«Оптика атмосферы и океана», **28**, № 4 (2015)

УДК 535.31; 535.36

Алгоритм трассировки пучков для задачи рассеяния света на атмосферных ледяных кристаллах. Часть 1. Теоретические основы алгоритма

А.В. Коношонкин[1,2], Н.В. Кустова[2], А.Г. Боровой[1,2]*

[1]Национальный исследовательский Томский государственный университет
634050, г. Томск, пр. Ленина, 36
[2]Институт оптики атмосферы им. В.Е. Зуева СО РАН
634021, г. Томск, пл. Академика Зуева, 1

Поступила в редакцию 17.02.2015 г.

Представлен алгоритм трассировки пучков, позволяющий получить решение задачи рассеяния света на атмосферных ледяных кристаллических частицах в приближении геометрической оптики. Подробно рассматривается построение матриц Джонса и Мюллера, положенное в основу алгоритма. Уделяется особое внимание интерфейсу программной реализации алгоритма, что облегчает его внедрение в сторонний проект. Разработанный алгоритм находится в свободном доступе с открытым исходным кодом.

Ключевые слова: геометрическая оптика, алгоритм трассировки пучков, рассеяние света, ледяные кристаллы; geometrical optics, beam splitting algorithm, light scattering, ice crystals.

Введение

Задача рассеяния света на крупных несферических частицах вызывает пристальный интерес всего мирового научного сообщества (см. [1, 2]). Строгое решение уравнений Максвелла для крупных, по сравнению с длиной падающей волны, частиц не удается получить ввиду недостатка вычислительных мощностей современных компьютеров. Поэтому данная задача обычно решается в приближении геометрической оптики [3, 4]. В последнее время с появлением метода физической оптики удалось показать переход от точного решения задачи рассеяния света на несферических частицах к приближенному решению геометрической оптики [5, 6]. Использование метода физической оптики позволило определить границу применимости приближения геометрической оптики [7], очертив широкий круг задач, который может быть успешно решен в рамках геометрической оптики.

Стоит отметить, что бурное развитие приближения геометрической оптики для решения задач рассеяния на несферических частицах в 1970–1990 гг. произошло задолго до того, как была обоснована граница его применимости, чему способствовали наглядность и простота компьютерной реализации метода [8–14]. Традиционно приближение геометрической оптики реализуется на основе алгоритма трассировки лучей, поскольку он является самым очевидным и наглядным. Такой подход прост, но содержит несколько недостатков: необходимо отдельно исследовать достоверность полученного решения ввиду стохастического начального расположения лучей; ряд проблем связан с генератором случайных начальных координат падающих лучей; увеличение точности влечет за собой существенное увеличение количества начальных лучей и, следовательно, вычислительной сложности. Тем не менее такая реализация приближения геометрической оптики получила широкое распространение, во многом обусловленное наличием в свободном доступе открытого алгоритма A. Macke [15], разработанного более 20 лет назад и активно используемого в настоящее время [16, 17].

Вполне очевидно, что для случая кристаллических частиц (грани которых плоские) алгоритм трассировки лучей не является оптимальным, поскольку при отражении/преломлении светового потока от/на плоской грани образующие световой пучок лучи имеют одинаковые характеристики и их отдельный расчет является избыточным. В связи с этим многими авторами предпринимались попытки сгруппировать лучи в так называемые лучевые трубки (см., например, [18, 19]). Однако еще более эффективно трассировать не лучевые трубки, а непосредственно плоскопараллельные пучки света. Впервые такой подход был предложен А.А. Поповым в 1984 г. [20] и реализован M. Del Guasta [21] в 1995 г., затем независимо реализован Д.Н. Ромашовым в 2001 г. [22] и А.Г. Боровым в 2003 г. [23], также повторен L. Bi в 2014 г. [5] в рамках метода физической оптики. К сожалению, из всех реализаций только алгоритм авторов настоящей статьи является открытым и находится в свободном доступе [24].

* Александр Владимирович Коношонкин (sasha_tvo@iao.ru); Наталья Валентиновна Кустова; Анатолий Георгиевич Боровой (borovoi@iao.ru).

Описание алгоритма трассировки пучков [23] приводится, чтобы сделать его доступным всему мировому оптическому научному сообществу. В настоящей статье рассматриваются необходимая теоретическая база, положенная в основу алгоритма, а также интерфейс программы. Сравнению результатов, полученных алгоритмом трассировки пучков и алгоритмами других авторов, посвящена вторая часть статьи (с. 331–337).

Программный код алгоритма написан на языке C++ в виде удобной библиотеки, облегчающей внедрение алгоритма трассировки пучков в сторонний код.

