Journal of Biomedical Optics

BiomedicalOptics.SPIEDigitalLibrary.org

Acceleration of Monte Carlo simulation of photon migration in complex heterogeneous media using Intel many-integrated core architecture

Anton V. Gorshkov
Mikhail Yu. Kirillin

SPIE.

Journal of Biomedical Optics 20(8), 085002 (August 2015)

Acceleration of Monte Carlo simulation of photon migration in complex heterogeneous media using Intel many-integrated core architecture

Anton V. Gorshkov[a,b] and Mikhail Yu. Kirillin[a,b,*]
[a]Institute of Applied Physics of RAS, Ulyanov Street, 46, Nizhny Novgorod 603950, Russia
[b]N.I. Lobachevsky State University of Nizhny Novgorod, Gagarin Street, 23, Nizhny Novgorod 603000, Russia

Abstract. Over two decades, the Monte Carlo technique has become a gold standard in simulation of light propagation in turbid media, including biotissues. Technological solutions provide further advances of this technique. The Intel Xeon Phi coprocessor is a new type of accelerator for highly parallel general purpose computing, which allows execution of a wide range of applications without substantial code modification. We present a technical approach of porting our previously developed Monte Carlo (MC) code for simulation of light transport in tissues to the Intel Xeon Phi coprocessor. We show that employing the accelerator allows reducing computational time of MC simulation and obtaining simulation speed-up comparable to GPU. We demonstrate the performance of the developed code for simulation of light transport in the human head and determination of the measurement volume in near-infrared spectroscopy brain sensing. [DOI: 10.1117/1.JBO.20.8.085002]

Keywords: photon transport; simulation; Monte Carlo technique; scattering; many-integrated core architecture; Intel Xeon Phi.
Paper 150267R received Apr. 21, 2015; accepted for publication Jul. 6, 2015; published online Aug. 7, 2015.

1 Introduction

Monte Carlo (MC) simulations have become a gold standard tool in biomedical optics since presentation of classical MC modeling of light (MCML) code developed by Wang et al.[1] The simulations can be efficiently employed in situations where application of theoretical approaches, such as diffusion theory, is limited. This occurs when studying light propagation at comparatively small distances in biotissues, which are characterized by high anisotropy factor ($g > 0.8$).

The MCML code provides modeling of light transport in multilayered heterogeneous tissues with plane-parallel geometry of layers.[1] Later Boas et al. created an MC modeling system called "tMCimg" to address complex geometry of layers based on cubic voxelization of the considered medium;[2,3] however, such approaches do not provide an accurate account of photon refraction at the layers boundary. A similar approach with cubic voxelization was proposed by Berrocal et al., when simulating light transport in sprays.[4] Li et al. described an MC code "MOSE" for light propagation in complex geometry media based on a triangle mesh model.[5]

Various efforts have been made to reduce simulation time, such as algorithm modification and the use of different hardware accelerators. Several modifications were proposed to enhance the MC algorithm. Zolek et al. applied approximate calculation of logarithmic and trigonometric functions,[6] Wang et al. proposed a hybrid model of MC method and diffusion theory,[7] and Alerstam et al. employed a "white MC model" for time-resolved photon migration.[8] However, all these acceleration techniques are applied at the expense of the precision or the flexibility of the MC simulation method.

An alternative approach consists in employing potential of different accelerators. Luu et al. developed an MC modeling algorithm based on the MCML code for field-programmable gate array (FPGA)[9] for the case of plane-parallel geometry. The increase in speed was about 28 times relative to a single threading CPU. However, employment of FPGA as a simulation accelerator has one significant drawback: high complexity of the development process. Thus, a reported rough estimate of job complexity measured in personal effort is 12 person-months (1 person-year). Alerstam et al. created MC code with the help of compute unified device architecture (CUDA) technology for a GPU[10] for simulation of photon migration in semi-infinite homogeneous scattering media. Presented GPU implementation is about 1080 times faster than the conventional CPU code. The GPU code "gpu-MOSE" developed by Li et al.[11] is an improved version of "MOSE"[5] based on a triangle mesh model. The increase in speed is about 10 times relative to a single threading processor. As one can see, the efficiency of GPU use for the MC simulation in case of complex geometry is much less compared to a simple case of semi-infinite homogeneous media. Note that for achieving such acceleration, one should significantly reconstruct the code according to GPU optimization rules. A more complex approach, which employs a peer-to-peer network of CUDA GPUs for performing MC simulation, is presented by Doronin and Meglinski.[12] Using both distributed computing and GPU acceleration allows a significant reduction in simulation time.

