ВЕКТОРИЗАЦИЯ МЕТОДА РАСПРОСТРАНЯЮЩЕГОСЯ ПУЧКА И ЕГО РЕАЛИЗАЦИЯ ПО ТЕХНОЛОГИИ CUDA

Алексеев В.А. [2], *Головашкин Д.Л.* [1]
[1] *Институт систем обработки изображений РАН,*
[2] *Самарский государственный аэрокосмический университет им. С.П.Королева*

Аннотация

Разработан векторный алгоритм метода решения уравнения Гельмгольца (BPM подход), основанный на представлении вычислений в модели SIMD (Single Instruction, Multiple Data). Реализация данного алгоритма на графическом процессоре NVIDIA GeForce GTS 250 по технологии CUDA продемонстрировала ускорение вычислений в 22,5 раза по сравнению с расчетами на центральном процессоре Intel Core Duo E7500.

Ключевые слова: уравнение Гельмгольца, BPM метод, векторный алгоритм, графический процессор, покомпонентное произведение, CUDA.

Введение

Развитие оптических технологий определяет необходимость совершенствования известных и создания новых методов моделирования распространения электромагнитного излучения. В области волоконной и интегральной оптики получили широкую известность варианты метода распространяющегося пучка (BPM – Beam propagation method): Fast Fourier transform Beam propagation method (FFT-BPM) [1], Method of lines Beam propagation method (MoL-BPM) [2], Finite-difference Beam propagation method (FD-BPM) [3], Finite elements Beam propagation method (FE-BPM) [4], модифицированный метод распространяющегося пучка [5]. Объединяет упомянутые варианты дискретизация функции амплитудно-фазового распределения поля и численное решение уравнения Гельмгольца относительно этой функции с заданными начальными условиями. Популярность метода объясняется простотой реализации и достаточной строгостью подхода в приведенных областях оптики.

К сожалению, развитие метода сдерживается его высокой вычислительной сложностью. Так, производители известного пакета OlympIOs [6] вынуждены включить в него модуль для параллельных вычислений. Компания RSoft предлагает различные способы оптимизации параметров и решения под требования заказчика [7]. К настоящему времени известна альтернативная технология снижения длительности расчетов, основанная на векторизации алгоритма. Такие алгоритмы известны достаточно давно [8,9], однако до последнего времени не получили распространения в силу недоступности соответствующей аппаратной базы широкому кругу исследователей. Технология CUDA [10], получившая всеобщее признание с 2007 года, открывает возможность реализации векторных алгоритмов на обычных персональных компьютерах.

С ее появлением вернулся интерес к векторизации хорошо известных численных методов: Гаусса-Зайделя [11], FDTD [12], Ray Tracing [13] и многих других. Для BPM подхода аналогичные исследования в доступной литературе найдены не были, данной работой авторы рассчитывают заполнить этот пробел.

1. Векторный алгоритм для разностной схемы Кранка-Николсона

Представляя скалярный вариант BPM подхода (двумерный случай), принято [3] рассматривать уравнение Гельмгольца в виде:

$$\frac{\partial^2 U}{\partial z^2}+\left(\frac{\partial^2}{\partial y^2}+k_0^2 n^2\right)U=0, \qquad (1)$$

где под величиной U, определенной на вычислительно области $D=\{(y,z):0\le y\le L_y;0\le z\le L_z\}$, понимают напряженность электрического поля (в случае TE-поляризованного излучения) или напряженность магнитного (в случае TM-поляризованного излучения). Используя приближение «медленно меняющейся огибающей» [2], пренебрегают значением второй производной по z, тем самым понижая порядок дифференцирования. Тогда уравнение (1) представляют как:

$$\frac{\partial \Psi}{\partial z}=\frac{i}{2k_0 n_*}\left(k_0^2 n_*^2-k_0^2 n^2-\frac{\partial^2}{\partial y^2}\right)\Psi, \qquad (2)$$

где $\Psi(y,z)=U(y,z)\exp(ik_0 n_* z)$, k_0 – волновое число в вакууме, n_* – базовый показатель преломления, характеризующий точность огибающей. При этом распространение излучения в неоднородной среде считают сходным с распространением в однородной среде с показателем преломления n_*. Уравнение (2) называют уравнением Гельмгольца в форме Френеля.