Теоретическая основа алгоритма

Матрицы Джонса и Мюллера

В алгоритме трассировки пучков решение задачи рассеяния света на отдельной частице представляется посредством матриц Джонса **J** и/или Мюллера **M**. Матрица Джонса выражает рассеянное поле $\mathbf{E}_s$ в некоторой точке через падающее поле $\mathbf{E}_i$:

$$\mathbf{E}_s = \mathbf{J}\mathbf{E}_i. \tag{1}$$

В случае, когда оптические характеристики света определяются параметрами Стокса, используется матрица Мюллера, которая связывает параметры Стокса рассеянного поля $\mathbf{I}_s$ с параметрами Стокса падающего поля $\mathbf{I}_i$:

$$\mathbf{I}_s = \mathbf{M}\mathbf{I}_i. \tag{2}$$

Системы координат

Заданию системы координат необходимо уделять особое внимание. В отличие от работы [3], где используется плоскость рассеяния для определения системы координат, мы в алгоритме трассировки пучков систему координат определяли иначе (рис. 1).

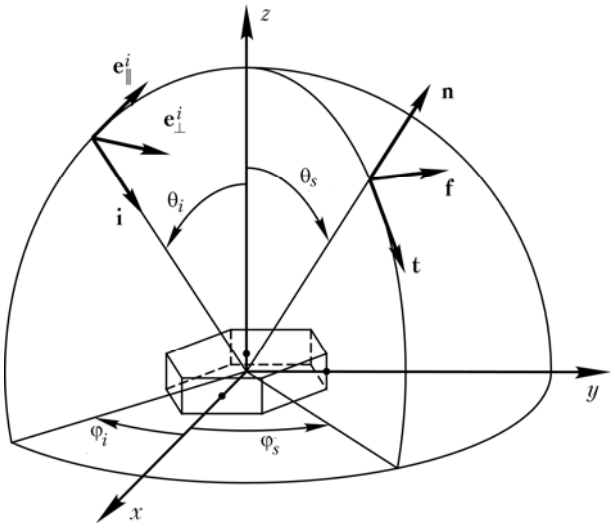


Рис. 1. Системы координат, используемые в алгоритме трассировки пучков

В алгоритме используются 4 системы координат: Глобальная декартова система координат (x, y, z), Сфера направлений рассеяния (θ, φ), Система координат падающего поля ($\mathbf{e}_\parallel^i, \mathbf{e}_\perp^i, \mathbf{i}$), Система координат рассеянного поля (**t**, **f**, **n**).

Центр Сферы направлений рассеяния совпадает с центром Глобальной декартовой системы координат. Зенитный угол θ отсчитывается от оси z, азимутальный угол φ – от оси x. Базисный вектор **i** Системы координат падающего поля, определяющий направление распространения плоскопараллельной падающей волны в Глобальной декартовой системе координат, имеет вид

$$\mathbf{i} = \{-\sin(\theta_i)\cos(\varphi_i), -\sin(\theta_i)\sin(\varphi_i), -\cos(\theta_i)\}, \tag{3}$$

где θ_i, φ_i – направление падения. Второй базисный вектор $\mathbf{e}_\perp$ определяется следующим выражением:

$$\mathbf{e}_\perp^i = \{-\sin(\varphi_i), \cos(\varphi_i), 0\}, \tag{4}$$

во всех точках, кроме полюсов Сферы направлений рассеяния, где базисный вектор **i** и ось z сонаправлены и противонаправлены, в этих случаях $\mathbf{e}_\perp^i = \{0, -1, 0\}$ и $\mathbf{e}_\perp^i = \{0,1,0\}$ соответственно. Оставшийся базисный вектор $\mathbf{e}_\parallel^i$ находится по формуле $\mathbf{e}_\parallel^i = \mathbf{e}_\perp^i \times \mathbf{i}$.

Стоит отметить, что H.C. van de Hulst [3], C.F. Bohren и D.R. Huffman [25] выбирают базисный вектор $\mathbf{e}_\perp^i$ направленным в противоположную сторону, а оставшийся базисный вектор $\mathbf{e}_\parallel^i$ определяют по формуле $\mathbf{e}_\parallel^i = \mathbf{i} \times \mathbf{e}_\perp^i$, что ведет к различию в знаках недиагональных элементов матрицы Джонса. Этот факт необходимо учитывать при сопоставлении результатов.