In this paper, we present a technical approach of porting MC simulation code to Intel many-integrated core (MIC) architecture. The CPU and cluster versions of the MC code considered are previously described in this paper[13] and were employed for

*Address all correspondence to: Mikhail Yu. Kirillin, E-mail: mkirillin@yandex.ru

1083-3668/2015/$25.00 © 2015 SPIE

simulation of optical diffuse spectroscopy signal in noninvasive brain sensing. In brief, this code is based on principles of standard MCML implementation and can be used for modeling of light transport in heterogeneous turbid media with complex geometry of layers. A boundary of a layer is described with a triangulated surface (or surfaces). An intersection search algorithm employs a bounding volume hierarchy (BVH) tree as accelerating structure. In our code we avoid using a classical Russian roulette approach, setting a minimal photon weight threshold instead. Additionally, we input a class of detectors and store information about the photons passed through it. For storing photon trajectories, a data rectangular grid in the three-dimensional (3-D) space is employed.

One of the highest computational-cost applications of MC technique in biomedical optics is simulation of near-infrared spectroscopy (NIRS) brain sensing.[2,3,11,13–15] Usually, simulation of photon transport to distances of several centimeters is required for this problem, resulting in hundreds of scattering events to be simulated for each trajectory. In this respect, early papers (see, for example, Ref. 2) considered elementary photon movement as a movement for a transport length instead of movement for a free path length, which allowed reduction in the number of processed scattering events for a factor of $1/(1-g)$. For head tissues, the g value varies between 0.8 and 0.99 in accordance with different literature,[16] which may result in speed increase up to 100 times when applying this approach. However, this approach may provide incorrect results when calculating radiation parameters close to the point of incidence of probing radiation. Additionally, application of this approach is incorrect when the layer thickness is smaller than the transport length. For example, cerebral spinal fluid (CSF) is characterized by relatively small scattering coefficient, and thus a diffuse regime cannot be reached within it. An accurate account of each scattering event offer the ability to overcome these problems, while modern achievements of computational hardware allow achievement of acceptable calculation times. The latter approach was employed in our simulations to generate photon trajectory maps, which enable an evaluating distribution of probing radiation reaching detector within the human head, thus determining measurement volume.

2 Many-Integrated Core Implementation of Monte Carlo Technique

Intel MIC architecture (or Intel MIC) is a multiprocessor computer architecture that combines many Intel x86 CPU cores onto a single chip. Prototype products were announced and released to developers in 2010. Commercial products named Intel Xeon Phi coprocessor are now available. Like a GPU, Intel Xeon Phi can be used as a hardware accelerator for traditional programs, and it was initially designed for that purpose. The key advantage of this architecture is that any developer can build and run source code on a coprocessor using standard existing programming tools and methods, such as OpenMP and MPI. The same program code written for Intel MIC products can be compiled and executed on a standard Intel Xeon processor. Familiar programming models remove training barriers, allowing the developer to focus on the problems rather than software engineering. This is especially important for research teams in applied fields.

However, only programs with a high degree of parallelism can be efficiently accelerated with the Intel Xeon Phi coprocessor. Code vectorization is an additional advantage. Note that these requirements are also actually for efficient acceleration of code execution with GPU. MC simulation fits these requirements, and so is an appropriate example for porting to an Intel MIC coprocessor.

There are several programming modes employed for code development for coprocessors. The first one, so-called offload mode, allows use of the Intel Xeon Phi as an additional accelerator for CPU. This mode is similar to CUDA employment while programming for GPU. The main code is executed on a processor, and some critical parts of it are sent to the coprocessor and executed there. In this mode, one needs to control data transfers between CPU and Xeon Phi, and therefore the source code should be modified. The second mode allows execution of the program on a coprocessor only, without using a CPU. A special compilation procedure for using this mode is required. The third available mode, called symmetric mode, consists of simultaneous employment of both CPU and Xeon Phi. In this mode, the same code is executed on both devices, while communication between them is performed by means of MPI commands. If the code is already optimized for execution on a cluster, it can be used in this mode without any modifications. In all modes, for getting parallel code within the scope of each device, OpenMP technology can be employed.

