. This phenomenon is probably already captured by an accurate radial particle distribution, as was done in the present research.
- **Complex Artery Geometry.** Eventually, in the far future, it will be interesting to consider flow in a complex artery geometry. As for the Fluid Dynamics is concerned, this is not too difficult to implement, since the Transport Phenomena Group is already able to do so. Despite it being relatively easy to implement, it is still not a high priority, as it will only complicate the problem at the present stage.

7.2.3. Combined Physics: Producing Results

Apart from improving the code, it will be nice to actually execute the code on many more test cases. Currently, only proof-of-principle simulations were performed, since most time of this thesis project was spent on developing the code, rather than using it to produce results.

- **Improve spatial resolution.** The top priority for producing results is to first find the proper spatial resolution required to simulate speckles that are not subgrid features. In a very brief

study in Sec. 5.5.2 it was found that the speckles have a size of about $\alpha = 0.003°$. Since it is not feasible to increase the number of pixels much further (unless the Optics code is first optimised), this can only be done by considering a smaller viewing angle for the camera.

- **Reynolds number dependency (master curve?).** In Loozen's experiments [33], he observed how the speckle contrast changes as a function of Reynolds number. He does so for different camera apertures (viewing angles) and integration times. It will be nice to reproduce these figures, at least for small integration times.
  Judging from the consistent shape of these figures, I'd even go as far as to speculate that there may exist a master curve. In other words, there may exist a predictable rescaling to collapse all his lines on to one master curve. It will be nice to investigate whether such a transformation does indeed exist, and if it does, how it relates to the integration time and viewing angle of the camera. Note that this transformation may not be a simple constant rescaling, since the integration time is strongly correlated with velocity and thus Reynolds number.

- **Does multiscattering indeed remove the gradient in the mean intensity?** In Sec. 5.3, a gradient in the mean intensity was observed, which is absent in the experiments. The cause was found to be the shape of the amplitude scattering matrix. In Chap. 6, it was studied what the cause for the simulated presence or the experimental absence is. Whereas it couldn't be proven using the simulations at hand within the timeframe of this research, it was hypothesised that multiscattering is responsible for this behaviour, because it was found negligible in the simulations. It should be investigated whether this hypothesis is indeed true, by considering a much denser case.

- **Heartbeat.** As was mentioned in item "Pulsatile Flow" of Sec. 7.2.2, it would be nice to study a pulsatile flow and then reconstruct how the forcing depends on time. This may be used to measure a heartbeat. In the present work, it was already found that the speckle pattern would be blurred as a consequence of the finite integration time of the camera, resulting in a decreased speckle contrast. This was found despite the simulations having a poor resolution. I am fully confident that, even with a poor resolution, it should be possible to plot the speckle contrast as a function of time (in a transient flow) and thereby reproducing the time-dependence of the used forcing.

- **Extract other information.** The whole point of considering the full scattered field was to not limit ourselves to a scalar quantity. For example, a pulse oximeter only considers the light absorption, which is a scalar quantity. In the present work we have limited ourselves to the speckle contrast, in line with Loozen's experiments [33], but this too is a scalar quantity. In analogy, if we wish to know a velocity, but only consider a speed, then vital information is lost (in this case, the direction of motion). By considering a scalar quantity while we have a full field of information, information is lost in the same manner. Other metrics must be found to extract more information than the speckle contrast can provide us with.

- **Speckle Size.** Directly linked with the spatial resolution, the speckle size should be studied. This will provide another metric for comparison with the experiments. However, this may be difficult to achieve, due to the complex system of lenses present in Loozen's experiments, and the difference in particle concentration.

A

Mie Theory - Details

This appendix gives some more details to the arguments given in Sec. 2.1, which was the derivation of Mie theory. Consequently, this appendix is not meant to be read as a stand-alone text.

A.1. Why are terms omitted from the expansion?

Instead of writing (2.21-2.24), we can include the ignored terms, like such:

$$\vec{E}_1 = \sum_{n=1}^{\infty} E_n \sum_{m=0}^{n} \left(c_{mn}\vec{M}_{omn}^{(1)} + d_{mn}\vec{N}_{emn}^{(1)} + v_{mn}\vec{N}_{omn}^{(1)} + w_{mn}\vec{M}_{emn}^{(1)} \right), \tag{A.1}$$

$$\vec{H}_1 = \frac{k_1}{\mathrm{i}\omega\mu_1} \sum_{n=1}^{\infty} E_n \sum_{m=0}^{n} \left(c_{mn}\vec{N}_{omn}^{(1)} + d_{mn}\vec{M}_{emn}^{(1)} + v_{mn}\vec{M}_{omn}^{(1)} + w_{mn}\vec{N}_{emn}^{(1)} \right), \tag{A.2}$$

$$\vec{E}_s = \sum_{n=1}^{\infty} E_n \sum_{m=0}^{n} \left(a_{mn}\vec{M}_{omn}^{(3)} + b_{mn}\vec{N}_{emn}^{(3)} + p_{mn}\vec{N}_{omn}^{(3)} + q_{mn}\vec{M}_{emn}^{(3)} \right), \tag{A.3}$$

$$\vec{H}_s = \frac{k_2}{\mathrm{i}\omega\mu_2} \sum_{n=1}^{\infty} E_n \sum_{m=0}^{n} \left(a_{mn}\vec{N}_{omn}^{(3)} + b_{mn}\vec{M}_{emn}^{(3)} + p_{mn}\vec{M}_{omn}^{(3)} + q_{mn}\vec{N}_{emn}^{(3)} \right), \tag{A.4}$$

