GPU accelerated computation of Polarized Subsurface BRDF for Flat Particulate Layers

Charly Collin,[1, *] Sumanta Pattanaik,[1, †] and Kadi Bouatouch[2, ‡]

[1] *University of Central Florida, United States*

[2] *IRISA, Université de Rennes 1, France*

compiled: February 13, 2015

BRDF of most real world materials has two components, the surface BRDF due to the light reflecting at the surface of the material and the subsurface BRDF due to the light entering and going through many scattering events inside the material. Each of these events modifies light's path, power, polarization state. Computing polarized subsurface BRDF of a material requires simulating the light transport inside the material. The transport of polarized light is modeled by the Vector Radiative Transfer Equation (VRTE), an integro-differential equation. Computing solution to that equation is expensive. The Discrete Ordinate Method (DOM) is a common approach to solving the VRTE. Such solvers are very time consuming for complex uses such as BRDF computation, where one must solve VRTE for surface radiance distribution due to light incident from every direction of the hemisphere above the surface. In this paper, we present a GPU based DOM solution of the VRTE to expedite the subsurface BRDF computation. As in other DOM based solutions, our solution is based on Fourier expansions of the phase function and the radiance function. This allows us to independently solve the VRTE for each order of expansion. We take advantage of those repetitions and of the repetitions in each of the sub-steps of the solution process. Our solver is implemented to run mainly on graphics hardware using the OpenCL library and runs up to seven times faster than its CPU equivalent, allowing the computation of subsurface BRDF in a matter of minutes. We compute and present the subsurface BRDF lobes due to powders and paints of a few materials. We also show the rendering of objects with the computed BRDF. The solver is available for public use through the authors' web site.

Keywords: Scattering;polarization; BRDF; Parallel processing; Radiative transfer

1. Introduction

The propagation of light in scattering and absorbing media is modeled by the Radiative Transfer Equation (RTE) [1]. One of the most common RTE solution methods is the Discrete Ordinate Method (DOM), which is the method used in publicly available solvers, such as DISORT [2].

To take into account polarization nature of the light, the vector radiative transfer equation (VRTE), a variant of the RTE, has been proposed. To support polarization, the VRTE incorporates Stokes vector representation of polarized light and Mueller matrix representation of polarized BRDF and phase function. As its scalar equivalent (RTE), the VRTE is an integro-differential equation, but rather than working with scalar values, the VRTE is expressed in terms of Stokes vectors and Mueller matrices [3]. The DOM solution can still be used for the numerical solution of the VRTE [4], the difference being that the vector problem involves complex arithmetic. There are a number of references to such DOM based solver in literature. ARTS [5] and VLIDORT [6] are two well known public domain DOM based VRTE solvers.

Like many numerical solvers, a DOM VRTE solver is computationally expensive. As we will notice in later sections, in DOM based solution the order of computation increases polynomially with the number incident directions. Consequently, for applications involving a large number of incident and outgoing directions, serial implementation of DOM solver can take long computation time (say hours).

In this paper, we describe a GPU based parallel BRDF computation which relies heavily on a DOM solution for VRTE for layered materials. Our solver running on a off the shelf GPU, carries out the computation in much faster (about seven times) than serial version, and hence allows us to compute a BRDF in a few minutes.

An overview of the BRDF computation and light transport solution is given in section 2. Section 3 presents a detailed cost analysis of the VRTE solution. Section 4 describes our GPU solver and presents the computation time as a function of various parameters, follows it with the analysis of the results.

2. Solution

Our goal is to compute the subsurface component of the BRDF $\mathbf{F_r}$. This corresponds to the light entering the material, getting scattered and exiting at the same or at points very close to the point of light entry. This

* Charly Collin: charly.collin@bobbyblues.com

† Sumanta Pattanaik: sumant@cs.ucf.edu

‡ Kadi Bouatouch: kadi.bouatouch@irisa.fr

component satisfies the following equation:

$$\mathbf{F_r}(\mu', \phi', \mu, \phi)\mathbf{E}(\mu', \phi') = \mathbf{I}(\mu, \phi), \tag{1}$$

where μ is the cosine of the zenith angle and ϕ is the azimuthal angle of a direction. In this equation, $\mathbf{E}(\mu', \phi')$ and $\mathbf{I}(\mu, \phi)$ are respectively the Stokes representation of polarized incident irradiance and exiting radiance, and $\mathbf{F_r}$ is the BRDF Mueller matrix. The components of the Stokes vectors $[I, Q, U, V]$ are: I the total radiance, Q the difference between the linearly polarized components of radiance along the horizontal and vertical axis, U the difference between the linearly polarized components at 45 degrees and 135 degrees and V the difference between the right circularly and left circularly polarized components.

Computing $\mathbf{F_r}$ for a pair of incident and outgoing directions (μ', ϕ', μ, ϕ) requires solving the VRTE inside the material to compute the outgoing radiance $\mathbf{I}(\mu, \phi)$ at the surface as a function of incident irradiance from direction (μ', ϕ'). In this section we briefly present the VRTE and its DOM solution for layered homogeneous materials and a method for computing the BRDF Mueller matrix from the VRTE solution.

2.A. VRTE

Polarized light transport in a plane-parallel participating medium is modeled through the VRTE:

$$\mu\frac{\partial}{\partial\tau}\mathbf{I}(\tau, \mu, \phi) + \mathbf{I}(\tau, \mu, \phi) = \mathbf{J}(\tau, \mu, \phi), \tag{2}$$

with τ the optical depth in the medium. $\mathbf{I}$ is the polarized radiance and $\mathbf{J}$ is the source term, representing the scattering contribution and is defined as:

$$\mathbf{J}(\tau, \mu, \phi) = \frac{\omega(\tau)}{4\pi}\int_{-1}^{1}\int_{0}^{2\pi}\mathbf{P}(\tau, \mu, \phi, \mu', \phi')\mathbf{I}(\tau, \mu', \phi')d\phi' d\mu' + \mathbf{Q}(\tau, \mu, \phi), \tag{3}$$

where ω is the single scattering albedo of the medium, $\mathbf{P}$ the phase matrix, and $\mathbf{Q}$ the inhomogeneous source term which represents the direct contribution from the light source and is defined as:

$$\mathbf{Q}(\tau, \mu, \phi) = \frac{\omega(\tau)}{4\pi}\mathbf{P}(\tau, \mu, \phi, \mu_0, \phi_0)\mathbf{I_0}\exp^{-\frac{\tau}{\mu_0}}, \tag{4}$$

with (μ_0, ϕ_0) as the direction towards the light source and $\mathbf{I}_0$ the radiance stokes vector incident from the light source at the top layer.

For layered materials represented by the superposition of homogeneous layers, the phase function and single scattering albedo are unique in each layer and do not depend on τ anymore. If we further assume that the layer is particulated and is composed of spherical particles or particles with random azimuthal orientation, then the phase matrix becomes a function of the scattering angle θ defined for a pair of incident direction (μ, ϕ) and outgoing direction (μ', ϕ') as:

$$\theta = \mu\mu' + \sqrt{1-\mu^2}\sqrt{1-\mu'^2}\cos(\phi - \phi'). \tag{5}$$

2.B. Azimuthal separation

Because the phase matrix is a function of the scattering angle, its Fourier decomposition can be written as [7]:

$$\mathbf{P}(\mu, \phi, \mu', \phi') = \sum_{m=0}^{L-1}\sum_{k=1}^{2}\Phi_k^m(\phi' - \phi)\mathbf{A}^m(\mu, \mu')\mathbf{D}_k, \tag{6}$$

$$\mathbf{\Phi}_1^m(\phi) = (2 - \delta_{0,m})diag(\cos m\phi, \cos m\phi, \sin m\phi, \sin m\phi), \tag{7}$$
$$\mathbf{\Phi}_2^m(\phi) = (2 - \delta_{0,m})diag(-\sin m\phi, -\sin m\phi, \cos m\phi, \cos m\phi),$$
$$\mathbf{D}_1 = diag(1, 1, 0, 0), \tag{8}$$
$$\mathbf{D}_2 = diag(0, 0, 1, 1),$$

where the $\mathbf{A}^m$'s are 4×4 matrices and are defined as follows [8]:

$$\mathbf{A}^m(\mu, \mu') = \sum_{l=m}^{L-1}\mathbf{P}_l^m(\mu)\mathbf{B}_l\mathbf{P}_l^m(\mu'). \tag{9}$$