Аналогичным образом через направление рассеяния θ_s, φ_s выражаются базисные вектора Системы координат рассеянного поля:

$$\mathbf{n} = \{\sin(\theta_s)\cos(\varphi_s), \sin(\theta_s)\sin(\varphi_s), \cos(\theta_s)\}, \tag{5}$$

$$\mathbf{f} = \begin{cases} \{0, -1, 0\}, & \mathbf{n} = \{0, 0, 1\}, \\ \{-\sin(\varphi_s), \cos(\varphi_s), 0\}, \forall \mathbf{n}, \\ \{0, 1, 0\}, & \mathbf{n} = \{0, 0, -1\}, \end{cases} \tag{6}$$

$$\mathbf{t} = \mathbf{f} \times \mathbf{n}. \tag{7}$$

В Системах координат падающего и рассеянного полей падающее поле $\mathbf{E}_i$ и рассеянное поле $\mathbf{E}_s$ имеют вид

$$\begin{aligned} \mathbf{E}_i(\mathbf{r},t) &= \mathbf{E}^i \exp(ik\mathbf{ir})\exp(-i\omega t), \\ \mathbf{E}_s(\mathbf{r},t) &= \mathbf{E}^s \exp(i\psi_s)\exp(ik\mathbf{nr})\exp(-i\omega t), \end{aligned} \tag{8}$$

где $\mathbf{E}^i$ и $\mathbf{E}^s$ – комплексные амплитуды:

$$\mathbf{E}^i = \begin{pmatrix} E_\parallel^i \\ E_\perp^i \end{pmatrix}, \ \mathbf{E}^s = \begin{pmatrix} E_\parallel^s \\ E_\perp^s \end{pmatrix}, \tag{9}$$

$E_\parallel^i$, $E_\perp^i$, $E_\parallel^s$, $E_\perp^s$ – комплексные величины; ψ_s – фазовый сдвиг; t – время; ω – частота; $\mathbf{r}$ – радиус-вектор точки наблюдения.

Таким образом, решение, записанное в виде комплексной матрицы Джонса размерностью 2×2 (1),

6*.

связывает рассеянное и падающее поля, определенные в (9). Для представления решения в виде матрицы Мюллера необходимо определение параметров Стокса:

$$\mathbf{I} = \begin{pmatrix} I \\ Q \\ U \\ V \end{pmatrix}, \tag{10}$$

где

$$\begin{aligned} I &= \left|E_{\parallel}\right|^2 + \left|E_{\perp}\right|^2; \\ Q &= \left|E_{\parallel}\right|^2 - \left|E_{\perp}\right|^2; \\ U &= -(E_{\parallel}E_{\perp}^{*} + E_{\perp}E_{\parallel}^{*}); \\ V &= -i(E_{\parallel}E_{\perp}^{*} - E_{\perp}E_{\parallel}^{*}). \end{aligned} \tag{11}$$

Такое определение совпадает с [1] и формально отличается от [3, 25], однако если учесть упомянутое отличие в задании базиса $\mathbf{e}_{\perp}^{i}$, то оно (11) совпадет с определением авторов [1, 3, 25].

Матрица Мюллера может быть легко получена из матрицы Джонса:

$$\mathbf{M} = \mathbf{\Gamma}(\mathbf{J} \otimes \mathbf{J}^{*})\mathbf{\Gamma}^{-1}. \tag{12}$$

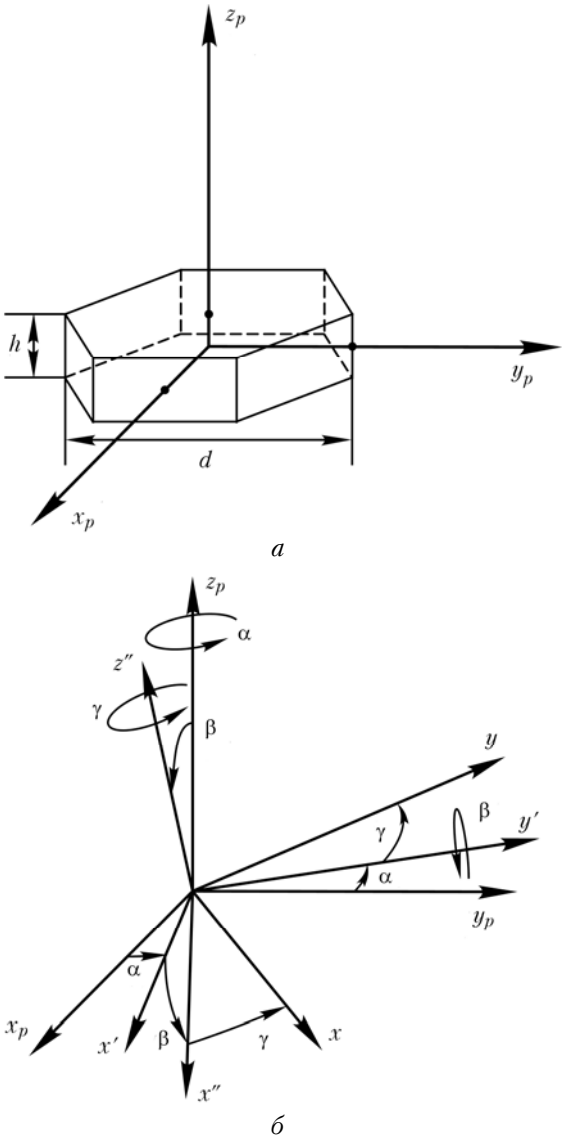
Здесь

$$\mathbf{\Gamma} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 \\ 0 & -1 & -1 & 0 \\ 0 & -i & i & 0 \end{pmatrix}. \tag{13}$$