A.1.1. Extract the Coefficients using Orthogonality

Define the functional inner product of two vectors as:

$$\left\langle \vec{a}, \vec{b} \right\rangle = \int_0^{2\pi} \int_0^{\pi} \vec{a} \cdot \vec{b} \sin\theta \mathrm{d}\theta \mathrm{d}\varphi, \tag{A.5}$$

then it is possible to show that under this inner product all $\vec{M}_{pmn}$ and $\vec{M}_{p'm'n'}$ are orthogonal[1] (same for $\vec{N}$), and all $\vec{M}$ with all $\vec{N}$, i.e., the VSHs are linearly independent if tested using this inner product. This permits the coefficients to be extracted individually, e.g.:

$$\left\langle \vec{E}_s, \vec{M}_{omn} \right\rangle = E_n a_{mn} \left\langle \vec{M}_{omn}^{(3)}, \vec{M}_{omn} \right\rangle, \tag{A.6}$$

where $\vec{M}_{omn}$ may have any Bessel function, since it is merely a constant in the integral of (A.5), as it only depends on the radial coordinate.

It is, however, not trivial to apply this functional inner product to the BCs (2.10-2.13), as the BCs are anisotropic. However, since $\vec{E} \cdot (c_1\hat{\theta} + c_2\hat{\varphi})$ for any $\{c_1, c_2\}$ satifies the tangential BC and

[1] That is, the functional inner product equals $0\ \forall(p \neq p' \cup m \neq m' \cup n \neq n')$. Here, p refers to either e or o.

since $\vec{M} \cdot \hat{r} = 0$, it immediately follows that the functional inner product with $\vec{M}^{\mathrm{no}\rho}$ will satisfy the tangential BC too. That is, c_1 and c_2 above must be the same for both $\vec{E}_1$ and $\vec{E}_2$ to satisfy the BC. This is only the case if the ρ-dependent terms (which are material-dependent!) of $\vec{M}$ are excluded, which we call per definition $\vec{M}^{\mathrm{no}\rho}$. This does not affect the orthogonality, since the ρ-dependent part is merely a constant in the integral of (A.5). So, we may write:

$$\left\langle \vec{E}_1, \vec{M}^{\mathrm{no}\rho}_{emn} \right\rangle_{\rho=y} = E_n w_{mn} \left\langle \vec{M}^{(1)}_{emn}, \vec{M}^{\mathrm{no}\rho}_{emn} \right\rangle_{\rho=y} = E_n q_{mn} \left\langle \vec{M}^{(3)}_{emn}, \vec{M}^{\mathrm{no}\rho}_{emn} \right\rangle_{\rho=x} = \left\langle \vec{E}_s, \vec{M}^{\mathrm{no}\rho}_{emn} \right\rangle_{\rho=x}, \tag{A.7}$$

where $\vec{M}^{(1)}_{emn}$ was chosen for convenience, because the incident field does not possess the term, and $x \equiv k_2 a$ and $y \equiv k_1 a$ are as used in Eq. (2.26) (and below it) and represent the sphere's boundary in ρ-space. Henceforth, the superscript $\mathrm{no}\rho$ will be left out, but it is always present on the VSH with which we take the functional inner product.

$\vec{N}$ is a different story: when applying (A.6) using some $\vec{N}$, $\vec{E} \cdot \hat{r}$ is involved and we cannot 'simply' use the tangential BC (2.12):

$$\left\langle \vec{E}_1, \vec{N}_{omn} \right\rangle_{\rho=y} = E_n v_{mn} \left\langle \vec{N}^{(1)}_{omn}, \vec{N}_{omn} \right\rangle_{\rho=y} \neq E_n p_{mn} \left\langle \vec{N}^{(3)}_{omn}, \vec{N}_{omn} \right\rangle_{\rho=x} = \left\langle \vec{E}_s, \vec{N}_{omn} \right\rangle_{\rho=x}, \tag{A.8}$$

because

$$\left(c_0 \vec{E}_1 \cdot \hat{r} + c_1 \vec{E}_1 \cdot \hat{\theta} + c_2 \vec{E}_1 \cdot \hat{\varphi} \right)\Big|_{\rho=y} \neq \left(c_0 \vec{E}_2 \cdot \hat{r} + c_1 \vec{E}_2 \cdot \hat{\theta} + c_2 \vec{E}_2 \cdot \hat{\varphi} \right)\Big|_{\rho=x},$$

which would only hold if $\epsilon_1 = \epsilon_2$ in (2.10). In Bohren & Huffman [10] this problem is not mentioned anywhere, although the solution is lurking from deep within the derivation. As it turns out, the $\hat{r}$-component of $\vec{N}$ satisfies orthogonality on its own, i.e.,

$$\left\langle \left(\vec{N}_{pmn} \cdot \hat{r} \right) \hat{r}, \vec{N}_{p'm'n'} \right\rangle = 0 \qquad \forall (p \neq p' \cup m \neq m' \cup n \neq n'). \tag{A.9}$$

So, whereas the $\hat{\theta}$ and $\hat{\varphi}$ components need each other, $\hat{r}$ can do its own job. Now, since $\left(\vec{N} \cdot \hat{r} \right) \hat{r}$ satisfies orthogonality alone, it then follows by the linearity of the functional inner product (A.5), that $\vec{N}^r = \vec{N} - \left(\vec{N} \cdot \hat{r} \right) \hat{r}$ must too[2]. $\vec{N}^r$ is a shorthand notation to fix the inequality in Eq. (A.8):

$$\left\langle \vec{E}_1, \vec{N}^r_{omn} \right\rangle_{\rho=y} = E_n v_{mn} \left\langle \vec{N}^{(1)}_{omn}, \vec{N}^r_{omn} \right\rangle_{\rho=y} = E_n p_{mn} \left\langle \vec{N}^{(3)}_{omn}, \vec{N}^r_{omn} \right\rangle_{\rho=x} = \left\langle \vec{E}_s, \vec{N}^r_{omn} \right\rangle_{\rho=x}, \tag{A.10}$$