Записывая разностную схему для (2), наложим на D следующую сеточную область:

$$D_h=\left\{(y_i,z_k):y_i=ih_y,h_y=\frac{L_y}{N};z_k=kh_z,h_z=\frac{L_z}{K}\right\}, \qquad (3)$$

где h_y и h_z – шаги дискретизации, характеризующие сходимость разностного решения. Тогда схема Кранка-Николсона для уравнения (2) [14] при ис-

следовании распространения излучения в свободном пространстве ($n_* = n_0$) примет вид [3]:

$$\frac{\Psi_i^{k+1} - \Psi_i^k}{h_z} = \frac{i\theta}{2k_0 n_*}\left(\frac{\Psi_{i+1}^k - 2\Psi_i^k + \Psi_{i-1}^k}{h_y^2}\right) + \\ + \frac{i(1-\theta)}{2k_0 n_*}\left(\frac{\Psi_{i+1}^{k+1} - 2\Psi_i^{k+1} + \Psi_{i-1}^{k+1}}{h_y^2}\right), \quad (4)$$

где θ – весовой коэффициент разностной схемы. Введение указанного упрощения не влияет на векторизацию вычислений и преследует исключительно методические цели.

Рассмотрим в качестве начального условия поле $\Psi(y,0)$ при $z=0$, записав следующую его дискретизацию $\Psi_i^0 = \Psi_0(ih_y)$. Краевые условия соответствуют расположению идеального проводника на границах D при $y=0$ и $y=L_y$: $\Psi_0^k = \Psi_N^k = 0$.

Приняв весовой коэффициент θ =0,5 [14], перепишем (4) как:

$$a\left(\Psi_{i+1}^{k+1} + \Psi_{i-1}^{k+1}\right) - (1+2a)\Psi_i^{k+1} = -F_i^k, \quad (5)$$

где $F_i^k = a\left(\Psi_{i+1}^k + \Psi_{i-1}^k\right) + (1-2a)\Psi_i^k$ и $a = \frac{ih_z}{4k_0 n_* h_y^2}$.

Задача свелась к решению системы линейных алгебраических уравнений (СЛАУ) вида $Ax = b$ с ленточной матрицей. Приступая к векторизации вычислений по (5), выберем для решения системы метод однопараметрической итерации [14]. К сожалению, как отмечено в монографиях Дж. Голуба и В. Лоуна [8] и Дж. Ортеги [9], прямые методы решения СЛАУ с узкой лентой эффективно не векторизуются. В выбранном методе приближение к решению x^{s+1} (на $s+1$ итерации) равно произведению матрицы P на вектор приближенного решения x^s (на s итерации), сложенный с произведением вектора правой части b на параметр τ. При этом $P = (A - \tau E)$, E – единичная матрица, $\tau = 2/(\lambda + \Lambda)$, λ и Λ – минимальное и максимальное собственные числа матрицы A соответственно. Такой выбор параметра τ обеспечивает оптимальную скорость сходимости [15]. При моделировании среды с меняющимся показателем преломления, возникает необходимость в пересчете собственных чисел матрицы A. Авторы полагают возможным в силу плавности изменения среды производить пересчет не на каждом пространственном слое разностной схемы, а через определенные промежутки. Тогда доля длительности расчета значений λ и Λ в общем времени вычислений может быть минимизирована.

Рассмотрим следующую схему компактного хранения матрицы по диагоналям. Ранее Дж. Голуб и В. Лоун [8] применяли аналогичный подход для хранения симметричной матрицы.

Пусть матрица A имеет вид (рис. 1):

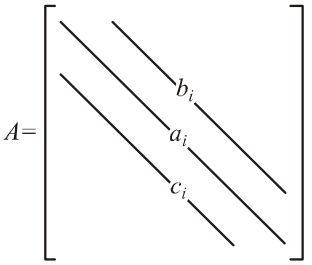

Рис. 1. Ленточная матрица A, где верхняя ширина ленты и нижняя ширина ленты q=p=1, соответственно $1 \le i \le N$ для a_i, $1 \le i \le N-1$ для b_i и c_i

Тогда при хранении по диагоналям представим матрицу A в виде:

$$A_{diag} = [a_1, \ \cdots, \ a_N], \\ B_{diag} = [b_1, \ \cdots, \ b_{N-1}], \quad (6) \\ C_{diag} = [c_1, \ \cdots, \ c_{N-1}].$$