Modern Intel MIC coprocessors contain nearly 60 cores with x86 architecture; each core can simultaneously execute four threads (nearly 240 threads in general). Moreover, 64 KB L1 and 512 KB L2 cache memory are located on each core. An accelerator also has several gigabytes (6 or 8) GDDR5 onboard memory, but the latency for this memory is much greater compared to CPU RAM.

There are at least two significant differences between the CPU and the coprocessor. First, an accelerator has many more threads that can be simultaneously executed. Consequently, the program for a coprocessor should be more parallel, and one should employ synchronization much carefully. Second, memory access time for an accelerator is significant; therefore, it is important to efficiently use facilities of hardware caches to reduce memory access time. Another useful feature of Intel Xeon Phi is support of long vector instructions. However, in MC algorithm, there are no significant facilities for vectorization.

3 Model Parameters

In simulations, we employed a human head model consisting of six layers, namely scalp, fat, skull, CSF, gray matter, and white matter. The boundaries of these layers are adopted from real head geometry obtained from magnetic resonance imaging (MRI) data. A total of 2,186,446 triangles were employed to describe the layer boundaries. This approach allowed us, on the one hand, to create model geometry close to that of a real head and, on the other head, to avoid consideration of numerous small areas with varying optical properties that could significantly increase computational time. Layer boundary geometries employed in simulations are shown in Fig. 1. The optical properties of the layers adopted from the literature were stated earlier in this paper[13] and are summarized in Table 1, together with layer thicknesses.

The simulation of the NIRS system consisted of simulation of photon transport from a source to detectors situated on the surface of a human head. Simulation was performed for two wavelengths of NIR range corresponding to different values of absorption coefficients of oxy- and deoxyhemoglobin similar to operation of real NIRS systems.[17] The source–detector

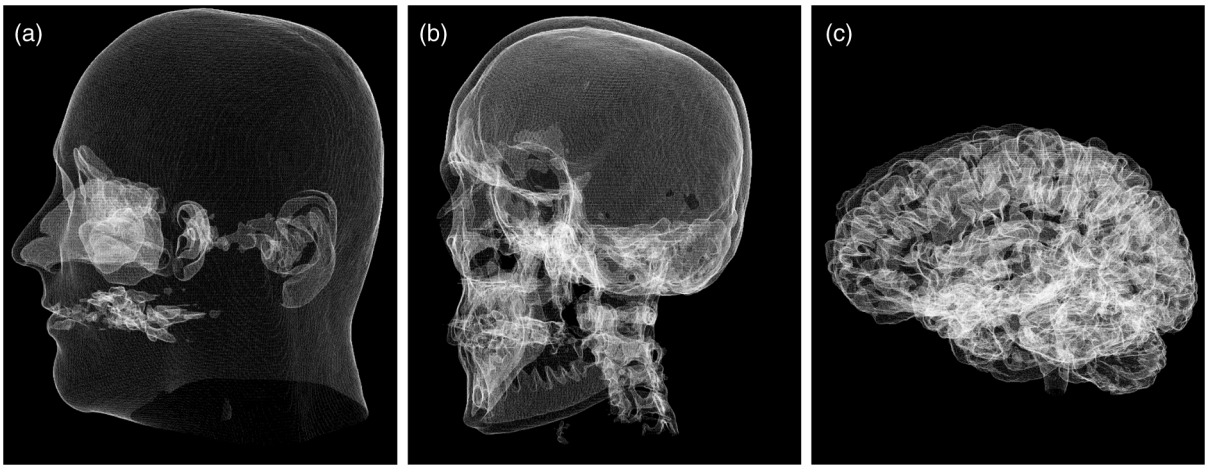

Fig. 1 Triangulated shapes of human head layer boundaries employed in simulations: (a) skin, (b) skull, (c) gray matter.

Table 1 Optical properties of human head layers at $\lambda = 830$ nm.

Head layer	l (mm)	μ_s (mm^{-1})	μ_a (mm^{-1})	n	g
Scalp	2.1	14.3	0.025	1.4	0.86
Fat	3.2	10	0.1	1.4	0.9
Skull bone	6.9	25	0.02	1.55	0.94
Cerebral spinal fluid	2.5	1	0.004	1.33	0.99
Gray matter	5.8	25	0.02	1.4	0.96
White matter	60	26.7	0.02	1.4	0.85

separation was chosen as 20, 30, and 40 mm, which corresponds to typical distances used in experiments.