A.1.2. Find the Coefficients

Now that we have a way to extract individual coefficients, we can do so for all coefficients for both $\vec{E}$ and $\vec{H}$, which will yield exactly enough equations to solve the problem. Since the process is identical for every coefficient, let us solely observe what happens to v_{mn} and p_{mn}. These coefficients belong to the VSH $\vec{N}_{omn}$ (in the equations for $\vec{E}$ (A.1 and A.3)) and is not contained within the incident field (2.19)[3]. We can use Eq. (A.10) and its equivalent for $\vec{H}$:

$$E_n v_{mn} \left\langle \vec{N}^{(1)}_{omn}, \vec{N}^r_{omn} \right\rangle_{\rho=y} = E_n p_{mn} \left\langle \vec{N}^{(3)}_{omn}, \vec{N}^r_{omn} \right\rangle_{\rho=x}, \tag{A.11}$$

$$\frac{k_1}{\mathrm{i}\omega\mu_1} E_n v_{mn} \left\langle \vec{M}^{(1)}_{omn}, \vec{M}_{omn} \right\rangle_{\rho=y} = \frac{k_2}{\mathrm{i}\omega\mu_2} E_n p_{mn} \left\langle \vec{M}^{(3)}_{omn}, \vec{M}_{omn} \right\rangle_{\rho=x}. \tag{A.12}$$

If we now use the form of the VSHs, cf. (2.16 and 2.17), and the definition of the functional inner product (A.5), it follows that the only common factors, are the material-dependent terms:

$$\frac{\left\langle \vec{N}^{(1)}_{omn}, \vec{N}^r_{omn} \right\rangle_{\rho=y}}{\left\langle \vec{N}^{(3)}_{omn}, \vec{N}^r_{omn} \right\rangle_{\rho=x}} = \frac{\psi_n'(y)/y}{\xi_n'(x)/x}, \qquad \frac{\left\langle \vec{M}^{(1)}_{omn}, \vec{M}_{omn} \right\rangle_{\rho=y}}{\left\langle \vec{M}^{(3)}_{omn}, \vec{M}_{omn} \right\rangle_{\rho=x}} = \frac{\psi_n(y)/y}{\xi_n(x)/x}, \tag{A.13}$$

[2] Note that $\vec{N}^r$ is just $\vec{N}$ with its $\hat{r}$-component set to 0. Here, the superscript r stands for 'reduced'.
[3] Finding the non-zero Mie coefficients in which the incident field does contribute is performed in Sec. A.2.

where ψ_n and ξ_n are the Ricatti-Bessel functions as introduced in (2.29 and 2.30). Upon substitution we may then write:

$$\frac{p_{mn}}{v_{mn}} = \frac{x}{y}\frac{\psi_n'(y)}{\xi_n'(x)}, \tag{A.14}$$

$$\frac{p_{mn}}{v_{mn}} = \frac{x}{y}\frac{k_1\mu_2}{k_2\mu_1}\frac{\psi_n(y)}{\xi_n(x)} = \frac{\mu_2}{\mu_1}\frac{\psi_n(y)}{\xi_n(x)} = \frac{x^2}{y^2}\frac{\epsilon_1}{\epsilon_2}\frac{\psi_n(y)}{\xi_n(x)}, \tag{A.15}$$

from which it is seen that, unless we have *very* specific material properties[4], $p_{mn} = v_{mn} = 0$. A similar derivation will yield that $p_{mn} = q_{mn} = v_{mn} = w_{mn} = 0$ and $a_{mn} = b_{mn} = c_{mn} = d_{mn} = 0$ if $m \neq 1$. In other words, all coefficients for the VSHs which are not contained within the incident field are 0. By analogy, if there is no force to hit a string, the string will not start vibrating spontaneously. Physically, we require a source term for things to get into motion. Also, by linear independency, hitting string 1 cannot cause any string other than string 1 to vibrate.

A.2. Derive the Mie coefficients

In the case that $m = 1$, the incident field will provide a source for two VSHs (for every n). Using the notation of the previous section together with the expansions as given in Eqs. (2.21-2.24), the BCs for a_n and c_n read:

$$\left\langle \vec{E}_s, \vec{N}^r_{e1n} \right\rangle_{\rho=x} + \left\langle \vec{E}_i, \vec{N}^r_{e1n} \right\rangle_{\rho=x} = \left\langle \vec{E}_1, \vec{N}^r_{e1n} \right\rangle_{\rho=y}, \tag{A.16}$$

$$\left\langle \vec{H}_s, \vec{M}_{e1n} \right\rangle_{\rho=x} + \left\langle \vec{H}_i, \vec{M}_{e1n} \right\rangle_{\rho=x} = \left\langle \vec{H}_1, \vec{M}_{e1n} \right\rangle_{\rho=y}, \tag{A.17}$$

which after cancelling E_n and i may be reduced to

$$a_n \left\langle \vec{N}^{(3)}_{e1n}, \vec{N}^r_{e1n} \right\rangle_{\rho=x} + d_n \left\langle \vec{N}^{(1)}_{e1n}, \vec{N}^r_{e1n} \right\rangle_{\rho=y} = \left\langle \vec{N}^{(1)}_{e1n}, \vec{N}^r_{e1n} \right\rangle_{\rho=x}, \tag{A.18}$$

$$a_n \frac{k_2}{\mu_2} \left\langle \vec{M}^{(3)}_{e1n}, \vec{M}_{e1n} \right\rangle_{\rho=x} + d_n \frac{k_1}{\mu_1} \left\langle \vec{M}^{(1)}_{e1n}, \vec{M}_{e1n} \right\rangle_{\rho=y} = \frac{k_2}{\mu_2} \left\langle \vec{M}^{(1)}_{e1n}, \vec{M}_{e1n} \right\rangle_{\rho=x}. \tag{A.19}$$