Задание частицы

Текущая реализация алгоритма трассировки пучков рассматривает только выпуклые частицы. Для удобства в программе уже определены некоторые базовые формы частиц, такие как шестигранная призма, «пуля», дрокстал и т.п. Подробную информацию о типах частиц можно найти в Руководстве пользователя [24]. Частица определяется координатами своих вершин в Глобальной декартовой системе координат. Для этого задаются необходимые размеры частицы в микронах, например для шестигранной призмы задаются высота h и диаметр d окружности, описанной возле основания. Начальная ориентация частицы описана в Руководстве пользователя (рис. 2).

При необходимости частица поворачивается на три угла Эйлера α, β, γ. Ввиду большого количества различных определений углов Эйлера, обозначим сначала системы координат следующим образом: x_p-y_p-z_p — начальная система координат, x'-y'-z' — система координат после первого поворота, x''-y''-z'' — система координат после второго поворота и x-y-z — конечная система координат. Тогда первый поворот на угол α соответствует повороту вокруг оси z_p, второй поворот на угол β — повороту вокруг оси y', третий поворот на угол γ — повороту вокруг оси z'' (см. рис. 2).



а *б*

Рис. 2. Начальная ориентация шестигранной призмы (*а*), углы Эйлера (*б*)

Алгоритм трассировки пучков

Задача алгоритма трассировки пучков заключается в нахождении матрицы Джонса из (1) и/или матрицы Мюллера из (2). Алгоритм работает рекурсивно.

На первом этапе, когда плоскопараллельная падающая волна попадает на плоскую грань кристалла, образуются отраженный и преломленный плоскопараллельные пучки света. Направление распространения каждого из образовавшихся пучков диктуется законами отражения/преломления. Геометрия пучков определяется гранью кристалла. Матрицы Джонса и фазы пучков рассчитываются для каждого пучка в своих локальных системах координат, представленных на рис. 3. Отметим, что

данная система координат совпадает с [26] и отличается от [25].

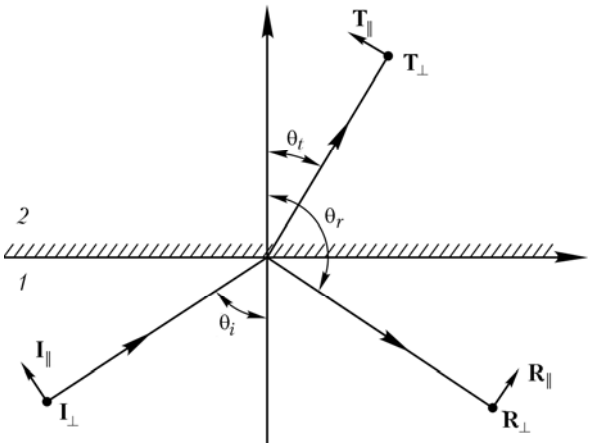


Рис. 3. Системы координат для расчета матриц Джонса отраженного и преломленного пучков

Если представить падающую волну в базисе $\{\mathbf{I}_\perp, \mathbf{I}_\parallel\}$, то легко определить амплитуды отраженной и преломленной волн в базисах $\{\mathbf{R}_\perp, \mathbf{R}_\parallel\}$ и $\{\mathbf{T}_\perp, \mathbf{T}_\parallel\}$ соответственно:

$$\mathbf{E}^{\mathbf{R}} = \mathbf{J}_{\mathbf{R}}\mathbf{E}^{\mathbf{I}} = \begin{pmatrix} F_\parallel^{\mathbf{R}} & 0 \\ 0 & F_\perp^{\mathbf{R}} \end{pmatrix}\mathbf{E}^{\mathbf{I}}, \qquad (14)$$

$$\mathbf{E}^{\mathbf{T}} = \mathbf{J}_{\mathbf{T}}\mathbf{E}^{\mathbf{I}} = \begin{pmatrix} F_\parallel^{\mathbf{T}} & 0 \\ 0 & F_\perp^{\mathbf{T}} \end{pmatrix}\mathbf{E}^{\mathbf{I}} \qquad (15)$$