Here, $\mathbf{B}_l$ matrices contains the expansion coefficients that define the scattering function and the $\mathbf{P}_l^m$ matrices are the associated Legendre matrices satisfying [8]:

$$\mathbf{P}_l^m(-\mu) = (-1)^{l-m}\mathbf{D}\mathbf{P}_l^m(\mu)\mathbf{D}, \tag{10}$$
$$\mathbf{D} = diag(1, 1, -1, -1). \tag{11}$$

Similarly, the Stokes vectors can be expanded as a Fourier series:

$$\mathbf{I}(\tau, \mu, \phi) = \frac{1}{2}\sum_{m=0}^{L-1}\sum_{k=1}^{2}\mathbf{\Phi}_k^m(\phi - \phi_0)\mathbf{I}_k^m(\tau, \mu). \tag{12}$$

Hence, the VRTE can be expanded as a set of equations:

$$\mu\frac{\partial}{\partial\tau}\mathbf{I}_k^m(\tau, \mu) = -\mathbf{I}_k^m(\tau, \mu) + \frac{\omega}{2}\int_{-1}^{1}\mathbf{A}^m(\mu, \mu')\mathbf{I}_k^m(\tau, \mu')d\mu' + \mathbf{Q}_k^m(\tau, \mu), \tag{13}$$

where $m \in \{0, L-1\}$, $k \in \{1, 2\}$ and:

$$\mathbf{Q}_k^m(\tau, \mu) = \frac{\omega}{4\pi}\mathbf{A}^m(\mu, \mu_0)\mathbf{D}_k\mathbf{I}_0\exp^{-\frac{\tau}{\mu_0}}. \tag{14}$$

Using DOM approach, the equation is converted into a discrete set of equations by discretizing μ using a double Gauss quadrature as follows [9]:

$$\mu\frac{\partial}{\partial\tau}\mathbf{I}_k^m(\tau, \mu_i) = -\mathbf{I}_k^m(\tau, \mu_i) + \frac{\omega}{2}\sum_{n=-N}^{N}\alpha_n\mathbf{A}^m(\mu_i, \mu_n)\mathbf{I}_k^m(\tau, \mu_n) + \mathbf{Q}_k^m(\tau, \mu_i), \tag{15}$$

where N the quadrature size, α_n are the quadrature weights, and μ_i, μ_n are the quadrature nodes. In the rest of this paper, we will assume quadrature nodes are always positive, and will differentiate upward from downward directions by using μ_n and $-\mu_n$ respectively.

2.C. Solution

To compute the exiting radiance field at any optical depth τ, we must solve the equation 15 for all m and k values. This equation is a first order differential equation and hence its solution can be written as the sum of the homogeneous solution and a particular solution. In the case of multiple layer materials, a solution has to be computed for each layer. For simplicity, the homogeneous and particular solutions are presented here for a single layer only. To simplify the equations furthermore, we drop the scripts k and m in the following equations, as the solution presented here is valid for all order of expansion.

2.C.1. Homogeneous solution

The homogeneous solution is the solution to the VRTE where the inhomogeneous source term $\mathbf{Q}$ is zero:

$$\begin{aligned}\pm\mu_i \frac{\partial}{\partial \tau}\mathbf{I}(\tau, \pm\mu_i) = &- \mathbf{I}(\tau, \pm\mu_i) \qquad (16)\\ &+ \frac{\omega}{2}\sum_{n=1}^{N} \alpha_n \mathbf{A}(\pm\mu_i, \mu_n)\mathbf{I}(\tau, \mu_n)\\ &+ \frac{\omega}{2}\sum_{n=1}^{N} \alpha_n \mathbf{A}(\pm\mu_i, -\mu_n)\mathbf{I}(\tau, -\mu_n).\end{aligned}$$

Such equations are known to have exponential solutions, so $\mathbf{I}$ can be substituted in 16 as:

$$\mathbf{I}(\tau, \pm\mu_i) = \mathbf{\Phi}(\nu, \pm\mu_i) \exp^{-\tau/\nu} . \qquad (17)$$

This leads to the following set of homogeneous equations:

$$\begin{aligned}-\frac{\pm\mu_i}{\nu}\mathbf{\Phi}(\nu, \pm\mu_i) = &\sum_{n=1}^{N}\left(\frac{\omega}{2}\alpha_n \mathbf{A}(\pm\mu_i, \mu_n) - \delta_{i,n}\right)\mathbf{\Phi}(\nu, \mu_n) \qquad (18)\\ &+ \sum_{n=1}^{N}\left(\frac{\omega}{2}\alpha_n \mathbf{A}(\pm\mu_i, -\mu_n) - \delta_{i,n}\right)\mathbf{\Phi}(\nu, -\mu_n).\end{aligned}$$

This set of equations can be written as a matrix operation using the following $4N$ vectors

$$\mathbf{\Phi}^{\pm}(\nu) = [\mathbf{\Phi}^T(\nu, \pm\mu_1), \mathbf{\Phi}^T(\nu, \pm\mu_2), \cdots, \mathbf{\Phi}^T(\nu, \pm\mu_N)]^T,$$

as

$$\mathbf{M}^{-1}\left(\frac{\omega}{2}\mathbf{W}\mathcal{A} - \mathbf{I}_{8N}\right)\begin{bmatrix}\mathbf{\Phi}^+(\nu)\\ \mathbf{\Phi}^-(\nu)\end{bmatrix} = -\frac{1}{\nu}\begin{bmatrix}\mathbf{\Phi}^+(\nu)\\ \mathbf{\Phi}^-(\nu)\end{bmatrix}, \qquad (19)$$

where $\mathbf{I}_{8N}$ is the identity matrix of size $8N$, and

$$\begin{aligned}\mathbf{M} &= \mathit{diag}(\mu_1\mathbf{I_4}, \cdots, \mu_N\mathbf{I_4}, -\mu_1\mathbf{I_4}, \cdots, -\mu_N\mathbf{I_4}), \quad (20)\\ \mathbf{W} &= \mathit{diag}(\alpha_1\mathbf{I_4}, \ldots, \alpha_N\mathbf{I_4}, \alpha_1\mathbf{I_4}, \ldots, \alpha_N\mathbf{I_4}). \quad (21)\end{aligned}$$

$\mathcal{A}$ in the above equation is a $8N \times 8N$ matrix defined as follow:

$$\mathcal{A} = \begin{bmatrix}\mathcal{A}^1 & \mathcal{A}^2\\ \mathcal{A}^3 & \mathcal{A}^4\end{bmatrix}, \qquad (22)$$

where each subblock of the $4N \times 4N$ $\mathcal{A}^i$ matrices are:

$$\begin{aligned}\mathcal{A}^1_{i,j} = \mathbf{A}(\mu_i, \mu_j), \mathcal{A}^2_{i,j} = \mathbf{A}(-\mu_i, \mu_j),\ \mathcal{A}^3_{i,j} = \mathbf{A}(\mu_i, -\mu_j),\\ \mathcal{A}^4_{i,j} = \mathbf{A}(-\mu_i, -\mu_j). \qquad (23)\end{aligned}$$

The matrix form of the homogeneous equation 19 represents an eigenproblem, the solution to which yields $4N$ eigenvalues ν_j and eigenvectors $\mathbf{\Phi}^{\pm}_j$. The homogeneous solution is then expressed as a linear combination of those solutions:

$$\mathbf{I}^h_+(\tau) = \sum_{j=1}^{4N} A_j \mathbf{\Phi}^+(\nu_j) \exp^{\frac{-\tau}{\nu_j}} + B_j \mathbf{\Phi}^-(\nu_j) \exp^{\frac{-(\tau_0-\tau)}{\nu_j}}, \qquad (24)$$

$$\mathbf{I}^h_-(\tau) = \mathbf{\Delta}\sum_{j=1}^{4N} A_j \mathbf{\Phi}^-(\nu_j) \exp^{\frac{-\tau}{\nu_j}} + B_j \mathbf{\Phi}^+(\nu_j) \exp^{\frac{-(\tau_0-\tau)}{\nu_j}}, \qquad (25)$$

where

$$\begin{aligned}\mathbf{I}^h_{\pm}(\tau) &= [\mathbf{I}^T(\tau, \pm\mu_1), \mathbf{I}^T(\tau, \pm\mu_2), \cdots, \mathbf{I}^T(\tau, \pm\mu_N)], \quad (26)\\ \mathbf{\Delta} &= \mathit{diag}(\mathbf{D}, \mathbf{D}, \cdots, \mathbf{D}), \quad (27)\end{aligned}$$

and where τ_0 is the full thickness of the medium. The computation of the unknowns A_j's and B_j's in Equations 24 and 25 will be discussed later.