Now, upon applying relations similar to Eq. (A.13), the functional inner products are replaced like such:

$$a_n y \xi_n'(x) + d_n x \psi_n'(y) = y \psi_n'(x), \tag{A.20}$$

$$a_n \mu_1 \xi_n(x) + d_n \mu_2 \psi_n(y) = \mu_1 \psi_n(x), \tag{A.21}$$

where $y = mx$ could be used to write them in the same form as Bohren & Huffman use [10].

Finally, two similar equations may be derived for the coefficients b_n and c_n. Upon solving this system of equations, the Mie coefficients as given in Eqs. (2.31 and 2.32) will follow.

[4] These very specific material properties do not serve a physical problem, since nothing in nature can have an infinite accuracy, which would be required in order to ever satisfy those material properties.

B

Code Implementation - Details

B.1. Optics

B.1.1. Classes / Data structure

Fig. B.1 shows a class UML diagram for the MSFF code (it is effectively identical for the SSFF code).

The classes are primarily a way to store the parameters of the algorithm in a very convenient way. SphereManager provides an interface for the algorithm to access the spheres. Sphere permits easy storage of per-sphere information (see Fig. 3.3 and the surrounding text). Camera holds information about the camera's pixels and accumulates the electric field.

The modules are primarily the computational machinery of the algorithm. main is the starting point of the code, which can be an external code like OpenFOAM. MieAlgorithmFF performs the algorithm as described in Sec. 3.1. bhmie is the algorithm from Bohren & Huffman [10], adapted for the present purposes (see Sec. 3.1.1). IO is responsible for the Input & Output (I/O) part of the code: read the input parameters, read the particle positions and write the intensity to a file.

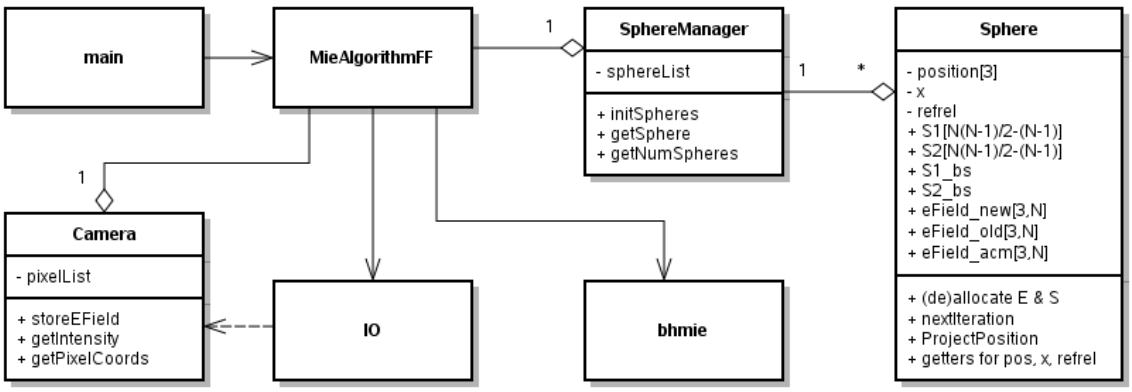


Figure B.1: Class UML diagram for the MSFF code. The code consists of both modulated programming and Object-Oriented Programming (OOP). Modules are shown as just a name. Objects are shown as a class: name, variables, methods. The normal arrows show how modules depend on each other with the source of the arrow being the owner. The striped arrow shows a dependency without ownership.

B.1.2. Pseudocode of the Algorithms

The implemented algorithm may be represented using pseudocode, as shown in Algs. 1-5. Alg. 1 shows the core, top level, of the algorithm, whereas the others show the second level of the algorithm with reference to the appropriate equations. All pseudocodes should speak for themselves.

Algorithm 1: Global structure of the MieAlgorithmFF module. See Fig. 3.2 in particular.

Input: A set of particles $\{\vec{r}, a, m\}$, $\vec{k}_i$, and the camera (size, position, orientation)
Result: The intensity profile on the camera written to a file

1 init(): read input files and instantiate classes
2 Alg. 2: initialScatter(): $p = 1, 2$ scattering order
3 Alg. 3: multiScatter(): $p > 2$ scattering order
4 Alg. 4: scatter2Camera(): scatter the accumulated field (3.8) to the camera, cf. (3.7).
5 output(): compute the resulting intensity (3.9) and write to an output file

Algorithm 2: Global structure of the initialScatter algorithm

Input: SphereManager, $\vec{k}_i$, $\vec{E}_i$
Result: Computed $[S]$ (to be used by 'multiScatter') and updated the $\vec{E}$'s of all spheres