($F_\parallel, F_\perp$ – коэффициенты Френеля). Для перехода из одной системы координат в другую используется матрица поворота, в частности для перехода из Системы координат падающего поля (3) в локальную систему координат $\{\mathbf{I}_\perp, \mathbf{I}_\parallel\}$ – матрица поворота **L:**

$$\mathbf{E}^{\mathbf{I}} = \mathbf{L}\mathbf{E}^{i}, \qquad (16)$$

где

$$\mathbf{L} = \begin{pmatrix} \cos\xi & \sin\xi \\ -\sin\xi & \cos\xi \end{pmatrix} \qquad (17)$$

и ξ – угол между базисами $\mathbf{e}_\perp^i$ и $\mathbf{I}_\perp$.

На втором этапе все отраженные от частицы (внешние) пучки в случае выпуклой частицы считаются покинувшими частицу и передаются обработчику пучков для их корректного учета согласно поставленной задаче. Все преломленные (внутренние) пучки считаются исходными и трассируются аналогично первому этапу с тем отличием, что здесь возможно полное внутреннее отражение, а форма пучка определяется пересечением исходного пучка и соответствующей грани кристалла. Последняя процедура, несмотря на кажущуюся простоту, вызывает основные трудности при реализации алгоритма трассировки пучков.

Фазовый сдвиг пучка вычисляется исходя из оптической длины пути пучка. Для этого падающая плоскопараллельная волна считается исходящей от плоскости, удаленной на расстояние D от центра Глобальной декартовой системы координат и перпендикулярной направлению падения (рис. 4).

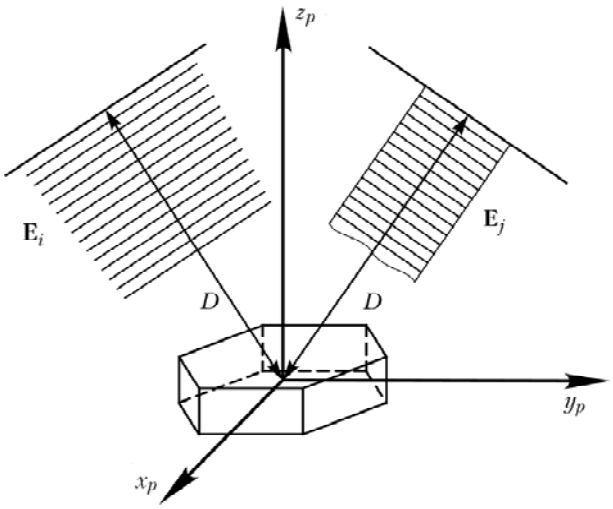


Рис. 4. Определение оптической длины пучка

Любой выходящий пучок также трассируется до плоскости, удаленной на расстояние D от центра Глобальной декартовой системы координат и перпендикулярной направлению выхода. В таком случае любой луч пучка имеет одинаковую оптическую длину ζ, равную его геометрической длине, с учетом всей трассировки, умноженной на соответствующий показатель преломления. Расстояние D задается пользователем и должно быть больше максимального размера частицы. Его значение влияет лишь на постоянный для всех пучков фазовый сдвиг, который в большинстве случаев не является существенным. Учет фазового сдвига достигается умножением матрицы Джонса **J** на множитель

$$\exp(i\psi) = \exp\left(i\frac{\xi}{\lambda}2\pi\right) \qquad (18)$$

(λ – длина волны света).

Таким образом, алгоритм трассировки пучков представляет рассеянное поле в виде суммы покинувших частицу пучков:

$$\mathbf{E}_j(\mathbf{r}) = \mathbf{J}_j\mathbf{E}^i\eta_j(\mathbf{r})\exp(i\psi_j)\exp(ik\mathbf{n}_j\mathbf{r}), \qquad (19)$$

где функция формы имеет вид

$$\eta_j(\mathbf{r}) = \begin{cases} 1 & \text{внутри } j\text{-го пучка,} \\ 0 & \text{вне его.} \end{cases} \qquad (20)$$

В дальней зоне поле записывается в следующем виде:

$$\mathbf{I}_s(\mathbf{n}) = \sum \mathbf{I}_j(\mathbf{n}) = \sum s_j\delta(\mathbf{n} - \mathbf{n}_j)\mathbf{M}_j\mathbf{I}_i, \qquad (21)$$

где **n** — направление рассеяния; δ — дельта-функция Дирака; s_j, $\mathbf{n}_j$, $\mathbf{M}_j$ — площадь, направление и матрица Мюллера j-го пучка соответственно.

Строго говоря, этот ряд является бесконечным, однако в большинстве случаев быстро сходится. Алгоритм останавливает рекурсию, когда ее длина превышает заданное пользователем значение. Более того, алгоритм не учитывает пучки, размер и энергия которых меньше заданного пользователем значения. Более подробную информацию можно найти в Руководстве пользователя.