Выбирая более сложный дифференциальный шаблон, можно придти к матрице с большей шириной верхней и нижней ленты, для общности положим ширину верхней и нижней ленты произвольной и равной p и q соответственно. Тогда элемент матрицы $A(i,j)$ хранится в одном из векторов A_{diag}, B_{diag} и C_{diag} в соответствии со следующим правилом:

$$A(i,j) = \begin{cases} A_{diag}(i), & i = j, \\ A(j+h, j) = C_{diag}((h-1)N - \\ \quad -h(h-1)/2 + j), & i \ge j+h, \\ A(i, i+m) = B_{diag}((m-1)N - \\ \quad -m(m-1)/2 + i), & i \le j-m, \end{cases} \quad (7)$$

где h и m - номера диагоналей в нижней и верхней ленте, для которых справедливо неравенство $1 \le h < p$ и $1 \le m < q$.

Умножение матрицы на вектор $z = Ax$ (основная операция в методе) может быть записана следующим образом в нотации из [8].

Алгоритм умножения матрицы на вектор:

$z = A_{diag}.* x$ %умножение главной диагонали на вектор x

for $m = 1 : q$ %проход по верхней ленте

$t = (m-1)N - m(m-1)/2$ % номер 1-го элемента поддиагонали в массиве, хранящем верхние ленты

$z(1 : N-m) = z(1 : N-m) +$

$+ B_{diag}(t+1 : t+N-m).* x(m+1 : N);$

end

for $h = 1 : p$ %проход по нижней ленте

$t = (h-1)N - h(h-1)/2$ % номер 1-го элемента поддиагонали в массиве, хранящем нижние ленты

$$z(h+1:N) = z(h+1:N) +$$
$$+C_{diag}(t+1:t+N-h).*x(1:N-h);$$

end

Правило хранения матрицы по диагоналям и векторный алгоритм представлены для ленточных матриц, с произвольной шириной верхней и нижней ленты. Заметим, что во внутренних циклах происходит покомпонентное произведение векторов, обозначенное в алгоритме «.*» (на многих процессорах производится аппаратно), и осуществляется доступ к последовательно расположенным в памяти данным (что значительно ускоряет вычисления).

2. Реализация векторного алгоритма на видеокарте NVIDIA GeForce GTS 250

Результаты, представленные далее, получены на видеокарте GeForce GTS 250.

Таблица 1. Основные характеристики GPU NVIDIA GeForce GTS 250

Характеристика	Значение
Количество мультипроцессоров, шт.	16
Размер видеопамяти, Мб	1024
Максимальное число потоков в блоке, шт.	512
Максимальная размерность блока потоков (*x*, *y*, *z*), шт.	512*512*64
Максимальная размерность сетки блоков, шт.	65535 * 65535 * 1
Тактовая частота ядра, МГц	702
Тактовая частота памяти, МГц	1000

Кроме того, использовался процессор Intel Core Duo E7500.

Таблица 2. Основные характеристики CPU Intel Core Duo E7500

Характеристика	Значение
Тактовая частота ядра, ГГц	2,93
Тактовая частота шины CPU, МГц	1066
Кеш L1, Кб	64*2
Кеш L2, Кб	3072
Пропускная способность шины процессор-чипсет, ГБ/с	8,528
Ширина шины L2 кеша, бит	256

В вычислительных экспериментах по определению ускорения моделирование распространения электромагнитного излучения проводилось для полого волновода с идеальными проводящими стенками. В качестве начального условия была взята мода волновода:

$$\Psi(y,0) = \sin\left(\pi\frac{yn}{L_y}\right),$$

где L_y – его ширина. Данные условия выбраны для простоты верификации разностного решения. Длина волны принималась равной $\lambda = 1$ мкм, ширина волновода $L_y = 10$ мкм, $L_z = 100$ мкм, показатель преломления среды $n = 1$.

Число узлов сеточной области *N* менялось от 100 до 3000. Соответственно $h_y = L_y / N$, $h_z = h_y / \pi$ (для асимптотической устойчивости [14]).

Исследование велось в операционной системе Microsoft Windows XP, 32-bit (Service Pack 3) с установленным драйвером NVIDIA CUDA 2.3 driver. Скалярный и векторный алгоритмы написаны с использованием программного пакета Microsoft Visual Studio 2008 Express Edition.