4 Results and Discussion

4.1 *Efficiency of Many-Integrated Core Implementation of Monte Carlo*

As mentioned above, there are several ways to port the parallel program to a coprocessor without modifications. However, as any hardware device, Intel Xeon Phi has its own specifics. Therefore, to obtain high-performance results, one should take into account the accelerator's architecture features in the course of code adaptation.

We implement several optimized versions of our code in order to reduce computational time (Figs. 2 and 3). For the experiments, we employ a test system with two Intel Xeon X5680 (6 cores, 3.33 GHz, 32 GB RAM,)and an Intel Xeon Phi SE10X coprocessor (60 cores, 240 threads, 1.1 GHz, 8 GB RAM).

Our previously described code[13] (original version) was used as a reference. To compare the efficiency of the same code executed at the accelerator, we chose the coprocessor-only mode.

The MC method is initially highly parallel, so the main aim of the optimizations was to reduce memory latency. Our first optimization was changing the data structure for storing photon trajectories. Initially, the total number of photon trajectory nodes in each grid cell was stored in a 3-D grid array. However, this approach required large memory size for each thread, which was unacceptable for Intel MIC with hundreds of threads; instead, the photon trajectory nodes were stored as an array of grid coordinates for each thread (list version).

In the version of the code for CPU, we tried to avoid any thread synchronization by the use of copies of shared arrays that resulted in employing additional memory. To reduce

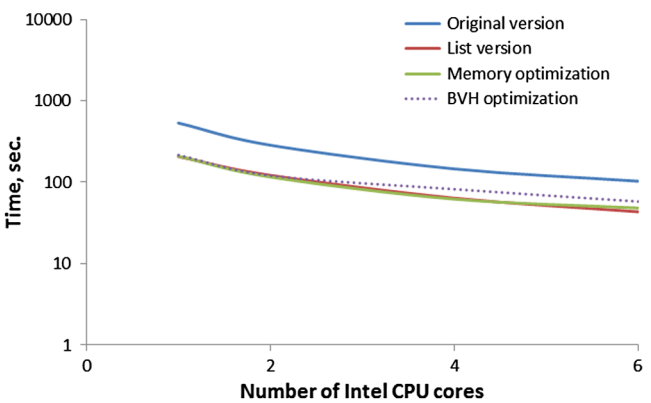

Fig. 2 Execution time of the different versions of Monte Carlo (MC) simulation code (10^5 photons) at CPU.

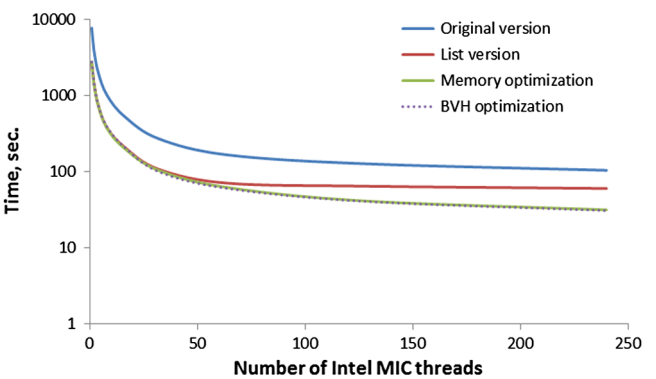

Fig. 3 Execution time of the different versions of MC simulation code (10^5 photons) at Intel Xeon Phi coprocessor.

memory requirements, we returned to synchronization scheme as the second optimization. It should be noted that employing synchronization can cause reduced performance efficiency, especially in the case of hundreds of simultaneous threads. However, in MC simulations, synchronization events happen at different moments in time for different threads due to the stochastic base of the method, and this does not significantly affect performance.

The third optimization was directed to reduce memory access time by intersection search. A BVH tree structure was updated to store vertex data as a single continuous memory region. The aim of this optimization was to more effectively employ hardware cache.

As a result, after implementation of all optimizations, the code performance was 1.8 times faster for the CPU version and 3.4 times faster for MIC version.

If we compare performance of the single-threading CPU version and MIC version (240 threads) in the test simulation for a real human head geometry based on MRI data,[13] the latter is 11 times faster. If we compare results for the multithreading CPU code and the code for Intel Xeon Phi (Fig. 4), the MIC program is nearly two times faster than the program, which is executed on a 6-core CPU. Therefore, if we had an optimized version of code for multicore CPU initially, we would obtain two times acceleration by using Intel Xeon Phi just after code recompilation.