Интерфейс программы

Существует большое количество приложений задачи рассеяния света. К ним можно отнести рассеяние света на хаотически или преимущественно ориентированных кристаллах, в направлении строго назад или по всей сфере направлений рассеяния, и т.п. Это ведет к невозможности написания одной универсальной программы, позволяющей решать любую задачу рассеяния света. При этом метод геометрической оптики, основанный на алгоритме трассировки пучков, для всех перечисленных задач одинаков. Такое положение дел привело авторов настоящей статьи к идее реализации алгоритма трассировки пучков в виде отдельной библиотеки, некоторого набора для разработчика (SDK), который легко применяется для решения конкретной задачи рассеяния. В качестве примера вместе с данной библиотекой поставляется готовый проект решения задачи рассеяния света на хаотически ориентированных ледяных частицах.

Представляемый авторами в свободном доступе с открытым исходным кодом метод геометрической оптики реализован на языке C++ с использованием только стандартных компонентов, что обеспечивает его кроссплатформенность и хорошую совместимость с популярными средами разработок. Алгоритм распространяется по лицензии GNU GPL. Поставляемый вместе с набором для разработчика проект решения задачи рассеяния света написан с использованием библиотек Qt 5.1 и также является кроссплатформенным. Проект однопоточный.

Для удобства опишем сначала проект. Он состоит из основного файла «main.cpp», файла параметров «params.dat», файла проекта Qt Creator «Random.pro», скомпилированного файла «random.exe», библиотеки алгоритма трассировки пучков «Lib\…», Руководства пользователя с описанием классов «refman.pdf».

Параметры задачи рассеяния света на хаотически ориентированных частицах задаются в текстовом файле параметров «params.dat». Каждая строка отвечает за свой параметр, который располагается до символа «//», после которого идет комментарий. Параметры приведены ниже.

Процесс решения задачи рассеяния выглядит следующим образом: задаются необходимые параметры в файле «params.dat». Проект компилируется и запускается. Файл параметров должен лежать в одной папке со скомпилированным бинарным файлом («random.exe» для Windows). На экране появится консольное окно, отображающее процесс решения. Результатом решения будут два файла «out.dat» и «M.dat», содержащие общие сведения о решении и ненулевые элементы матрицы Мюллера в зависимости от зенитного угла θ на Сфере направлений рассеяния.

Для решения произвольной задачи рассеяния света разработчику необходимо написать свой собственный файл «main.cpp» в соответствии с имеющимся в проекте файлом.

Параметры задачи рассеяния света

Номер параметра	Значение
1	Вид частицы. Существует 5 видов частиц: шестигранная призма, шестигранная пуля, пирамида, усеченная призма, чашеобразная призма
2	Определяет, использовать ли для задания параметров усечения призмы угол в 56°. Если данный параметр установлен в единицу, то параметры 5 и 6 игнорируются и усечение устанавливается под углом 56°
3	Радиус описанной вокруг основания частицы окружности
4	Полувысота частицы
5	Высота усечения частицы
6	Радиус усечения частицы
7	Вещественный показатель преломления
8	Количество поворотов частицы вокруг оси симметрии
9	Количество поворотов частицы относительно зенитного угла
10	Количество интервалов, на которые разделен зенитный угол рассеяния от 0 до 180° и, соответственно, количество матриц Мюллера в выходном файле
11	Глубина рекурсии для алгоритма трассировки пучков, где 0 — только зеркальное отражение
12	Позволяет исключить из решения пики вперед и назад
13	Определяет, будет ли частица поворачиваться на дискретный угол вокруг оси симметрии и относительно зенитного угла либо она будет поворачиваться псевдослучайным образом с использованием функции random()
14	Определяет, сколько траекторий пучков учитывать: 0 — все пучки, положительное целое число — заданное число траекторий пучков. Далее идет список траекторий

Разберем подробно ключевые особенности файла «main.cpp».

К файлу обязательно должен быть подключен файл «\Lib\particle.hpp», в котором находятся ссылки на подключение всех необходимых для алгоритма трассировки пучков библиотечных файлов, описание прототипов и функций. Желательно также подключение файлов: «\Lib\Mueller.hpp», содержащего описание функции вычисления матрицы Мюллера из матрицы Джонса согласно (12); «\Lib\PhysMtr.hpp», содержащего прототипы двумерных массивов вещественных и комплексных матриц; «\Lib\trajectory.hpp», содержащего прототипы списка траекторий, участвующих в расчете.