2.C.2. Particular solution

A particular solution to VRTE equation 15 can be found by looking for a solution $\mathbf{I}$ similar to the inhomogeneous term (14). This latter has the form:

$$\mathbf{Q}(\tau, \mu) = \mathbf{X}(\mu) \exp^{\frac{-\tau}{\mu_0}} . \qquad (28)$$

So we look for a solution which can be expressed as:

$$\mathbf{I}^p(\tau, \mu) = \mathbf{Z}(\mu) \exp^{\frac{-\tau}{\mu_0}} . \qquad (29)$$

Substituting (29) in (15) yields the following set of particular equations:

$$\begin{aligned}-\frac{\pm\mu_i}{\mu_0}\mathbf{Z}(\pm\mu_i) = &-\mathbf{Z}(\pm\mu_i)\\ &+ \frac{\omega}{2}\sum_{n=1}^{N} \alpha_n \mathbf{A}(\pm\mu_i, \mu_n)\mathbf{Z}(\mu_n)\\ &+ \frac{\omega}{2}\sum_{n=1}^{N} \alpha_n \mathbf{A}(\pm\mu_i, -\mu_n)\mathbf{Z}(-\mu_n)\\ &+ \mathbf{X}(\pm\mu_i). \qquad (30)\end{aligned}$$

Introducing the $\mathbf{Z}^{\pm}$ and $\mathbf{X}^{\pm}$ vectors:

$$\begin{aligned}\mathbf{Z}^{\pm} &= [\mathbf{Z}^T(\pm\mu_1), \mathbf{Z}^T(\pm\mu_2), \ldots, \mathbf{Z}^T(\pm\mu_N)]^T, \quad (31)\\ \mathbf{X}^{\pm} &= [\mathbf{X}^T(\pm\mu_1), \mathbf{X}^T(\pm\mu_2), \ldots, \mathbf{X}^T(\pm\mu_N)]^T, \quad (32)\end{aligned}$$

This set of equation can be written as:

$$\left(-\frac{1}{\mu_0}\mathbf{M} + \mathbf{I}_{8N} - \frac{\omega}{2}\mathbf{W}\mathcal{A}\right)\begin{bmatrix}\mathbf{Z}^+\\ \mathbf{Z}^-\end{bmatrix} = \begin{bmatrix}\mathbf{X}^+\\ \mathbf{X}^-\end{bmatrix}. \qquad (33)$$

Once solved, equation 33 yields the $\mathbf{Z}^{\pm}$ vectors and the particular solution.

2.C.3. Boundary conditions

Having both homogeneous and particular solutions, the radiance field can be computed at any depth τ in the layer as:

$$\mathbf{I}(\tau,\mu_i)=\mathbf{I}^h(\tau,\mu_i)+\mathbf{I}^p(\tau,\mu_i). \tag{34}$$

However the homogeneous solution still contains $8N$ unknown A_j's and B_j's. These unknowns can be computed from $8N$ equations with known $\mathbf{I}$ values. The boundary condition of the layer gives us those known values.

- Light is incident at the top of the material layer from a single direction, which has already been accounted for as the inhomogeneous term. Thus $\mathbf{I}(0,\mu)=0$ when $\mu<0$.
- The radiance field at the bottom of the medium is due to the reflection of incident radiance field at the bottom boundary and is governed by $\mathbf{R}$, the base material BRDF:

$$\mathbf{I}(\tau_0,\mu)=\int_0^1 \mathbf{R}(\mu,-\mu')\mu'\mathbf{I}(\tau_0,-\mu')d\mu' +\mathbf{R}(\mu,-\mu_0)\frac{\mu_0}{2\pi}\mathbf{I}_0\exp^{-\tau_0/\mu_0}. \tag{35}$$

  The second term on the righthand side of the equation is due to the reflection of the attenuated incident radiance.
  For a simplifying situation where there is no reflection from the base $\mathbf{I}(\tau_0,\mu)=0$.

Each of those boundary conditions yields a set of $4N$ linear equations. The unknowns A's and B's can be computed by solving this linear system of $8N$ equations. In the case of multiple layer materials, an homogeneous solution is computed at each layer. Each extra layer adds an extra $8N$ unknown combination factors. To compute those we use the fact that the radiance field is continuous between two layers, so at each layer boundary we have a set of $8N$ linear equations.

2.C.4. Reconstruction

Equation 34 gives the solution only at the quadrature angles. For arbitrary angles, the solution to the VRTE needs to be reconstructed. This is achieved by following the Source Function Integration technique [1][10] which yields for any positive μ value:

$$\mathbf{I}(\tau,\mu)=\mathbf{I}(\tau_0,\mu)\exp^{-(\tau_0-\tau)/\mu} + \int_\tau^{\tau_0}\mathbf{S}(t,\mu)\exp^{-(t-\tau)/\mu}dt, \tag{36}$$

$$\mathbf{I}(\tau,-\mu)=\int_\tau^0 \mathbf{S}(t,-\mu)\exp^{-(t-\tau)/\mu}dt, \tag{37}$$

where:

$$\mathbf{S}(\tau,\mu)=\frac{\omega}{2}\sum_{i=-N}^{N}\alpha_n\mathbf{A}(\mu,\mu_i)\mathbf{I}(\tau,\mu_i) + \mathbf{Q}(\tau,\mu). \tag{38}$$

The integral terms in equations 36, 37 have a close form solution as all the terms are exponential functions of τ.

2.D. BRDF Computation

We wish to compute the BRDF Mueller matrix as defined in equation 1. For a given pair of directions (μ,μ_0), solving equation 15 results in a single Stokes Vector. To compute the 16 unknowns of the BRDF Mueller matrix, we need solutions to several incident irradiance Stokes vectors $\mathbf{E}_{inc}$:

$$\mathbf{F_r}(\mu,\mu',\phi-\phi')=[\mathbf{I}^1(0,\mu',\phi'),\ldots,\mathbf{I}^n(0,\mu',\phi')] \times[\mathbf{E}^1_{inc}(\mu,\phi),\ldots,\mathbf{E}^n_{inc}(\mu,\phi)]^{-1}, \tag{39}$$

where n is at least 4 when the incident Stokes vectors are linearly independent.

3. Computation cost

In this section we analyze the computation cost associated with each step of the VRTE solution, and identify the possible GPU parallelization steps.

3.A. Homogeneous solution

The homogeneous solution is the first step of the VRTE solution. As it is independent of the incident radiance, it needs to be computed only once per order of expansion. Homogeneous solution involves solving an eigenproblem. As can be seem from equation 19, the problem has a size of $8N\times 8N$. Because of the symmetry of the problem, the eigenproblem size is reduced by half by introducing the following vectors and matrices [4]:

$$\mathcal{X}=\mathbf{M}(\mathbf{\Phi}^+ + \mathbf{\Phi}^-),$$
$$\mathcal{Y}=\mathbf{M}(\mathbf{\Phi}^+ - \mathbf{\Phi}^-), \tag{40}$$

$$\mathbf{E}=\left(\mathbf{I}_{4N}-\frac{\omega}{2}\sum_{l=m}^{L}\mathbf{\Pi}_l\mathbf{B}_l\left[\mathbf{I}_{4N}+(-1)^{l-m}\mathbf{D}\right]\mathbf{\Pi}_l^T\mathbf{W}\right)\mathbf{M}^{-1},$$
$$\mathbf{F}=\left(\mathbf{I}_{4N}-\frac{\omega}{2}\sum_{l=m}^{L}\mathbf{\Pi}_l\mathbf{B}_l\left[\mathbf{I}_{4N}-(-1)^{l-m}\mathbf{D}\right]\mathbf{\Pi}_l^T\mathbf{W}\right)\mathbf{M}^{-1}, \tag{41}$$

where:

$$\mathbf{\Pi}=\left[\mathbf{P}_l^m(\mu_1),\mathbf{P}_l^m(\mu_2),\cdots,\mathbf{P}_l^m(\mu_N)\right]^T \tag{42}$$

Using the vectors and matrices introduced above gives us the following two eigenproblems:

$$(\mathbf{FE})\,\mathcal{X}=\lambda\mathcal{X}, \tag{43}$$
$$(\mathbf{EF})\,\mathcal{Y}=\lambda\mathcal{Y}. \tag{44}$$

Solution of either of these problems gives us the required eigenvectors and eigenvalues. For example, solving equation 43 we can retrieve the solution to equation 19 as:

$$\nu_j^2=1/\lambda_j,$$
$$\mathbf{\Phi}^{\pm}(\nu_j)=\frac{1}{2}\mathbf{M}^{-1}(\mathbf{I}_{4N}\pm\nu_j\mathbf{E})\mathcal{X}(\lambda_j). \tag{45}$$

While reducing the size of the eigen solution implies an extra eigen solution reconstruction step, it is overall less expensive to do so.