Приведем ядро (kernel в терминах [10]) для GPU, выполняющее основную операцию описанного выше векторного алгоритма (умножение матрицы на вектор).

```
__global__ void
matrixMulfVextor( cuFloatComplex* d_Bdiag,
cuFloatComplex* d_Cdiag, cuFloatComplex*
d_Adiag, cuFloatComplex* d_x0, cuFloatComplex*
d_bt, cuFloatComplex* d_x, int N)

//номер элемента в блоке
{int i = blockIdx.x * blockDim.x+threadIdx.x;

//умножение главной диагонали на вектор x
и суммирование с вектором правой части.
if(i<N){
    d_x[i]=cuCaddf(cuCmulf(d_Adiag[i],d_x0[i]),
d_bt[i]);
} __syncthreads();

//умножение верхней ленты на вектор x
 if(i<N-1){
    d_x[i]=cuCsubf(d_x[i],
    cuCmulf(d_Bdiag[i],d_x0[i+1]));
} __syncthreads();

// умножение нижней ленты на вектор x
if(i>0 && i<N) {
d_x[i]=cuCsubf(d_x[i],
    cuCmulf(d_Cdiag[i-1],d_x0[i-1]));
} __syncthreads();

// переход с k(итерации) на k+1(итерацию)
if(i<N) {
    d_x0[i].x = d_x[i].x; d_x0[i].y = d_x[i].y;
}__syncthreads();
```

На вход в ядро передаются вектора главной диагонали (A_{diag} из (6)), верхней и нижней лент (B_{diag} и C_{diag} из (6)) и правой части. Размер блока равен 256 потокам (thread в терминах [10]), размер сетки (grid в терминах [10]) выбирался равным $N/256$. На каж-

дом мультипроцессоре может исполняться совокупность потоков до 24 (24 warp в терминах [10]) или 768 потоков (размер warp = 32 потока). За счет этого и достигается высокая производительность.

В скалярном варианте авторы остановились на итерационном методе решения (5) и для CPU как на методе не только обеспечивающем корректное сравнение эффективности обеих реализаций (векторной и скалярной), но и позволяющем в будущем использовать векторные инструкции SSE центрального процессора.

На рис. 2 виден эффект от применения векторных вычислений.

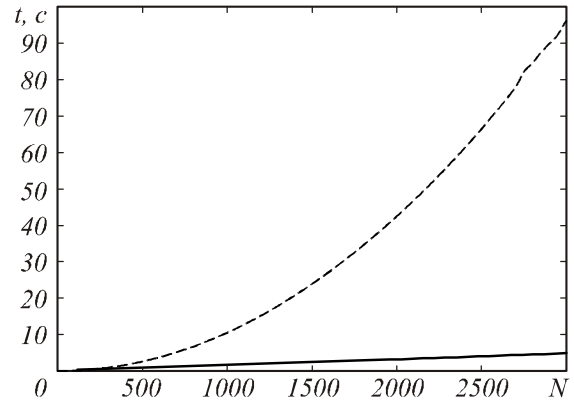

Рис. 2. Время выполнения BPM метода (пунктирной линией) на CPU и (сплошной) на GPU

Зависимость времени выполнения от размерности задачи для CPU параболическая, а для GPU - линейная. Это объясняется тем, что с ростом размерности задачи число скалярных операций увеличивается пропорционально квадрату размерности сеточной области, а число векторных операций растет линейно. Существенной разницы на участке размерности от 100 до 400 узлов сетки нет, что связано с возможностью центрального процессора быстро обрабатывать небольшое количество данных, которое может быть помещено в его кеш (cache). С ростом размерности задачи такая возможность исчезает и тысячи процессорных тактов уходят на пересылку данных из оперативной памяти в кеш и обратно. Кроме того, центральный процессор выполняет ряд системных инструкций, снижающих производительность. Длительность выполнения вычислений для одного шага по z на GPU в 22,5 раз меньше аналогичной операции для CPU для размерности исходной матрицы $N*N=3000*3000$. Проведенные исследования для значений $n>15000$ показали появление параболической зависимости длительности вычислений от размерности задачи и для векторного алгоритма, что связано с ограниченным размером памяти видеокарты. Однако среди реализаций BPM, встреченных авторами в литературе, отсутствуют сеточные области с упомянутой размерностью. Следует отметить, что, хотя векторное ускорение, получаемое при выполнении кода на видеокарте, велико, это вполне ожидаемый результат, учитывая особенности архитектуры GPU. В отличие от CPU, где большая часть ядра отдана под кеш и сложные арифметико-логические устройства (АЛУ), на графическом ядре размещено большое количество упрощенных АЛУ, которые имеют общую память на кристалле.