Additionally, we implemented the symmetric version of our code, which can be executed on both CPU and coprocessor at the same time. The version that employs all available computational resources (two Intel Xeon CPUs and Intel Xeon Phi) simultaneously was expected to provide maximal acceleration. Employing a coprocessor with two CPUs simultaneously provides nearly two times faster performance compared to two CPUs only, without significant code modifications. An additional advantage of the symmetric version is that it can be executed on a heterogeneous cluster with Intel coprocessors at its nodes without modifications.

Another important issue is the efficiency of the considered MIC architecture against the widely employed GPU. For performance comparison, we developed the GPU version of our code. As a base, we took the last optimized version and rewrote it employing CUDA technology with an additional optimization—overlapping computations and CPU–GPU memory transfers. For testing purposes, NVidia Tesla M2070 was employed. As seen from Fig. 5, the GPU version is slightly faster than the version for Intel Xeon Phi, but this difference is inconsiderable (nearly 9% for 10^7 photons).

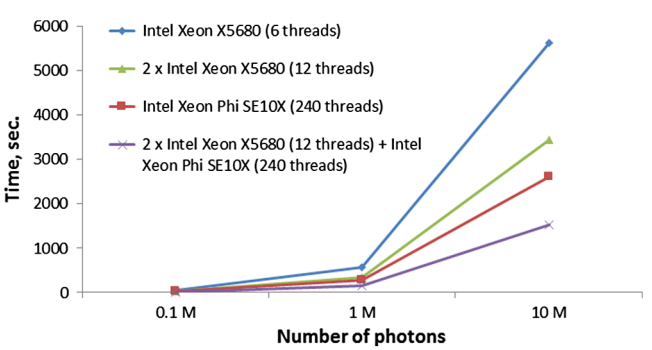

Fig. 4 Execution time of the MC simulation code at different hardware.

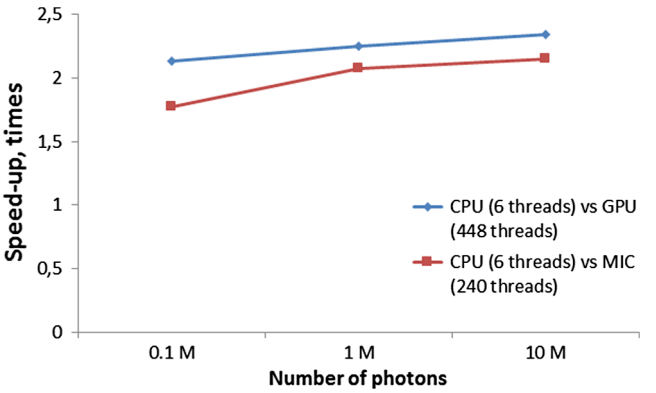

Fig. 5 Speed-up of the MC simulation code for GPU (NVidia Tesla M2070, 448 threads) and MIC (Intel Xeon Phi SE10X, 240 threads) against CPU (Intel Xeon X5680, 6 threads).

To provide a cost-efficiency comparison of GPU and Intel Xeon Phi, we should mention that the launch price of NVidia Tesla M2070 in 2011 was nearly $4000 and for Intel Xeon Phi SE10X in 2012 it was nearly $2500. Currently, both accelerators are no longer manufactured but accelerators with comparable performance from Intel and NVidia show comparable prices (from $2000 to $5000).

Another option for achieving two times acceleration is either employing 2 CPUs with 6 cores or 1 CPU with 12 cores. However, both cases are not cost-efficient, because the price of one Intel Xeon X5680 (3.33 GHz, 6 cores) is near $1600 and the price of a 12-core CPU, such as Intel Xeon E5-2697 v2 (2.7 GHz), is more than $2600.

4.2 *Monte Carlo Simulations of Near-Infrared Spectroscopy Brain Sensing*

Determination of measurement volume is an important problem of NIRS brain sensing. Unfortunately, it is impossible to accurately determine the measurement volume in experiment because it requires knowledge of photon trajectories, which travel from source to detector. The photon average trajectory approach can provide solutions for so-called "banana-shaped" photon trajectories (see, for example Ref. 18); however, they can be obtained only for simple geometries. Finite-difference techniques (for example, widely used NIRFAST code)[19] can be a solution for complex geometry cases; however, they cannot provide the opportunity to trace individual trajectories, providing only intensity distribution within a medium. In this connection, MC technique appears to be the most suitable for aims of NIRS brain sensing.