Thus the homogeneous solution is carried out in 3 steps:

- First the **E** and **F** matrices are computed. Each matrix is computed one 4×4 subblock at a time, each requiring 3 matrix multiplications. This step involves $N \times N \times L$ independent subblock computations that can be computed in parallel.

- The second step is to solve L eigenproblems, one for each order of expansion. This step also requires that the matrix FE is set up before carrying out the solution. So this step requires L matrix multiplications to compute the **FE** matrices, where dimension of each matrix is $4N \times 4N$. The matrix computation can be done in $L \times 4N \times 4N$ parallel computation $4N$ sum of elements.

- The final step is to compute the eigenvectors $\mathbf{\Phi}^{\pm}$ from the half-problem solution. This step involves $L \times 4N \times 4N$ sums of $4N$ elements, that also can be computed in parallel.

3.B. Particular solution

The particular solution requires solving $8N$ equations for $8N$ unknowns. This solution requires solving a linear system of $8N$ equations. Using a strategy similar to that done is homogeneous solution, the problem size is reduced [4] to solving $4N$ equations for $4N$ unknowns as follows. This is done by introducing the following vector:

$$\mathbf{G} = \mathbf{Z}^{+} + \mathbf{Z}^{-}, \tag{46}$$

and solving a system of $4N$ linear equations:

$$\left(\mathbf{FE} - \frac{1}{\mu_0^2}\right)\mathbf{G} = \left(\mathbf{FX}^{+} + \frac{1}{\mu_0}\mathbf{X}^{-}\right)\mathbf{M}^{-1}. \tag{47}$$

Once solved, the $\mathbf{Z}^{\pm}$ vectors as used in (31) can be retrieved as:

$$\mathbf{Z}^{\pm} = \frac{1}{2}\mathbf{M}^{-1}\left[\mathbf{I}_{4N} \pm \frac{1}{\mu_0}\mathbf{E}\right]\mathbf{G}. \tag{48}$$

Here are the steps for computing the particular solution:

- Creation of the problem matrix (equation 47): It involves, for each order of expansion, two $4N \times 4N$ matrix-vector multiplications, and thus a total of $2L$ such multiplications. This step can be executed as $L \times 4N$ parallel computations, each involving a $4N$ sum and two scalar multiplications.

- Computing the **G** vectors: It requires a total of L $4N \times 4N$ matrix inversions and multiplications.

- Finally, retrieving the $\mathbf{Z}^{\pm}$ vectors: It costs a total of L multiplications of $4N \times 4N$ matrices (to compute **EG**), as well as $2L$ sums of $4N$ vectors to retrieve $\mathbf{Z}^{+}$ and $\mathbf{Z}^{-}$.

3.C. Boundary conditions

The boundary conditions are used to find the $8N$ unknowns A's and B's for equation 24. In the simple case of zero reflection from the bottom, the boundary condition leads to the following two sets of equations:

$$\mathbf{I}_{-}^{hT}(0) + \mathbf{I}_{-}^{pT}(0) = \mathbf{0}_{4N}, \tag{49}$$

$$\mathbf{I}_{+}^{hT}(\tau_0) + \mathbf{I}_{+}^{pT}(\tau_0) = \mathbf{0}_{4N}, \tag{50}$$

where $\mathbf{0}_{4N}$ is a vertical zero vector of size $4N$. Introducing the following vector of unknown constants:

$$\mathbf{C}^{T} = [A_1, A_2, \cdots, A_{4N}, B_1, B_2, \cdots, B_{4N}], \tag{51}$$

we can write any homogeneous solution as:

$$\mathbf{I}_h(\tau, \pm\mu_i) = \mathbf{K}^{T}(\tau, \pm\mu_i)\mathbf{C}, \tag{52}$$

with:

$$\mathbf{K}(\tau, \pm\mu_i) = \begin{bmatrix} \mathbf{\Phi}^{T}(\nu_1, \pm\mu_i)\exp^{\frac{-\tau}{\nu_1}} \\ \vdots \\ \mathbf{\Phi}^{T}(\nu_{4N}, \pm\mu_i)\exp^{\frac{-\tau}{\nu_{4N}}} \\ \mathbf{\Phi}^{T}(\nu_1, \mp\mu_i)\exp^{\frac{-(\tau_0-\tau)}{\nu_1}} \\ \vdots \\ \mathbf{\Phi}^{T}(\nu_{4N}, \mp\mu_i)\exp^{\frac{-(\tau_0-\tau)}{\nu_{4N}}} \end{bmatrix}. \tag{53}$$

Therefore the boundary problem can be expressed using the following matrix notation:

$$\begin{bmatrix} \mathbf{K}_{+}^{T}(0) \\ \mathbf{K}_{-}^{T}(\tau_0) \end{bmatrix}\mathbf{C} = -\begin{bmatrix} \mathbf{I}_{-}^{p}(0) \\ \mathbf{I}_{+}^{p}(\tau_0) \end{bmatrix}, \tag{54}$$

where

$$\mathbf{K}_{+}(\tau) = [\mathbf{K}(\tau, -\mu_1), \cdots, \mathbf{K}(\tau, -\mu_N)], \tag{55}$$

$$\mathbf{K}_{-}(\tau) = [\mathbf{K}(\tau, \mu_1), \cdots, \mathbf{K}(\tau, \mu_N)]. \tag{56}$$

Thus in the simple case of no reflection from the bottom boundary, the solution can be split into the following two steps:

- The creation of the left hand side matrix (equation 54) involves, for each order of expansion, $8N \times 8N$ scalar operations.

- Computing the **C** vectors requires a total of L matrix inversions.

3.C.1. Base Reflection

For solution with nonzero base boundary reflection the righthand side of the equation 50 is nonzero and hence we have to take into that computation into account. We rewrite equation 50 as

$$\mathbf{I}_{+}^{hT}(\tau_0) + \mathbf{I}_{+}^{pT}(\tau_0) = \mathbf{L}_{refl}. \tag{57}$$

$\mathbf{L}_{refl}$ is derived by discretizing the integral part of the righthand side of equation 35 for every quadrature angle μ_i as:

$$\mathbf{L}_{refl}(\mu_i) = \sum_{j=1}^{N}\alpha_j\mathbf{R}(\mu_i, -\mu_j)\mu_j\mathbf{I}(\tau_0, -\mu_j) + \mathbf{L}_{refl}^{s}(\mu_i),$$

where $\mathbf{R}$ is the 4×4 Mueller BRDF matrix, and $\mathbf{L}^s_{refl}$ represents the reflection of the attenuated source term in equation 35, i.e.

$$\mathbf{L}^s_{refl}(\mu_i) = \mathbf{R}(\mu_i, -\mu_0)\frac{\mu_0}{2\pi}\mathbf{I}_0 \exp^{-\tau_0/\mu_0} . \qquad (58)$$

Using equation 52 we rewrite the reflection equation as:

$$\mathbf{L}_{refl}(\mu_i) = \mathbf{L}^h_{refl}(\mu_i)\mathbf{C} + \mathbf{L}^p_{refl}(\mu_i) + \mathbf{L}^s_{refl}(\mu_i), (59)$$

where

$$\mathbf{L}^h_{refl}(\mu_i) = \sum_{j=1}^{N} \alpha_j \mathbf{R}(\mu_i, -\mu_j)\mu_j \mathbf{K}^T(\tau_0, -\mu_j), \qquad (60)$$

$$\mathbf{L}^p_{refl}(\mu_i) = \sum_{j=1}^{N} \alpha_j \mathbf{R}(\mu_i, -\mu_j)\mu_j \mathbf{I}_p(\tau_0, -\mu_j). \qquad (61)$$