Заключение

В настоящей работе создан векторный алгоритм для решения уравнений разностной схемы Кранка-Николсона. Данный алгоритм реализован с использованием технологии CUDA применительно к BPM-FD методу. Сравнение векторного алгоритма, выполняемого на GPU, со скалярным аналогом на CPU продемонстрировало увеличение производительности в 22,5 раза для размерности задачи 3000 * 3000. Таким образом, представляется перспективным использование технологии CUDA для вычислений и по другим BPM методам.

Благодарности

Работа выполнена при поддержке грантов РФФИ № 10-07-00553-а, 10-07-00453-а и 10-01-00723-а и гранта Президента РФ №НШ-7414.2010.9.

Литература

1. **Feit, M.D.** Light Propagation in Graded-Index Optical Fibers / M.D. Feit, J.A. Fleck // Applied Optics. – 1978. – Vol. 17, N. 24. – P. 3990-3998.
2. **Gerdes, J.** Beam-propagation algorithm based on the method of lines / J. Gerdes, R. Pregla // Journal of the Optical Society of America B. – 1991. – Vol. 8, N. 2. – P. 389-394.
3. **Huang, W.** The Finite-Difference Vector Beam Propagation Method: Analysis and Assessment / W. Huang, C. Xu, S.-T. Chu, S.K. Chaudhuri // Journal of Lightwave Technology. – 1992. – Vol. 10, N. 3. – P. 295-305.
4. **Lu, Y.Y.** Some Techniques for Computing Wave Propagation in Optical Waveguides / Y.Y. Lu // Communications in Computational Physics. – 2006. – Vol. 1, N. 6. – P. 1056-1075.
5. **Гаврилов, А.В.** Модифицированный метод распространяющегося пучка и его применение к расчету распространения в волноводах с изменяющимся профилем показателя преломления / А.В. Гаврилов // Компьютерная оптика. – 2008. – Т. 32, № 1. – С. 15-22.
6. OlympIOs: design, simulation and mask layout platform //URL: http://www.c2v.nl/products/software/olympios-software.shtml.
7. RSoft: Photonic Component Design Suite // URL: http://www.rsoftdesign.com/uicontrols/products/brochures/RSoftProductCatalog.pdf.
8. **Голуб, Дж.** Матричные вычисления / Дж. Голуб, Ч. Ван Лоун – М.: Мир, 1999. – 548 с.
9. **Ортега, Дж.** Введение в параллельные и векторные методы решения линейных систем / Дж. Ортега – М.: Мир, 1991. – 364 с.
10. CUDA, Published by NVIDIA Corporation – 2701 San Tomas Expressway Santa Clara, CA 95050, http://www.nvidia.com/object/cuda_home_new.html
11. **Евстигнеев, Н.М.** Интегрирование уравнения Пуассона с использованием графического процессора технологии // Вычислительные методы и программирование. – 2009. – Т. 10, № 2. – С. 82-87.
12. **Price, D.K.** GPU-based accelerated 2D and 3D FDTD solvers / D.K. Price, J.R. Humphrey, and E.J. Kelmelis –

Physics and Simulation of Optoelectronic Devices XV, vol. 6468 of Proceedings of SPIE, San Jose, Calif, USA, January 2007.
13. **Adinetz, A.** Implementing Classical Ray Tracing on GPU – a Case Study of GPU Programming / A. Adinetz, S. Berezin – Proceedings of Graphicon, 2006.
14. **Самарский, А.А.** Введение в численные методы / А.А. Самарский – М.: Наука, 1987.– 286 с.
15. **Косарев, В.И.** 12 лекций по вычислительной математике. Вводный курс / В.И. Косарев – М.: Физматлит, 2000. – 224 с.