In this framework, we simulated photon trajectories for selected human head geometry, and probing wavelengths of 830 and 900 nm usually employed in NIRS brains sensing. A typical 3-D photon trajectory map for source–detector separation of 40 mm and wavelength of 900 nm is presented in Fig. 6.

Unfortunately, 3-D presentation of the map is not suitable for accurate analysis. For demonstration of abilities of NIRS brain sensing in selected geometry, we built two-dimensional cross sections of the map in the plane based on radiation incidence vector and source–detector vector for different separations and wavelength (Fig. 7). Cross sections of layers boundaries shown in the same figure allow evaluation of the penetration depth of probing radiation, thus determining the measurement volume of

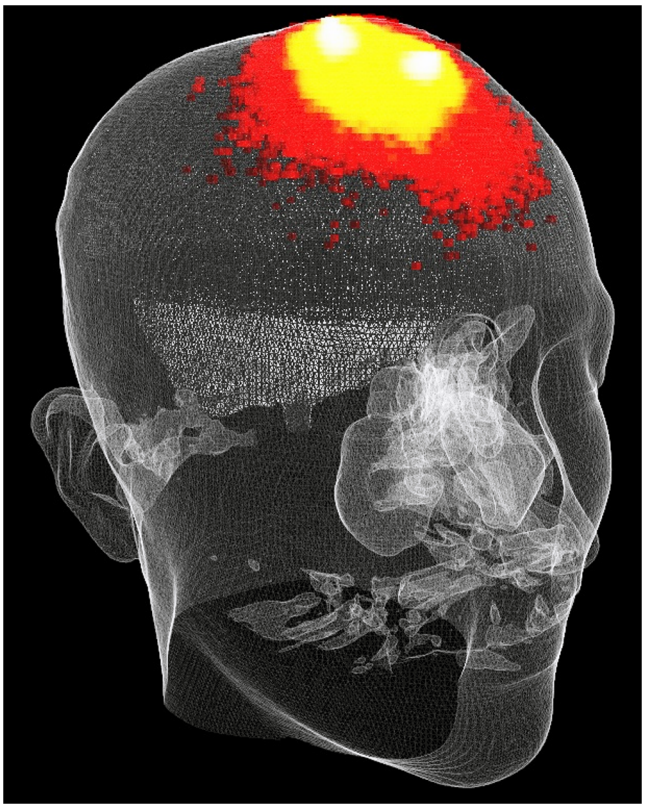

Fig. 6 Three-dimensional (3-D) photon trajectory map for source–detector separation of 40 mm and wavelength of 900 nm.

the NIRS system. This figure shows that probing radiation reaches the gray matter layer, thus proving the ability of the simulated NIRS system to monitor brain activity. On the other hand, one can see that for smaller source–detector separations, the majority of trajectories do not reach gray matter, so increasing this distance leads to in-depth shift of measurement volume.

Here, we demonstrated the abilities of the developed technique in simulation of NIRS brain sensing for complex geometry based on diagnostic MRI data. Accurate quantitative study of the measurement volume is the subject for our future work.

5 Conclusion

In this paper, we discuss the details of hardware acceleration of simulation of photon migration in complex-shape medium by means of modern technology solutions. In particular, we show that the Intel Xeon Phi allows one to increase the efficiency of MC simulation with arbitrary geometry of layers, providing 11 times increase in speed over traditional single-thread CPU code. While this result is comparable with GPU, the key advantage of developing software for Intel MIC architecture in contrast to GPU is the ability to develop an executable version with minimal modifications of the original source code optimized for a multicore CPU. Furthermore, to work with the Intel Xeon Phi one needs to know only "standard" technologies for programming on multicore CPUs (such as OpenMP and MPI) and does not require learning new ones, unlike CUDA for GPU.