Необходимо добавить описание прототипа функции-обработчика вышедших пучков

```
void Handler(Beam& bm);
```

и реализовать ее в соответствии с решаемой задачей.

Работа с алгоритмом трассировки пучков начинается с создания частицы, для которой будет строиться решение, например:

```
Crystal* Body = NULL;

Body = new Prism(RefIndex, Radius, Halh_Height,
                 Iteracii, i, Et, E_min, d, S_min).
```

Здесь задаются: показатель преломления, размеры частицы, глубина рекурсии для алгоритма трассировки, направление падения света $\mathbf{i}$ и базисный вектор $\mathbf{e}_{\perp}^{i}$, минимальная энергия пучка, при которой он отбрасывается, расстояние D для расчета оптической длины, минимальная площадь пучка, при которой он отбрасывается.

Далее необходимо указать, нужно ли вычислять оптический путь для расчета фазы вышедшего пучка света:

```
Body->Phase( ) = false;
```

и, если необходимо, повернуть частицу на углы Эйлера:

```
Body->ChangePosition(betta, gamma, alpha).
```

Процесс трассировки запускается командой

```
Body->FTforConvexCrystal(Handler);
```

которой передается параметр – указатель на функцию-обработчик вышедших пучков. Функция возвращает текущее для данной ориентации сечение рассеяния.

В результате приведенных выше действий запустится алгоритм трассировки пучков, который в рамках приближения геометрической оптики определит направление всех выходящих из частицы пучков, их поперечное сечение, матрицу Джонса и т.д. Пользователю предоставляется полная свобода по обработке вышедших пучков в функции-обработчике.

Функция-обработчик принимает каждый вышедший пучок (bm)

```
void Handler(Beam& bm);
```

и имеет полную информацию о нем, например площадь сечения

```
CrossSection(bm);
```

матрицу Джонса

```
bm( );
```

направление выхода

```
bm.r;
```

оптическую длину

```
bm.lng;
```

и т.п. (см. Руководство пользователя).

В поставляемом примере программа поворачивает частицу заданное количество раз вокруг своей оси и заданное количество раз наклоняет частицу. Запускает трассировку, отлавливает все вышедшие пучки и суммирует их матрицы Мюллера в выходном массиве. Затем этот массив выводится в файл «M.dat».

Для удобства файл «main.cpp» снабжен комментариями.

Большинство геометрических примитивов, используемых в программе, взяты из книги M. Laszlo [27] и существенно доработаны.

Заключение

Представлено подробное изложение разработанного авторами алгоритма трассировки пучков. Алгоритм существенно отличается от активно используемого в настоящее время алгоритма трассировки лучей. Особенности рассеяния света на кристаллических частицах позволили существенно снизить требования алгоритма трассировки пучков к вычислительным ресурсам по сравнению с алгоритмом трассировки лучей. Отсутствие у алгоритма трассировки пучков многих недостатков, присущих алгоритму трассировки лучей, обусловленных стохастичностью начальных положений лучей, делает его более перспективным для решения задачи рассеяния света на кристаллических частицах. Реализация представляемого алгоритма в виде отдельной библиотеки и наличие его в свободном доступе существенно облегчают его внедрение в сторонние проекты.

Работа выполнена при поддержке гранта РФФИ № 15-05-06100а, частичной поддержке РНФ (соглашение № 14-27-00022), при поддержке гранта Президента РФ (МК-6680.2015.5) и Программы повышения конкурентоспособности ТГУ.

1. *Mishchenko M.I., Hovenier J.W., Travis L.D.* Light Scattering by Nonspherical Particles: Theory, Measurements, and Geophysical Applications. San Diego: Academic Press, 1999. 690 p.
2. *Bi L., Yang P.* Physical-geometric optics hybrid methods for computing the scattering and absorption properties of ice crystals and dust aerosols // Light Scattering Reviews 8 / Ed. by A.A. Kokhanovsky. Chichester: Springer-Praxis, 2013. P. 69–114.