1 **forall the** *particles,* i **do**
2 | Allocate all arrays
3 **end**
4 **forall the** *'via' particles,* i **do**
5 | // Compute the $p = 1$ field:
6 | Compute $\vec{E}^0_{ii} = \vec{E}_i e^{\mathrm{i}kz_i}$, cf. (2.47)
7 | Store it directly into the accumulator $\vec{E}^{\mathrm{accum}}_{ii}$
8 | // Compute the $p = 2$ field:
9 | **forall the** *'target' particles,* $j \neq i$ **do**
10 | | Compute the scattering angle, θ_s, from the incident field via i to j.
11 | **end**
12 | Call bhmie($\{\theta_s\}$) to find all $[S]_{jii}$
13 | **forall the** *'target' particles,* $j \neq i$ **do**
14 | | Use $[S]_{jii}$ to scatter the field: from $\vec{E}^0_{ii}$ to $\vec{E}^1_{ji} = \vec{E}^1_{ji0}$, cf. Alg. 5.
15 | | Accumulate to $\vec{E}^{\mathrm{accum}}_{ji}$ (cf. (3.8)).
16 | **end**
17 | // Prepare for multiscattering:
18 | **forall the** *'target' particles,* $j \neq i$ **do**
19 | | **forall the** *'source' particles,* $l < j \cap l \neq i$ **do**
20 | | | Compute the scattering angle, θ_s, from l via i to j.
21 | | **end**
22 | **end**
23 | Call bhmie($\{\theta_s\}$) to find all $[S]_{jil}$
24 | Call bhmie(π) to find the backscattering matrix $[S]_{\mathrm{bs}} \simeq [S]_{jij}$
25 **end**

Algorithm 3: Global structure of the multiScatter algorithm

Input: SphereManager
Result: Computed all $\vec{E}$'s of all spheres. $\vec{E}_{ji}^{\text{accum}}$ will be scattered to the camera

forall the *scattering orders,* p **do**
  forall the *'via' particles,* i **do**
    Call sphere$_i$.nextIteration()
  end
  forall the *'via' particles,* i **do**
    forall the *'target' particles,* j **do**
      forall the *'source' particles,* l **do**
        Use $[S]_{jil}$ to scatter the field: from $\vec{E}_{il}^{p-1}$ to $\vec{E}_{jil}^{p}$, cf. Alg. 5
        Accumulate to $\vec{E}_{ji}^{p}$ cf. (3.3).
      end
    end
  end
  forall the i, j **do**
    Accumulate to $\vec{E}_{ji}^{\text{accum}}$ cf. (3.8).
  end
  If converged, then break the p-loop. Else continue.
end

Algorithm 4: Global structure of the scatter2Camera algorithm

Input: SphereManager, Camera
Result: The camera now has a measured electric field for every pixel, c

forall the *'via' particles,* i **do**
  forall the *'source' particles,* l **do**
    forall the *camera pixels,* c **do**
      Compute the scattering angle, θ_s, from l via i to c.
    end
  end
  Call bhmie($\{\theta_s\}$) to find all $[S]_{cil}$
  forall the *'source' particles,* l **do**
    forall the *camera pixels,* c **do**
      Use $[S]_{cil}$ to scatter the field: from $\vec{E}_{il}^{\text{accum}}$ to $\vec{E}_{cil}$, cf. Alg. 5
    end
    Call camera.addEField($\vec{E}_{cil}$) to slowly accumulate to $\vec{E}_{c}^{\text{scattered}}$ cf. (3.7)
  end
end
Optionally add $\vec{E}_{c}^{0}$ (which is incident on the camera, but not part of the scattered field)

B.2. Fluids

The version of OpenFOAM used is v2.4.x as obtained from the official github repository May 29th 2015 [38]. In addition, pyFoam and swak4Foam (see Apps. C.1 and C.2) are used. PyFoam was obtained June 9th 2015 using the official subversion repository [39]. Swak4Foam was obtained May 29th 2015 using the official subversion repository [40].

The cylinder, as shown in Fig. 3.7, is used with cyclic boundary conditions. A pressure gradient is applied using the fan BC. On the cylinder wall, a wall BC is applied: no-slip for the velocity and zero-gradient for the pressure.

The pimpleFoam solver has been used, with a maximum Courant number of 0.8. The initial condition for the velocity is set to 80% of u_{max} as predicted by Poiseuille flow (see Sec. 2.3.1), and for the pressure of a uniform value of 0.

Algorithm 5: Global structure of the S2E function

Input: $[S]$, $\vec{E}_i$, $\vec{r}$, $\vec{k}_i$
Output: $\vec{E}_s$
Result: Scattered an incoming PW to a scattered PW, cf. (3.4)
1 Find $\{\hat{E}_{i\parallel}, \hat{E}_{i\perp}, \hat{E}_{s\parallel}\}$ using clever cross products
2 Compute $E_{i\perp}$ and $E_{i\parallel}$ by projection
3 Compute $E_{s\perp}$ and $E_{s\parallel}$, cf. (2.44)
4 Compute $\vec{E}_s = E_{s\perp}\hat{E}_{i\perp} + E_{s\parallel}\hat{E}_{s\parallel}\}$
5 Include the $e^{ikr}/(-ikr)$ term of (2.44)

The mesh uses 15x15 cells for the middle block, 30 cells to resolve the radial part of the four outer blocks and 12 cells to resolve the axial part of the cylinder. Since cyclic BCs were chosen, using only one cell to resolve the axial part gives exactly the same solution, but this wasn't chosen this way in an anticipation for pulsatile flow. No grading of the mesh has been used.

Particles are instantaneously injected at $t = 0$, using randomly generated particle positions cf. the profile shown in Fig. 3.5. The only active force is sphereDrag, as was discussed in Sec. 3.2.2. The particles have a uniform constant size.