References

1. **Feit, M.D.** Light Propagation in Graded-Index Optical Fibers / M.D. Feit, J.A. Fleck // Applied Optics. – 1978. – Vol. 17, N. 24. – P. 3990-3998.
2. **Gerdes, J.** Beam-propagation algorithm based on the method of lines / J. Gerdes, R. Pregla // Journal of the Optical Society of America B. – 1991. – Vol. 8, N. 2. – P. 389-394.
3. **Huang, W.** The Finite-Difference Vector Beam Propagation Method: Analysis and Assessment / W. Huang, C. Xu, S.-T. Chu, S.K. Chaudhuri // Journal of Lightwave Technology. – 1992. – Vol. 10, N. 3. – P. 295-305.
4. **Lu, Y.Y.** Some Techniques for Computing Wave Propagation in Optical Waveguides / Y.Y. Lu // Communications in Computational Physics. – 2006. – Vol. 1, N. 6. – P. 1056-1075.
5. **Gavrilov, A.V.** The modified method of beam propagation and its application to distribution calculation in wave guides with a changing profile of an indicator of refraction / A.V. Gavrilov // Computer Optics. – 2008. – V. 32, N 1. – P. 15-22.
6. OlympIOs: design, simulation and mask layout platform //URL: http://www.c2v.nl/products/software/olympios-software.shtml.
7. RSoft: Photonic Component Design Suite // URL: http://www.rsoftdesign.com/uicontrols/products/brochures/RSoftProductCatalog.pdf.
8. **Golub, G.** Matrix Computations / G. Golub, C. Van Loan – Moscow: “Mir” publisher, 1999. – 548 p. – (in Russian).
9. **Ortega J.** Introduction to parallel and vector solution of linear systems / J. Ortega – Moscow: “Mir” publisher, 1991. – 364 p. – (in Russian).
10. CUDA, Published by NVIDIA Corporation – 2701 San Tomas Expressway Santa Clara, CA 95050, http://www.nvidia.com/object/cuda_home_new.html
11. Evstigneev, N.M. Integration of the equation of Puassona with use of the graphic processor of technology // Computing methods and programming. – 2009. – T. 10, N 2. – P. 82-87. – (in Russian).
12. **Price, D.K.** GPU-based accelerated 2D and 3D FDTD solvers / D.K. Price, J.R. Humphrey, and E.J. Kelmelis //– Physics and Simulation of Optoelectronic Devices XV, vol. 6468 of Proceedings of SPIE, San Jose, Calif, USA, January 2007.
13. **Adinetz, A.** Implementing Classical Ray Tracing on GPU – a Case Study of GPU Programming / A. Adinetz, S. Berezin – Proceedings of Graphicon, 2006.
14. **Samarskii, A.A.** Introduction in numerical methods /A.A. Samarskii – Moscow: “Nayka” publisher, 1987. – 286 p. – (in Russian).
15. **Kosarev, V.I.** 12 lectures on calculus mathematics. An introduction course / V.I. Kosarev – Moscow: “Fizmatlit” publisher, 2000. – 224 p. – (in Russian).

VECTORIZATION OF THE BEAM PROPAGATION METHOD USING CUDA TECHNOLOGY

V. A. Alekseeev [1], D. L. Golovashkin[2]
[1] S.P. Korolyov Samara State Aerospace University,
[2] Image Processing Systems Institute of the RAS

Abstract

We propose a vector algorithm for solving the helmholtz equation through BPM (beam propagation method) by representing the computation in the SIMD (single instruction, multiple data) model. The implementation of the given algorithm on graphic processor NVIDIA GeForce GTS 250 on technology CUDA has shown acceleration of calculations in 22.5 times in comparison with the computation on the central processor Intel Core Duo E7500.

Key words: helmholtz equation, beam propagation method, vector algorithm, graphics processor, componentwise product, cuda.

Сведения об авторах

Алексеев Вадим Александрович, аспирант кафедры наноинженерии, Самарский государственный аэрокосмический университет им. С.П. Королева. E-mail: *alekseev.v.a@inbox.ru.*

Область научных интересов: векторные и параллельные матричные вычисления, волоконная оптика и фотонные кристаллы.

Vadim Aleksandrovich Alekseev, the post-graduate student of chair nanoengineering the Samara State Space University of S.P.Korolev. E-mail: *alekseev.v.a@inbox.ru.*

Scientific interests: vector and parallel matrix calculations, fiber optics and photon crystals.

Головашкин Димитрий Львович, доктор физико-математических наук, доцент, старший научный сотрудник Института систем обработки изображений РАН. E-mail: *dimitriy@smr.ru*. Область научных интересов: разностное решение уравнений Максвелла (FDTD метод); дифракционная оптика; векторные и параллельные матричные вычисления.

Dimitry Lvovich Golovashkin, Doctor of Physical and Mathematical Sciences, Associate Professor, Senior Researcher of Image Processing Systems Institute Russian Academy of Scienes. E-mail: *dimitriy@smr.ru*. Scientific interests: FDTD method, subwave optics, vector and parallel algorithms for matrix computation.

Поступила в редакцию 28 апреля 2010 г.