3. *van de Hulst H.C.* Light scattering by small particles. N.Y.: Dover, 1981. 470 p.
4. *Takano Y., Liou K.N.* Solar radiative transfer in cirrus clouds. Part I. Single scattering and optical properties of hexagonal ice crystals // J. Atmos. Sci. 1989. V. 46, N 1. P. 3–19.
5. *Bi L., Yang P., Liu C., Yi B., Baum B.A.* Assessment of the accuracy of the conventional ray-tracing technique: Implications in remote sensing and radiative transfer involving ice clouds // J. Quant. Spectrosc. Radiat. Transfer. 2014. V. 146. P. 158–174.
6. *Borovoi A., Konoshonkin A., Kustova N.* The physics-optics approximation and its application to light backscattering by hexagonal ice crystals // J. Quant. Spectrosc. Radiat. Transfer. 2014. V. 146. P. 181–189.
7. *Коношонкин А.В., Кустова Н.В., Боровой А.Г.* Граница применимости приближения геометрической оптики для решения задачи обратного рассеяния света на квазигоризонтально ориентированных гексагональных ледяных пластинках // Оптика атмосф. и океана. 2014. Т. 27, № 8. С. 705–712.
8. *Jacobowitz H.* A method for computing the transfer of solar radiation through clouds of hexagonal ice crystals // J. Quant. Spectrosc. Radiat. Transfer. 1971. V. 11, N 6. P. 691–695.
9. *Wendling P., Wendling R., Weickmann H.K.* Scattering of solar radiation by hexagonal ice crystals // Appl. Opt. 1979. V. 18, N 15. P. 2663–2671.
10. *Cai Q., Liou K.N.* Polarized light scattering by hexagonal ice crystals: Theory // Appl. Opt. 1982. V. 21, N 19. P. 3569–3580.
11. *Hess M., Wiegner M.* COP: A data library of optical properties of hexagonal ice crystals // Appl. Opt. 1994. V. 33, N 33. P. 7740–7746.
12. *Macke A., Mueller J., Raschke E.* Single scattering properties of atmospheric ice crystal // J. Atmos. Sci. 1996. V. 53, N 19. P. 2813–2825.
13. *Muinonen K., Lamberg L., Fast P., Lumme K.* Ray optics regime for Gaussian random spheres // J. Quant. Spectrosc. Radiat. Transfer. 1997. V. 57, N 2. P. 197–205.
14. *Borovoi A., Grishin I., Naats E., Oppel U.* Light backscattering by hexagonal ice crystals // J. Quant. Spectrosc. Radiat. Transfer. 2002. V. 72, N 4. P. 403–417.
15. *Macke A.* Scattering of light by polyhedral ice crystals // Appl. Opt. 1993. V. 32, N 15. P. 2780–2788.
16. *Flatau P.J., Draine B.T.* Light scattering by hexagonal columns in the discrete dipole approximation // Opt. Exp. 2014. V. 22, N 18. P. 21834–21846.
17. *Neshyba S.P., Lowen B., Benning M., Lawson A., Rowe P.M.* Roughness metrics of prismatic facets of ice // J. Geophys. Res. A. 2013. V. 118, N 8. P. 3309–3318.
18. *Yang P., Liou K.N.* Geometric-optics-integral-equation method for light scattering by nonspherical ice crystals // Appl. Opt. 1996. V. 35, N 33. P. 6568–6584.
19. *Masuda K., Ishimoto H., Mano Y.* Efficient method of computing a geometric optics integral for light scattering by nonspherical particles // Pap. Meteorol. Geophys. 2012. V. 63. P. 15–19.
20. *Попов А.А.* Рассеяние электромагнитной плоской волны на полупрозрачном выпуклом многограннике произвольной формы // Изв. вузов. Физ. Депон. № 8006. 1984. 56 с.
21. *Del Guasta M.* Calcolo delle proprieta' ottiche dei cristalli di ghiaccio mediante il metodo del tracciamento dei raggi, Applicazione al LIDAR a retrodiffusione // Technol. Rep. TR/GCF/95.04. 1995. IROE CNR, Florence. 42 p.
22. *Ромашов Д.Н.* Рассеяние света гексагональными ледяными кристаллами // Оптика атмосф. и океана. 2001. Т. 14, № 2. С. 116–124.
23. *Borovoi A.G., Grishin I.A.* Scattering matrices for large ice crystal particles // J. Opt. Soc. Amer. A. 2003. V. 20, N 11. P. 2071–2080.
24. *Алгоритм* трассировки пучков. URL: https://github.com/sasha-tvo/Beam-Splitting
25. *Bohren C.F., Huffman D.R.* Absorption and Scattering of Light by Small Particles. New York: Wiley, 1983. 530 p.
26. *Born M., Wolf E.* Principles of Optics (4th ed.). Great Britain: Pergamon Press, 1970. 808 p.
27. *Laszlo M.J.* Computational geometry and computer graphics in C++. Lebanon: Prentice-Hall, 1995. 266 p.

A.V. Konoshonkin, N.V. Kustova, A.G. Borovoi. **Beam splitting algorithm for light scattering by atmospheric ice crystals. Part 1. Theory.**

The article discusses the theoretical basis of the beam splitting algorithm for the problem of light scattering by atmospheric ice crystals in the geometrical optics approximation. The paper considers the solution of light scattering problem in terms of Jones and Muller matrices, which is the basis of the beam splitting algorithm. Special attention is paid to the interface of the software implementation of the algorithm. The developed algorithm is freely available as open source software.