B.3. Other Codes

Several other codes were developped using the Python programming language. In this section, these codes are very briefly described from a merely functional perspective. All codes work only from the command line and have a usage message when executed with the '-h' option.

B.3.1. PreProcessing

phi2probdens.py converts a volume distribution to a number distribution, as is required for Fig. 3.5.

genCylParticles.py creates a particlePositions file using randomly generated coordinates using an accumulative radial particle number distribution as its input.

evolveBrownian.py takes a particlePositions file at '$t = 0$' en evolves it over some time T in N discrete steps. The parameter that describes the diffusion process is the difusion coefficient, D, in m^2/s. N particlePosition files are created.

B.3.2. Linking

convertFoam2OpticsParticles.py will take an OpenFOAM case directory and extract all particlePosition files from the file system and convert them to the format which the Optics code uses as its input.

templateSubstitutor.py is a very neat feature, which allows the user to create general files (templates) in which variables are defined, instead of hard-coded numbers and names. This script will then take a case-specific variables file and substitute the variables from the template to create the required file. This script was created to ease the linking process, but is extremely useful for much more purposes.

timeLooper.py is able to loop over particlePosition files and execute the Optics code for each file. This code is deprecated and instead replaced by tiny Shell scripts, which are created specific for each case.

run.sh are these tiny Shell scripts, which have the advantage of very easily running different cases in parallel on multiple machines. These scripts reside in each case's directory, as they are specifically created for certain cases. It is used e.g. as: "nohup nice -n 19 sh -c 'sleep 40m &&

./run.sh 1e-05 2e-05' &" would execute a case for two different times, $t = 1 \cdot 10^{-5}$s and $t = 2 \cdot 10^{-5}$s, after a 40 minutes delay. The mentioned times are ran in series. Executing the command four times for different times executes those times in parallel on four cores w.r.t. the other times. The sleep is required, since the Optics code uses a lot of Random Access Memory (RAM) for a relatively brief moment. Using ssh to connect to a different computer permits using the power of multiple computers.

B.3.3. PostProcessing

computeSpeckleContrast.py takes an intensity file and computes the speckle contrast, as was defined in Sec. 5.4.

convertDataTo2D.py converts the 3D (x, y, z) coordinates to the 2D (a, b)-tuple of the pixel-coordinates files.

timeIntegrateOptics.py is able to take a directory of intensity files with the time uniformly sampled (and no time may be missing). Based on the minimum and maximum time and the number of files it then automatically reasons what files it should take. Specifically, the '-R' option makes it perform the integration using different resolutions, by leaving out some files. E.g., if 101 files are present, the integration may be performed using (excluding the starting time) 100, 50, 25, 20, 10, 5, 4, 2 or 1 times. The result is an integrated intensity file or a directory of integrated intensity files if the '-R' option was specified.

errorBetweenFiles.py takes data files in 1D vector or 2D array format and outputs several scalar measures for the degree with which the data differs. E.g., the root mean square difference between the files.

correctGradient.py takes an intensity file and a 2D pixel-coordinates file and corrects the intensity file by dividing by the gradient of the mean intensity in a given direction. The result will have a zero gradient in its mean intensity in that direction. See Sec. 5.3.

interpolate.py takes an intensity file and a 2D pixel-coordinates file and outputs a new intensity file and a 2D pixel-coordinates file with a different resolution by means of linear interpolation. See Sec. 5.5.2.

meanOfFiles.py Takes a directory with files with their data in the same 1D vector or 2D array format and outputs a file in the same format which consists of the arithmetic mean of those files. See Sec. 6.1.

meanWeightedOfFiles.py Takes a directory with files with their data in the same 1D vector or 2D array format and outputs a file in the same format which consists of a weighted arithmetic mean of those files. It is weighted by first normalising each file before taking the mean. The mean value of the output file is equal to the global mean of the files. See Sec. 6.1.

C

OpenFOAM

Open Source Field Operation And Manipulation (OpenFOAM) is an open source (C++) CFD package which aims at representing differential equations in the code in a form as close to their mathematical tensorial representation as possible. It revolves around a wide variety of applications: solvers and utilities (see Fig. C.1). Solvers are able to solve a set of equations (e.g. a physical problem), whereas utilities assist with pre- and post-processing tools (e.g. meshing). While CFD is my concern, OpenFOAM does as well have functionality for other physical problems, like Computational Solid Mechanics and Electromagnetics. The user works at case-level, editing 'dictionaries', which are read by OpenFOAM; And in theory never needs to see a single line of C++ code.

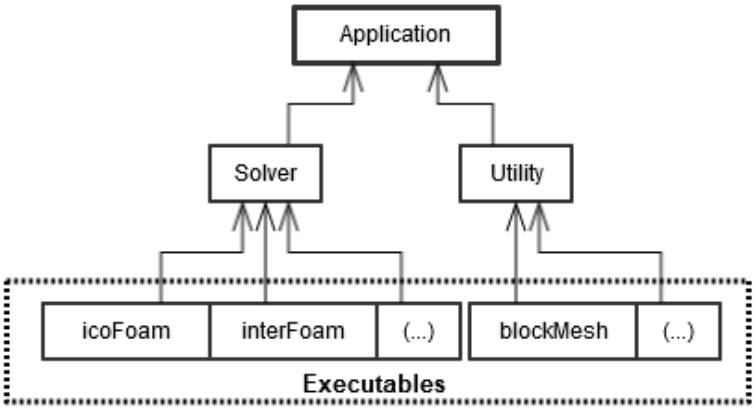


Figure C.1: Application structure of OpenFOAM. OpenFOAM consists of a wide variety of executables (each having their own 'main' method), which share a lot of code (libraries). The latter ensures consistency, maintainability, and eases the process of writing new applications.