Introducing the vectors:

$$\mathbf{L}^H_{refl} = \left[\mathbf{L}^{hT}_{refl}(\mu_1), \cdots, \mathbf{L}^{hT}_{refl}(\mu_N)\right]^T, \qquad (62)$$

$$\mathbf{L}^P_{refl} = \left[\mathbf{L}^{pT}_{refl}(\mu_1), \cdots, \mathbf{L}^{pT}_{refl}(\mu_N)\right]^T, \qquad (63)$$

$$\mathbf{L}^S_{refl} = \left[\mathbf{L}^{sT}_{refl}(\mu_1), \cdots, \mathbf{L}^{sT}_{refl}(\mu_N)\right]^T, \qquad (64)$$

we now write the boundary problem in the case of non-zero base reflection as:

$$\left(\begin{bmatrix} \mathbf{K}^T_+(0) \\ \mathbf{K}^T_-(\tau_0) \end{bmatrix} - \begin{bmatrix} \mathbf{0}_{4N} \\ \mathbf{L}^H_{refl} \end{bmatrix}\right)\mathbf{C} = \begin{bmatrix} \mathbf{0}_{4N} \\ \mathbf{L}^P_{refl} + \mathbf{L}^S_{refl} \end{bmatrix} - \begin{bmatrix} \mathbf{I}^p_-(0) \\ \mathbf{I}^p_+(\tau_0) \end{bmatrix}. \quad (65)$$

Thus base reflection adds an extra computation step to the boundary problem where the downward radiance field needs to be integrated at the bottom of the layer, and then used to update the boundary condition. The computation of equations 60 and 61 for every μ require summation involving $4N$ terms. However, equation 60 is evaluated $4N$ times for each order, so a total of $L \times 4N$ summations, whereas the equation 61 is evaluated only once. So in total this step can be computed as $L \times 4N + 1$ parallel summation evaluations. The base reflection also evaluates equation 58, $4N$ times. However, this evaluation is a simple and hence is not included in this discussion.

In the context of base reflection, the particular case of ideal depolarizing Lambertian base needs a special mention, because it reduces to computation cost to some extent. For such a reflection only the top-left element of Mueller BRDF matrix $\mathbf{R}$ is non-zero, i.e.

$$\mathbf{R}_{00}(\mu_i, -\mu_j) = 2\rho, \qquad (66)$$

where ρ is the Lambertian albedo.

Furthermore, because of the directional independence of Lambertian reflectors $\mathbf{L}^h_{refl}$, $\mathbf{L}^p_{refl}$ and $\mathbf{L}^s_{refl}$ in equations 60, 61, and 58 are independent of μ_i, hence are computed only once, and are duplicated to set up the corresponding vectors. Thus the computation cost is reduced by a factor of N.

3.D. Reconstruction

The reconstruction of the radiance field at any τ value requires the evaluation of equations 36 and 37 for any arbitrary $\pm\mu$ value where $\mu \in \{0, 1\}$, and for any azimuth angles ϕ. However, for BRDF computation the radiance field computation is limited only to $\tau = 0$ and to $+\mu$ values. However, this computation must be carried out for all incident μ_0 values. Independent of whether it is done for radiance field or for BRDF, the reconstruction step consists of the following four steps.

- Computing $\mathbf{I}(\tau_0, \mu)$: It is the radiance at the base of the layer (the first item on the right of equation 36). The cost of this step depends on the cost of the bottom boundary condition. In the simplest case where no light is reflected at the bottom of the boundary, that step has no cost as $\mathbf{I}(\tau_0, \mu)$. Otherwise, for every μ the the cost is a sum of $4N \times 4N$ terms involving homogeneous solution and a sum of $4N$ terms involving particular solution. For Lambertian base, this cost remains the same. However, the term is independent of μ, so needs to be computed only once.

- Computing $\mathbf{I}$ for equation 38: We analyze the cost in the following three substeps. Each of these substeps require N^2 matrices $\mathbf{A}(\mu, \mu')$ each of size 4×4, which must be obtained through a sum of up to L elements (see equation 9) each involving two 4×4 matrix multiplications.
    - Computing $\mathbf{I}^h$ component of $\mathbf{I}$: It requires $L \times 8N$ sums of $2N$ elements, each element of which is obtained through a 4×4 matrix ($\mathbf{A}(\mu, \mu')$) and 4 elements vectors ($\mathbf{\Phi}(\nu, \mu)$) multiplication.
    - Computing $\mathbf{I}^p$ component of $\mathbf{I}$: This step requires L sums of $2N$ elements, each element of which is obtained through a 4×4 matrix ($\mathbf{A}(\mu, \mu')$) and 4 elements vectors ($\mathbf{Z}(\mu)$) multiplication.
    - Computing Q term for equation 38 : It is the last term of equation 38. That terms needs to be evaluated at each order of expansion, and for every incident direction in the desired BRDF. For each of these direction, this step can be executed as a single 4×4 matrix ($\mathbf{A}(\mu, \mu_0)$) and 4 elements vectors ($\mathbf{I}_0$) multiplication. Note that this substep requires $\mathbf{A}(\mu, \mu_0)$ which is different from $\mathbf{A}(\mu, \mu')$ and hence must be computed independently from the other two substeps.

This reconstruction is done for every order of expansion, resulting in all the $\mathbf{I}^m(\tau, \mu)$. A final reconstruction step is needed to get the radiance at any azimuthal angle ϕ using the sum from equation 12.

4. Implementation, Results and Discussion

We implemented our VRTE solver in C++. For all the parallelizable steps of the solver (that were identified in section 3) we created sequential and parallel versions. The parallel version is written in OpenCL (a parallel computing language for multi core and many core devices) [11]. We chose between the sequential and parallel version through compiler directives. Instead of writing our own code for Eigen solution (used for Homogeneous solution) and matrix inversion (used for Particular solution and Boundary condition based solution), we used function calls from one of the two well established Linear Algebra C++ libraries: EIGEN [12], a library that relies on sequential computation, and MAGMA [13], a GPU accelerated library. Even though EIGEN is designed for sequential computation, it has an optional vectorization feature that could be enabled to take advantage of the vector instruction set of the CPU. Taking combinations of sequential and parallel implementation, EIGEN libray (vectorized and nonvectorized) and MAGMA library, we created six different configurations for our solver. These configurations are identified in Table 1. In the rest of this section we compare the computation times from these configurations, show the lobes of some of the computed BRDFs, and show some rendered images of a simple scene with objects having the computed BRDFs as their surface reflection properties. The phase functions used in this section (except for the benchmark test) are all computed using a publicly available Mie theory based code by Mischenko [18]. All the results shown in this section were obtained using a laptop with a Intel Core i7 as CPU and a Nvidia GTX 580 as GPU.

	EIGEN without vectorization	EIGEN with vectorization	MAGMA
Sequential solution	Conf. 1	Conf. 2	Conf. 3
Parallel solution	Conf. 4	Con.n 5	Conf. 6

Table 1. Various configurations of our VRTE solver.

4.A. Validation

To validate our implementation, we ran the Wauben and Hovenier benchmark test [14] on all the configurations of our solver. The benchmark uses 30 quadrature angles and 12 orders of expansion for the phase function.

The results from all the solver configurations are in perfect agreement with the benchmark results. Table 2 shows the computation time for running the benchmark on each of the solver configurations. The table shows a reasonably good improvement ($\approx$ 3 times) between the parallel solver and the sequential solver. Among the configurations, we noticed only minor improvements for using vectorized EIGEN library as compared to using its nonvectorized counterpart. The results from using MAGMA library showed a little discouraging slow down as compared to the computation time from using EIGEN library. We investigate this problem later in the following paragraphs.

	EIGEN without vectorization	EIGEN with vectorization	MAGMA
Sequential solution	2.79	2.74	2.94
Parallel solution	0.99	0.97	1.18

Table 2. Time (in seconds) required to compute the benchmark problem using each of the six configurations.

4.B. Radiance field due to a directional incidence

We ran our solver to compute radiance field due to a directional ($\mu_0 = 0.6$) monochromatic ($\lambda = 400$nm) light incident on homogeneous particle layers of thickness 100 ($\tau_0 = 100$) composed of spherical particles (radius $0.6\mu m$) from several materials. We used 40 quadrature nodes ($N = 40$) for the computation. The radiance field was computed for 11 zenith and 19 azimuth angles.