We demonstrated the abilities of the developed simulation product for solution of the problem of determination of measurement volume in NIRS brain sensing. The study was performed for MRI-based multilayer geometry that allowed, on the one hand, accounting for anatomical features and, on the other hand, avoiding computational-cost consideration of small

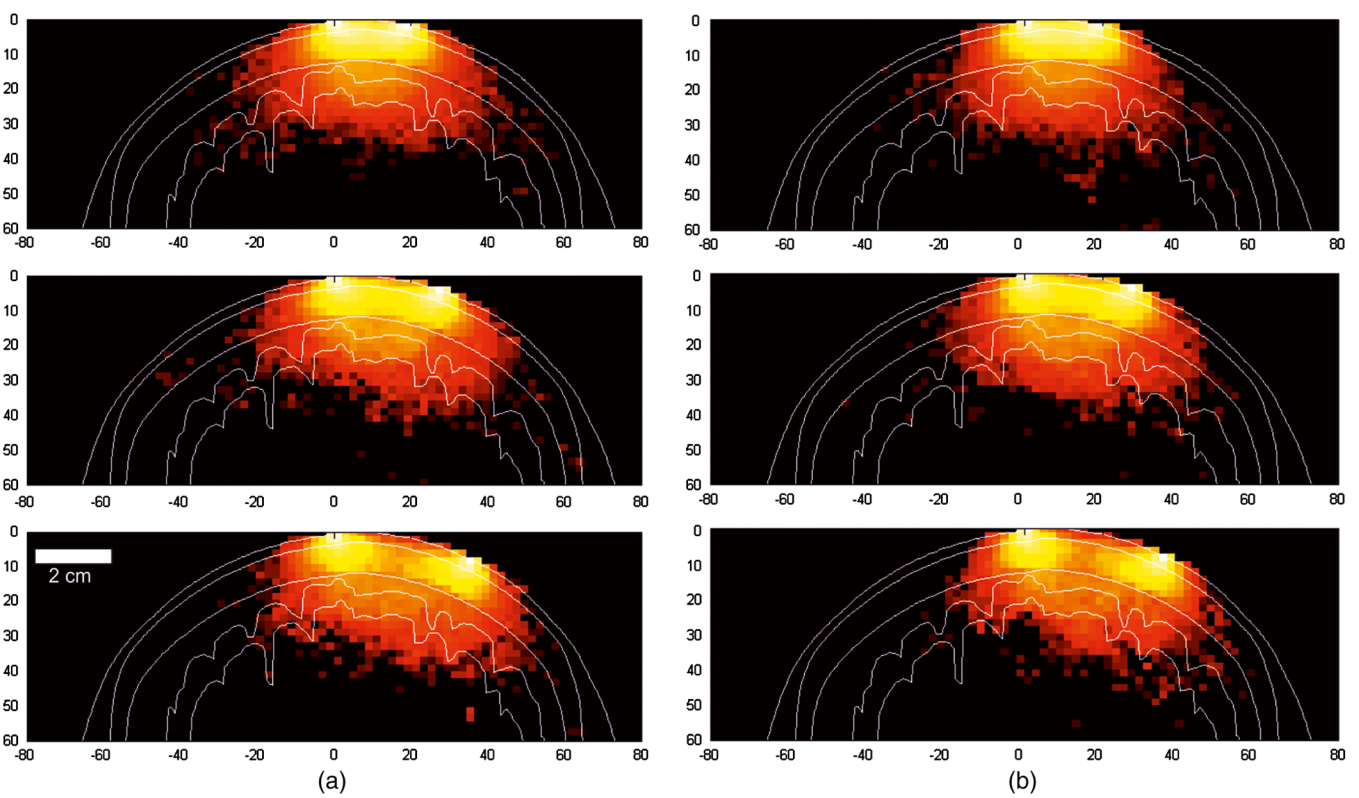

Fig. 7 Two-dimensional (2-D) photon trajectory maps for source–detector separations of 20 (top), 30 (middle), and 40 (bottom) mm for (a) 830 and (b) 915 nm.

optical inhomogeneities. Simulations allowed us to demonstrate in-depth shift of measurement volume with increase in source–detector separation.

Acknowledgments

Authors acknowledge Ekaterina Sergeeva (IAP RAS, Russia), Vesa Korhonen, and Vesa Kiviniemi (Oulu University Hospital, Finland) for useful discussions on the human head model. The work is financially supported by the Russian Foundation for Basic Research (Projects Nos. 14-02-31549 and 15-02-04270).