The user may develop and execute cases without writing a single line of C++, although some complicated problems may not yet be supported by the standard solvers/libraries of OpenFOAM. There exist third-party tools which greatly broaden the functionality of OpenFOAM, like 'swak4foam' (provides solver functionality) and 'pyFoam' (provides convenience for pre-/postprocessing).

C.1. swak4foam

Swiss Army-Knife for OpenFOAM (swak4foam) is a third-party tool which provides a wide variety of plug-ins for your solver. The creator, Bernhard Gschaider, describes it as: "Like that knife it rarely is the best tool for any given task, but sometimes it is more convenient to get it out of your pocket than going to the tool-shed to get the chain-saw.". It largely consists of parsers, allowing users to write functionality in OpenFOAM's 'dictionary' format, which swak4foam converts to 'FunctionObjects', which are executed by solvers, integrated deep within the core of OpenFOAM

(see Sec. C.3). Consequently, the user can do complicated things (e.g., time-dependent boundary conditions) without writing a single line of C++ code, which saves a lot of time (coding, debugging, validating). A full description of its functionality may be found on its wiki page [40].

C.2. pyFoam

pyFoam is a Python-library, which is able to read and edit OpenFOAM's case files, including the log file. The first functionality is useful, for example, for very easily perform parameter studies using the command line (or shell/python scripts). The second functionality allows the user to look at arbitrary data during runtime (and thereafter) in a graphed form (e.g., the residuals, or the height of a bubble, both as a function of time). A full description of its functionality may be found at its wiki page [39].

C.3. FunctionObject functionality

The 'FunctionObject' feature of OpenFOAM is a very useful feature, allowing the user to manipulate the simulation without writing a custom solver: they are plug-ins which may be used by any existing solver. In its original functionality, it was able to manipulate the fields (e.g., increase the temperature somewhere and/or after a certain time). In its current form, combined with swak4foam and the 'fvOptions' functionality, the user is e.g. able to add source-terms to the equations without editing the equations; Or add particles to the solver, even though the solver does not support particles. (As is used in the present research.) In fact, using 'codedFunctionObject' or 'pythonIntegration', allows the user to insert pieces of code into the simulation, without having to recompile the entire solver and to be re-used in multiple solvers. Fig. C.2 describes qualitatively how OpenFOAM 'comprehends' FunctionObjects.

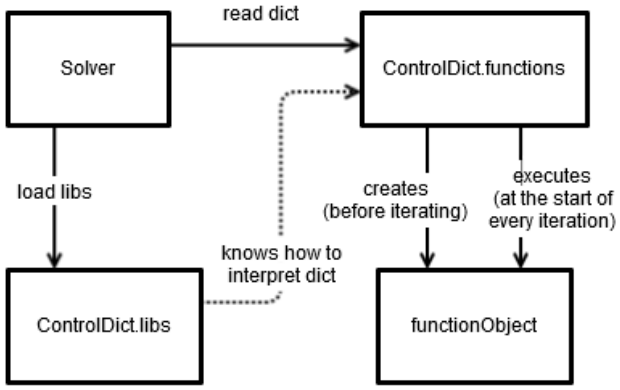


Figure C.2: Diagram showing qualitatively how OpenFOAM's solvers interpret and execute FunctionObjects.

Acronyms

BC Boundary Condition. 4, 6, 61–63, 67, 68

CFD Computational Fluid Dynamics. 13, 15, 23, 26, 27, 71

FF Far-Field. 7, 15, 16, 19, 21, 22, 29–32, 34, 39, 58

I/O Input & Output. 16, 65

IPW Incoming Plane Wave. 18

LPT Lagrangian Particle Tracking. 3, 13, 25

MSFF Multi-Scattering Far-Field. 15, 34, 35, 65

NF Near-Field. 16

OOP Object-Oriented Programming. 18, 65

PW Plane Wave. 4, 5, 8–10, 15–17, 21, 22, 29, 31, 32, 37, 42, 58, 68

RAM Random Access Memory. 18, 19, 46, 57, 69

RBC Red Blood Cell. 10, 15, 21, 24, 25, 42, 58, 59

SSFF Single-Scattering Far-Field. 15, 32–34, 65

VFF Very Far-Field. 16

VSH Vector Spherical Harmonic. 5–7, 58, 61–63

Bibliography

[1] A. M. K. Nilsson, P. Alsholm, A. Karlsson, and S. Andersson-Engels, *T-matrix computations of light scattering by red blood cells,* Appl. Opt. **37**, 2735 (1998).

[2] A. Fercher and J. Briers, *Flow visualization by means of single-exposure speckle photography,* Optics Communications **37**, 326 (1981).

[3] J. He, A. Karlsson, J. Swartling, and S. Andersson-Engels, *Light scattering by multiple red blood cells,* J. Opt. Soc. Am. A **21**, 1953 (2004).

[4] G. Mie, *Beiträge zur Optik trüber Medien, speziell kolloidaler Metallösungen,* Annalen der Physik **4**, 377 (1908).

[5] G. Mie, *Contributions to the optics of turbid media, particularly of colloidal metal solutions,* Royal Aircraft Establishment (1976), translation of [4].

[6] G. Mie, *Contributions to the optics of turbid media, particularly of colloidal metal solutions,* Sandia Laboratories (1978), translation of [4].