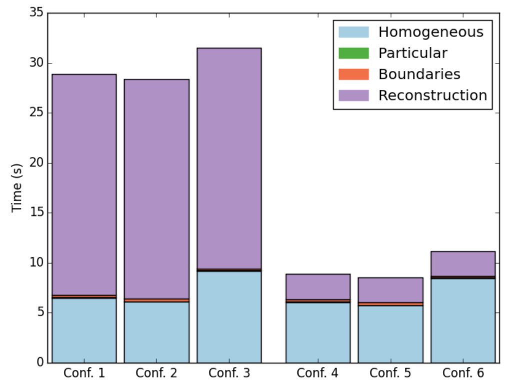


Fig. 1. Detailed cost of computing a radiance field with the different configurations.

Table 4 shows the total computation time and time spent for each major step of the computation on each of the six configurations for one of the materials (a layered suspension of gold particles). All other materials showed similar trends. Figure 1 shows the same information in a stacked bar chart. We summarize below our observations from the table:

- Total computation cost: All the solver configurations that included parallelization ran faster than their sequential counter part. We find a speed-up of more than 3 between the fastest ones among the sequential and parallel configurations.

- Cost of individual computation steps: The computation times for the individual steps do not all show an obvious trend. Independent of the solver configuration, we find that a lion's share of computation time is taken up by the reconstruction step. And this step is also the step that takes the maximum advantage (hence most speedup) from parallelization. If we ignore the configuration using MAGMA (discussed later), the remaining three steps of the computation (homogeneous, particular and boundary solution) do not show any significant speed difference. We justify this finding by pointing to the fact that independent of the configurations, the most expensive matrix computations (eigen solution and matrix inversion) for these steps are done sequentially for each order of expansion using calls to EIGEN library. The particular speed advantage associated with the computation of the matrices in parallel was mostly counter balanced by the time taken to transfer the data to and from the GPU device and in preparing the parameters for EIGEN function calls from the retrieved data. Thus the reconstruction step fully influenced the total speed up in computation time.

- Vectorized *vs.* nonvectorized EIGEN library: The vectorized EIGEN library consistently showed a slight improvement in computation time over the nonvectorized counterpart. Thus it was unnecessary to continue experimenting with the configurations that used nonvectorized EIGEN library. So we drop them from all the subsequent tables and plots.

- EIGEN library *vs.* MAGMA library: We observed in our benchmark validation step that the timing results from configurations using MAGMA library has a minor slow down. We notice the same trend here as well. We had expected that the GPU based MAGMA library would improve our computation time for eigen solution (homogeneous solution step) and for matrix inversion (particular solution and boundary handling step). We later found from studies conducted by independent researches [15] that though the GPU based matrix libraries (such as MAGMA) are supposed to accelerate the computation, for complex matrix related problems (such as eigen solution) initial setup times are high and noticeable speedup is only achieved for matrix sizes many time larger than those used in our VRTE solver.

  So we discontinued the configurations using MAGMA library.

Based on the above observations we continued our experiment using only two remaining configurations: Sequential implementation using vectorized EIGEN library (Configuration 2), and Parallel implementation using vectorized EIGEN library (Configuration 5).

We then went on to study the computation time trend of our configurations as a function of the various parameters of the solver: number of quadrature nodes, orders of expansion, number of outgoing directions for reconstruction. We carried out these experiments to further validate and to verify the scalability of our implementation. For each of the cases we measured total time and time spent by the individual steps.

Configuration:	1	2	3	4	5	6
Homogeneous:	6.50	6.11	9.18	6.07	5.75	8.43
Particular:	0.04	0.04	0.07	0.04	0.04	0.07
Boundary problem:	0.24	0.24	0.25	0.24	0.24	0.25
Reconstruction:	22.1	22.0	22.1	2.52	2.51	2.53
Total:	28.9	28.4	31.5	8.88	8.56	11.18

Table 3. Time (in seconds) to solve each step of the computation of the radiance field with the different configurations.

4.B.1. Influence of the quadrature size

The quadrature size improves the accuracy of the DOM computation. However, they also increase the computation time. In fact, according to the analysis carried out in the previous section, the computation speed should decrease quadratically or cubically (depending on the computation) with the increase in the quadrature size N. Figure 2 plots total computation time as a function of quadrature size and figure 3 plots the breakup of the computation time for execution of each step. The curves are consistently in agreement with our expectation. As discussed earlier, except for the reconstruction step, the computation times for the remaining steps were almost the same for the sequential and the parallel configuration. The reconstruction step of the parallel implementation (configuration 5) did not seem to be affected by the increase in in quadrature size. This could be because the problem size used in the experiments were still not enough to make a full utilization of GPU computation units. Overall, because of the accelerated reconstruction step, the speed of the sequential implementation (configuration 2) lagged behind in parallel implementation (configuration 5) by more than a constant factor.

4.B.2. Influence of the number of direction of the radiance field

The only step affected by the number of outgoing directions (μ) of the radiance field is the reconstruction step, and the cost analysis shows a linear relationship between this number and the reconstruction cost. The plots in figures 4 and 5 agree with our analysis. The computation time of all the steps except the reconstruction step were independent of the number of outgoing directions. As observed earlier the reconstruction step benefited the most from the reconstruction, and hence did the total computation time.

4.B.3. Influence of the order of expansion

The order L of the Fourier expansion of the phase function linearly affects the number of times the computation steps are executed. So the computation time is expected to vary linearly with L. Figures 6 and 7 plot the total

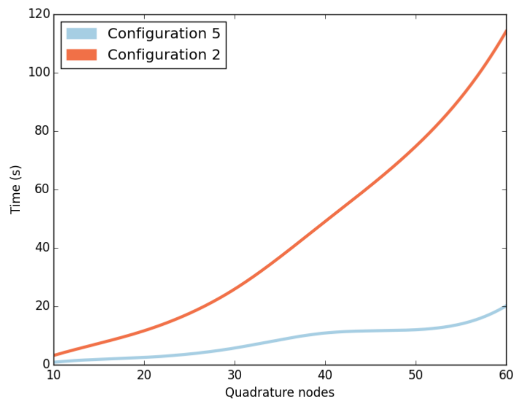


Fig. 2. Cost of the solution as a function of the quadrature size.

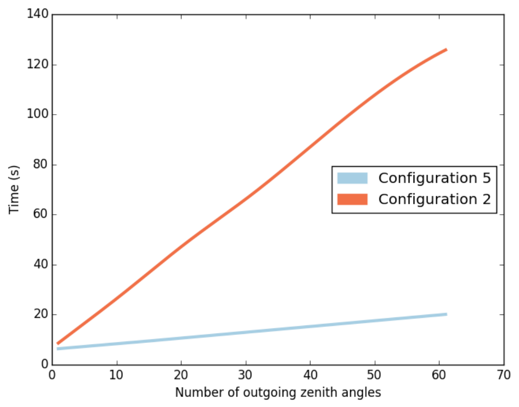


Fig. 4. Solution cost as a function of the number of outgoing directions.

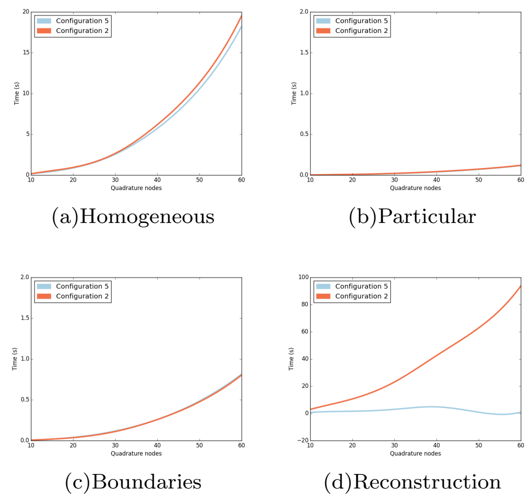


Fig. 3. Cost of each solution step for different quadrature sizes.

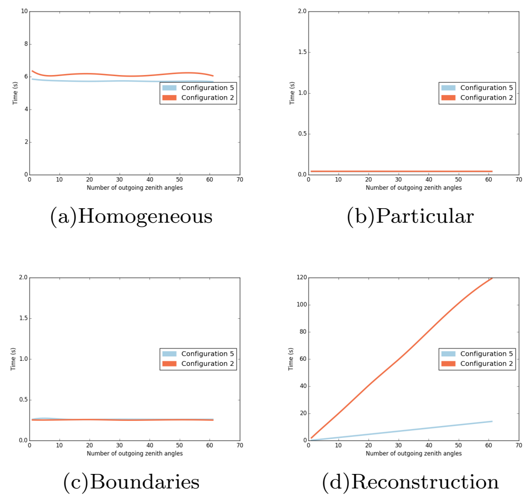


Fig. 5. Cost of each solution step for different numbers of outgoing directions.

computation time and computation times of the individual steps respectively, as a function of the order of expansion[19]. The plots show linear relation between the order of expansion and the total computation time. However, as in earlier observations, the parallel configuration did not perform any better for the computation of the homogeneous, particular and boundary steps as compared to the sequential configuration.