References

1. L. H. Wang, S. L. Jacques, and L. Q. Zheng, "MCML—Monte Carlo modeling of light transport in multilayered tissues," *Comput. Methods Prog. Biomed.* **47**(2), 131–146 (1995).
2. D. A. Boas et al., "Three dimensional Monte Carlo code for photon migration through complex heterogeneous media including the adult human head," *Opt. Express* **10**(3), 159–170 (2002).
3. C. C. Chuang et al., "Brain structure and spatial sensitivity profile assessing by near-infrared spectroscopy modeling based on 3D MRI data," *J. Biophoton.* **6**(3), 267–274 (2013).
4. E. Berrocal, I. Meglinski, and M. Jermy, "New model for light propagation in highly inhomogeneous polydisperse turbid media with applications in spray diagnostics," *Opt. Express* **13**(23), 9181–9195 (2005).
5. H. Li et al., "A mouse optical simulation environment (MOSE) to investigate bioluminescent phenomena in the living mouse with the Monte Carlo method," *Acad. Radiol.* **11**(9), 1029–1038 (2004).
6. N. S. Zolek, A. Liebert, and R. Maniewski, "Optimization of the Monte Carlo code for modeling of photon migration in tissue," *Comp. Methods Prog. Biomed.* **84**(1), 50–57 (2006).
7. L. V. Wang and S. L. Jacques, "Hybrid model of Monte Carlo simulation and diffusion theory for light reflectance by turbid media," *J. Opt. Soc. Am. A* **10**(8), 1746–1752 (1993).
8. E. Alerstam, S. Andersson-Engels, and T. Svensson, "White Monte Carlo for time-resolved photon migration," *J. Biomed. Opt.* **13**(4), 041304 (2008).
9. J. Luu et al., "FPGA-based Monte Carlo computation of light absorption for photodynamic cancer therapy," *17th IEEE Symp. on Field-programmable Custom Computing Machines*, pp. 157–164, IEEE Press, New York (2009).
10. E. Alerstam, T. Svensson, and S. Andersson-Engels, "Parallel computing with graphics processing units for high-speed Monte Carlo simulation of photon migration," *J. Biomed. Opt.* **13**(6), 060504 (2008).
11. N. Ren et al., "GPU-based Monte Carlo simulation for light propagation in complex heterogeneous tissues," *Opt. Express* **18**(7), 6811–6823 (2010).
12. A. Doronin and I. Meglinski, "Peer-to-peer Monte Carlo simulation of photon migration in topical applications of biomedical optics," *J. Biomed. Opt.* **17**(9), 090504 (2012).
13. A. V. Gorshkov and M. Yu. Kirillin, "Monte Carlo simulation of brain sensing by optical diffuse spectroscopy," *J. Comp. Sci.* **3**(6), 498–503 (2012).
14. K. Kurihara et al., "The influence of frontal sinus in brain activation measurements by near-infrared spectroscopy analyzed by realistic head models," *Biomed. Opt. Express* **3**(9), 2121–2130 (2012).
15. C. C. Chuang et al., "Patient-oriented simulation based on Monte Carlo algorithm by using MRI data," *Biomed. Eng. Online* **11**, 21 (2012).
16. V. V. Tuchin, *Tissue Optics: Light Scattering Methods and Instruments for Medical Diagnosis*, SPIE Press, Bellingham, Washington (2007).
17. V. O. Korhonen et al., "Light propagation in near-infrared spectroscopy of the human brain," *IEEE J. Sel. Top. Quantum Electr.* **20**(2), 7100310 (2014).
18. V. V. Lyubimov et al., "Application of the photon average trajectories method to real-time reconstruction of tissue inhomogeneities in diffuse optical tomography of strongly scattering media," *Phys. Med. Biol.* **47**(12), 2109 (2002).
19. H. Dehghani et al., "Near infrared optical tomography using NIRFAST: Algorithm for numerical model and image reconstruction," *Commun. Numer. Methods Eng.* **25**(6), 711–732 (2009).

Anton V. Gorshkov is an assistant lecturer at the Nizhny Novgorod State University. He graduated with honors from the Department of Computational Mathematics and Cybernetics of Nizhny Novgorod State University in 2011. In 2014, he received his PhD from Nizhny Novgorod Technical State University. His scientific interests include high performance computing in science, general-purpose computations on GPU and heterogeneous cluster systems, Monte Carlo techniques, and numerical simulations of light transport in scattering media.

Mikhail Yu. Kirillin is a senior researcher of the Laboratory of Biophotonics at the Institute of Applied Physics of RAS. He received his PhD degree from Moscow State University in 2003 and Dr.Sc. (Tech.) degree from University of Oulu in 2008. His scientific interests include optics of biotissues and other scattering media, optical tomography modalities, as well as theoretical description and numerical simulations (in particular, Monte Carlo technique) of light transport in scattering media.