[7] T. Wriedt, *Mie theory 1908-2008, introduction to the conference,* in *Present developments and Interdisciplinary aspects of light scattering* (2008).

[8] H. C. van de Hulst, *Light Scattering by Small Particles*, dover ed. (Dover Publications, 1981) (Original work published 1957).

[9] J. A. Stratton, *Electromagnetic Theory* (McGraw-Hill Book Company, 1941).

[10] C. F. Bohren and D. R. Huffman, *Absorption and Scattering of Light by Small Particles*, Wiley Professional Paperback ed. (John Wiley & Sons, Inc., 1998).

[11] P. W. Barber and S. C. Hill, *Light Scatter by Particles: Computational Methods*, Wiley Professional Paperback ed. (World Scientific Publishing Co. Pte. Ltd., 1998) (Original work published 1990).

[12] W. K. Purves, D. Sadava, G. H. Orians, and H. C. Heller, *Life: The Science of Biology*, 7th ed. (Sunderland, Mass: Sinauer Associates, 2004) p. 954.

[13] M. Shmukler, *Density of Blood* (The Physics Handbook, 2004).

[14] J. Cutnell and K. Johnson, *Physics*, 4th ed. (Wiley, 1998) p. 308.

[15] G. Elert, *Viscosity* (The Physics Hypertextbook, 1998-2015).

[16] P. A. M. M. Aarts, S. A. T. van den Broek, G. W. Prins, G. D. C. Kuiken, J. J. Sixma, and R. M. Heethaar, *Blood platelets are concentrated near the wall and red blood cells, in the center in flowing blood,* Arteriosclerosis **8**, 819 (1988).

[17] P. K. Kundu, I. M. Cohen, and D. R. Dowling, *Fluid Mechanics*, 5th ed. (Elsevier, 2012) pp. 110–114.

[18] E. W. Merrill, *Rheology of blood,* Physiological Reviews **49**, 863 (1969).

[19] S. Corssin and J. Lumley, *On the equation of motion for a particle in turbulent fluid,* Appl. Sci. Res. (A) **6**, 114 (1956).

[20] M. R. Maxey and J. J. Riley, *Equation of motion for a small rigid sphere in a nonuniform flow,* Phys. Fluids **26**, 883 (1983).

[21] W. J. Wiscombe, *Mie scattering calculations: advances in technique and fast, vector-speed computer codes*, Tech. Rep. (National Center for Atmospheric Research, 1979).

[22] W. J. Wiscombe, *Improved mie scattering algorithms,* Appl. Opt. **19**, 1505 (1980).

[23] J. M. Steinke and A. P. Shepherd, *Comparison of mie theory and the light scattering of red blood cells,* Appl. Opt. **27**, 4027 (1988).

[24] D. J. Vojir and E. E. Michaelides, *Effect of the history term on the motion of rigid spheres in a viscous fluid,* Int. J. Multiphase Flow **20**, 547 (1994).

[25] S. De Gruttola, K. Boomsma, and D. Poulikakos, *Computational simulation of a non-Newtonian model of the blood separation process,* Artificial Organs **29**, 949 (2005).

[26] C. E. Brennen, *Fundamentals of multiphsae flow*, reprint ed. (Cambridge Univ. Press, 2005).

[27] D. C. C. Stuart, *The Development of a Discrete Particle Model for 3D Unstructured Grids: Application to Magnetic Drug Targeting*, MSc thesis, Delft, University of Technology (2009).

[28] F. Durst, S. Ray, B. Ünsal, and O. A. Bayoumi, *The development lengths of laminar pipe and channel flows,* Journal of Fluids Engineering **127**, 1154 (2005).

[29] E. Peirano and B. Leckner, *Fundamentals of gas-solid flows applied to circulating fluidized bed combustion,* Progress in Energy and Combustion Science **24**, 259 (1996).

[30] R. I. Issa, *Solution of the implicitly discretised fluid flow equations by operator-splitting,* Journal of Computational Physics **62**, 40 (1985).

[31] J. H. Ferziger and M. Peric, *Computational Methods for Fluid Dynamics*, 3rd ed. (Springer-Verslag, 2002).

[32] J. W. Goodman, *Introduction to Fourier Optics*, 2nd ed. (The McGraw-Hill Companies, Inc., 1996).

[33] G. B. Loozen, *Monitoring pulsating flow with dynamic speckle fields.*, MSc thesis, Delft, University of Technology (2015).

[34] M. Draijer, E. Hondebrink, T. van Leeuwen, and W. Steenbergen, *Review of laser speckle contrast techniques for visualizing tissue perfusion,* Lasers Med Sci. **24**, 639 (2009).

[35] J. W. Goodman, *Speckle phenomena in optics: Theory and applications* (Roberts and Company Publishers, 2007).

[36] J. W. Goodman and G. Parry, *Laser speckle and related phenomena* (Springer, Berlin Heidelberg New York, 1984).

[37] D. E. Knuth, *Structured programming with go to statements,* ACM Journal Computing Surveys **6**, 268 (1974).

[38] OpenFOAM-2.4.x, *Github* (OpenFOAM Foundation, Maya 22nd 2015) https://github.com/OpenFOAM/OpenFOAM-2.4.x.

[39] B. Gschaider, *openfoamwiki.net:Contrib/PyFoam* (OpenFOAM Foundation, 2015) http://openfoamwiki.net/index.php/Contrib/PyFoam.

[40] B. Gschaider, *openfoamwiki.net:Contrib/swak4Foam* (OpenFOAM Foundation, 2015) http://openfoamwiki.net/index.php/Contrib/swak4Foam.