4.C. BRDF computation

Finally we show the computation times for BRDF computation. For the BRDF computation VRTE is solved for a number of incident directions. Note that polarized BRDF is a Mueller matrix, so we need a minimum of four incident Stokes vectors for every direction. All the parameters affecting the radiance field computation equally affect the BRDF computation. We repeated the experiment with the same set up as in section 4.B and measured the computation time as a function of the number of incident directions. Since all the steps except the step involving homogeneous solution had to be repeated for each incident directions and each incident Stokes vector, we expected the cost to increase linearly for those steps with the increase in the number of incident directions. Figures 8 and 9 plot the computation time as a function of the number of incident directions. The curves very much agree with the expectation: the computation time of the homogeneous solution step remained unchanged and all the other times showed linear trend.

4.D. Renderings

We used our solver to compute the BRDF for material layers composed of several particle types. Figure 10 shows for each material two computed BRDF lobes: one for an oblique incidence ($\mu_0 = 0.6$) and the other for normal incidence, and a rendered image of a scene com-

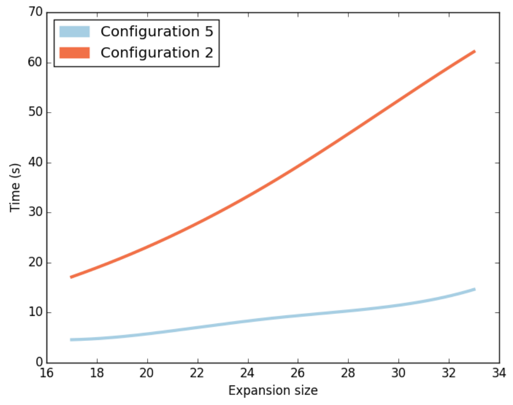


Fig. 6. Solution cost as a function of the expansion size.

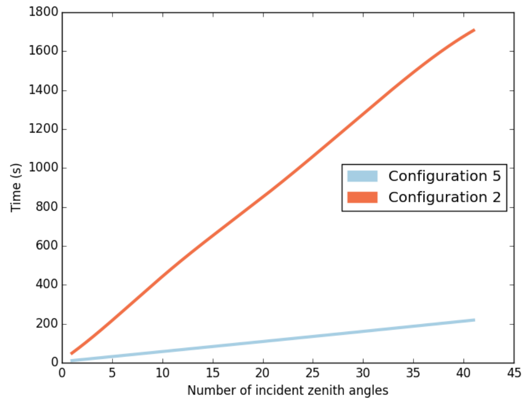


Fig. 8. Solution cost as a function of the number of incident directions.

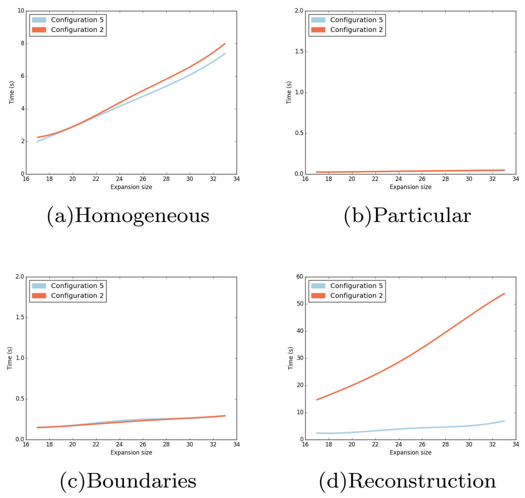


Fig. 7. Cost of each solution step for different expansion sizes.

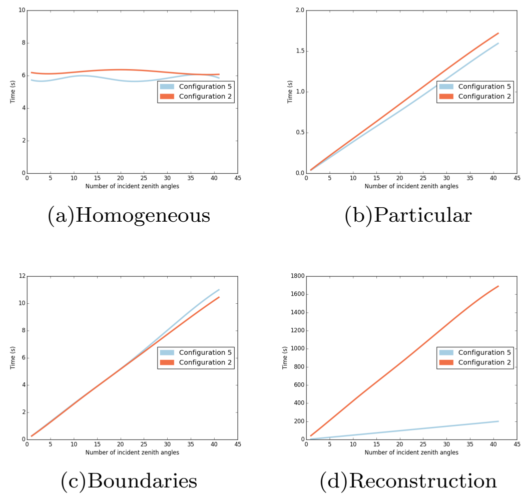


Fig. 9. Cost of each solution step for different numbers of incident directions.

posed of spherical objects with the computed BRDF as their surface properties. The scene was illuminated with synthetic skylight. Each row of the figure corresponds to a different material.

The current version of our parallel solver (configuration 5) computes a polarized subsurface BRDF in $\tilde{1}5$ minutes, where as the sequential solver (configuration 2) takes close to two hours for this computation.

We have made the solver used in this section available [16] for public use. The solver takes the material specification as input, and based on the user's choice computes the polarized radiance field at any layer thickness (τ value) or computes the polarized BRDF. The output is made available to the user in a tabular form for download. The page also provides a renderer to visualize the computed radiance field or the BRDF lobe. In the latter case, the renderer allows interactive viewing of the computed lobe as a function of incident direction. The rendered sphere images used in this section were obtained using a polarized path tracer. We have also made this path tracer available for public use [17].

5. Conclusion

In this paper we proposed parallelization of DOM based VRTE solvers for computing polarized light transport inside homogenous layered materials for computing polarized radiance field and for computing polarized subsurface BRDF. We analyzed the cost of each step of the solution, identified potential parallelization steps, and implemented those steps in OpenCL to run them in parallel in GPU. The major bottleneck was found to be in solving eigen problem and in computing matrix inversion. Though we attempted to use GPU based library for these computations, for our problem size we found only minor speedup for matrix inversion computation and minor setback for eigen solution computation. The parallelization gave us significant speedup in the final reconstruction step that dominated the cost of radiance field computation and more so in BRDF computation using the solver. So the overall computation speed-up for our parallel solver was found to be significant as compared to its sequential counterpart. Our parallel software configuration (that uses a sequential linear algebra library for eigen problem and linear system solution) is about seven times faster than the sequential configuration for BRDF computation with a reasonable number of incident and outgoing directions.

Though our parallel solver allows us to compute BRDF faster, we believe that we will be able to further improve its speed. We are still using an external Linear Algebra library and computing the expensive matrix problems (eigen solution and matrix inversion) sequentially, and furthermore solving them sequentially for each order of expansion. In future we plan to write our parallelized eigen solver and linear system solver that will allow us to compute eigen solution and matrix inversion for all the expansion orders in one function call each. This will result in an increase in problem size for better utilization of GPU resources, and will help us bring down the computation time further.

Acknowledgments

This work was supported by NSF grant IIS-1064427.

References

[1] Chandrasekhar, *Radiative transfer*, Dover publications, 1960.

[2] Stamnes, Knut, et al. *DISORT, a general-purpose Fortran program for discrete-ordinate-method radiative transfer in scattering and emitting layered media: documentation of methodology.* Goddard Space Flight Center, NASA (2000).

[3] Hecht and Zajac. *Optics*, volume 4. Addison Wesley San Francisco, CA, 2002.

[4] Siewert. *A discrete-ordinates solution for radiative-transfer models that include polarization effects.* Journal of Quantitative Spectroscopy and Radiative Transfer, 64(3):227-254, 2000.

[5] Eriksson, Patrick, et al. *ARTS, the atmospheric radiative transfer simulator, version 2.* Journal of Quantitative Spectroscopy and Radiative Transfer 112.10 (2011): 1551-1558.

[6] Spurr, Robert JD. *VLIDORT: A linearized pseudo-spherical vector discrete ordinate radiative transfer code for forward model and retrieval studies in multilayer multiple scattering media.* Journal of Quantitative Spectroscopy and Radiative Transfer 102.2 (2006): 316-342.

[7] Barichello, L. B., R. D. M. Garcia, and C. E. Siewert. *The Fourier decomposition for a radiative-transfer problem with an asymmetrically reflecting ground.* Journal of Quantitative Spectroscopy and Radiative Transfer 56.3 (1996): 363-371.

[8] Siewert, C. E. *On the phase matrix basic to the scattering of polarized light.* Astronomy and Astrophysics 109 (1982): 195.

[9] Stamnes, Knut, and Roy A. Swanson. *A new look at the discrete ordinate method for radiative transfer calculations in anisotropically scattering atmospheres.* Journal of the Atmospheric Sciences 38.2 (1981): 387-399.

[10] Thomas, Gary E., and Knut Stamnes. *Radiative transfer in the atmosphere and ocean.* Cambridge University Press, 2002.

[11] Stone, John E., David Gohara, and Guochun Shi. *OpenCL: A parallel programming standard for heterogeneous computing systems.* Computing in science & engineering 12.1-3 (2010): 66-73.

[12] http://eigen.tuxfamily.org/, last accessed on February 13th 2015

[13] http://icl.cs.utk.edu/magma/, last accessed on February 13th 2015

[14] Wauben, W. M. F., and J. W. Hovenier. *Polarized radiation of an atmosphere containing randomly-oriented spheroids.* Journal of Quantitative Spectroscopy and Radiative Transfer 47.6 (1992): 491-504.

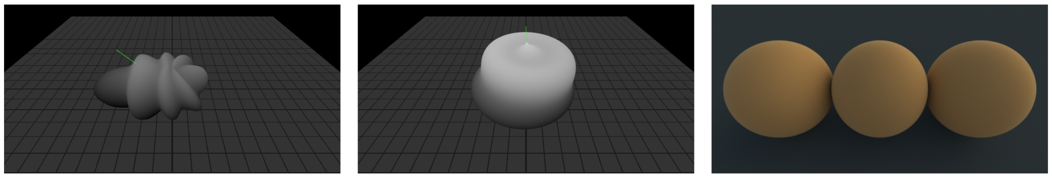
[15] https://github.com/bravegag/eigen-magma-benchmark, last accessed on February 13th 2015

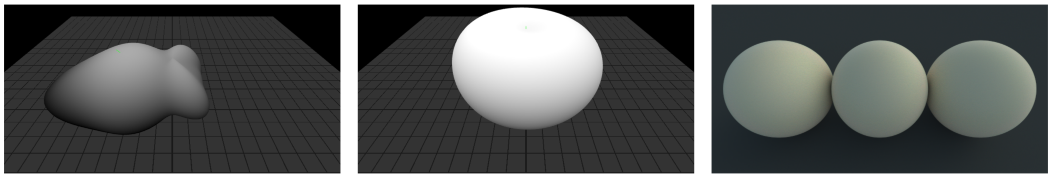
[16] http://graphics.cs.ucf.edu/tools/PIVERT/, last accessed on February 13th 2015

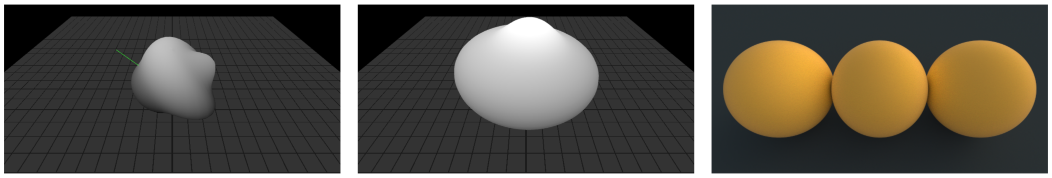
[17] http://graphics.cs.ucf.edu/tools/PIRATE/, last accessed on February 13th 2015

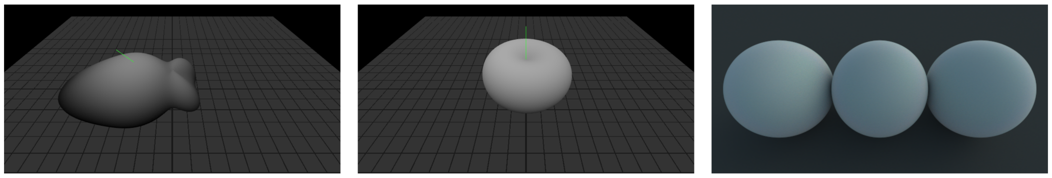
[18] http://www.giss.nasa.gov/staff/mmishchenko/brf/, last accessed on February 13th 2015

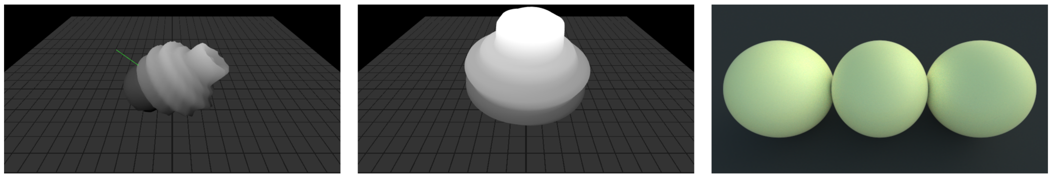
[19] The order of the Fourier expansion of the phase function depends on the material. For particles of a given material and a given wavelength of light, it obviously depends on the particle size. So to experiment with different orders of expansion, we varied the size of the particles (between $0.2\mu m$ and $0.7\mu m$) in the layer and then sorted the phase functions by their order of expansion.

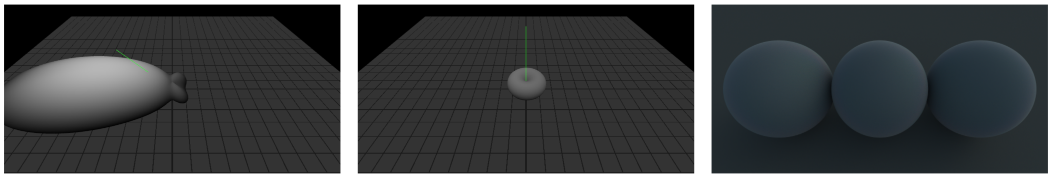

(a)Au - Particle size: $0.6\mu m$

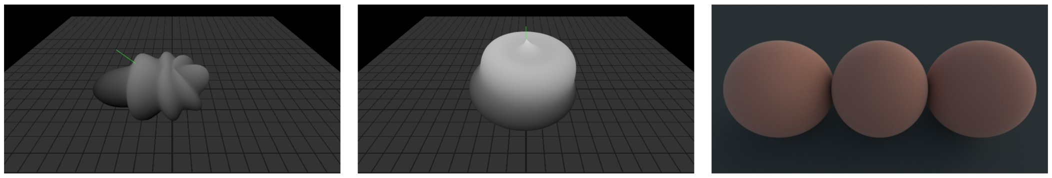

(b)Ag - Particle size: $0.3\mu m$


(c)$AlGaAs$ - Particle size: $0.3\mu m$ (Lobe scale factor = 0.5)


(d)$AlCu$ - Particle size: $0.3\mu m$


(e)$ZnSCuB$ - Particle size: $0.8\mu m$ (Lobe scale factor = 0.5)


(f)PbS - Particle size: $0.3\mu m$ (Lobe scale factor = 2.0)


(g)Cu - Particle size: $0.6\mu m$

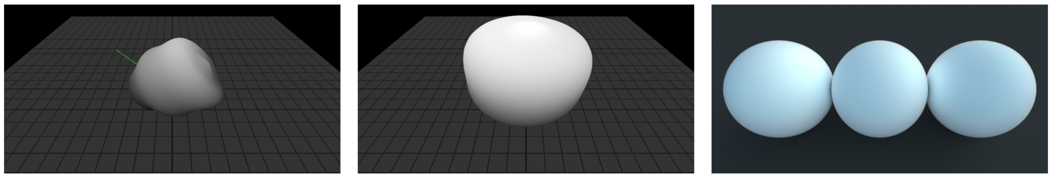

(h)TiO_2 - Particle size: $0.6\mu m$ (Lobe scale factor = 0.5)

Fig. 10. BRDF lobes and renderings for various materials and particle size. Lobes are shown for a wavelength $\lambda = 640nm$. Left column corresponds to an incident direction $\mu_i = 0.6$, middle column to a normal incident direction ($\mu_i = 1.0$). Right column shows a rendering using 9 wavelengths. Some of the lobes have been scaled to a reasonable size. For those lobes, the scale factor is given below the lobes.