FPGA-Assisted Strategy toward Efficient Reconstruction (FAStER) in Diffuse Optical Tomography

Yuanyuan Jiang, Sovanlal Mukherjee, James E. Stine, Charles F. Bunting, Daqing Piao

School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK, 74078, USA
Corresponding author: daqing.piao@okstate.edu

Abstract: The finite-element computation of photon fluence and adjoint photon fluence necessary to image reconstruction in steady-state DOT has been implemented on field-programmable-gate-array (FPGA). Preliminary results encourage further exploration toward efficient DOT image reconstruction using FPGA.

©2010 Optical Society of America
OCIS codes: (170.6960) Tomography; (170.3010) Image reconstruction techniques.

1. INTRODUCTION

Diffuse optical tomography (DOT) utilizes near-infrared (NIR) light to interrogate biological tissues at a depth up to several centimeters to recover the distribution of internal optical properties based on boundary measurements. The image reconstruction of DOT is most often rendered by diffusion-model-based forward computation and iterative non-linear optimization [1], which is inevitably computationally expensive. Consequently, using application-specific computer architecture to accelerate the DOT computation becomes attractive. A number of computer architectures useful for accelerating the data acquisition and processing in optical imaging have been demonstrated recently. Examples include using field-programmable gate array (FPGA) technology to accelerate raw data processing in optical imaging [2, 3], using FPGAs or graphic processing units (GPUs) to accelerate Monte Carlo computation of photon migration [4-6], using FPGAs to solve partial differential equations (PDEs) governing heat transfer [7] or wave propagation [8], and using GPU to perform finite-element-method (FEM) computation [9].

In this work the FEM solution to photon diffusion in biological tissue is implemented using an FPGA. The FPGA executes conjugate gradient (CG) solver of 12 linear equations formulated in an FEM framework, which are associated with 6 sources and 6 detectors, for computing the photon fluence rate and the adjoint fluence rate. Preliminary results demonstrate that a lower-end FPGA outperforms a higher-end PC in CG-based solution of the 12 linear equations, thereby encouraging further exploration toward efficient DOT image reconstruction using FPGA.

2. METHOD AND MATERIALS

2.1 Development of an open-code FEM-based forward solver for steady-state diffuse optical tomography

Implementing the DOT image reconstruction routine in FPGA requires an algorithm architecture that is transparent to FPGA. An open-code forward FEM solver for steady-state DOT reconstruction is developed. The solver is based on the steady-state photon diffusion equation [1] \( \nabla \cdot (\kappa \nabla \Phi) - \mu_a(r) \Phi = -q(r) \) (where \( \mu_a \) is the absorption coefficient, \( \kappa \) is the diffusion coefficient, \( \Phi \) is the photon fluence rate at position \( r \) ), and the boundary condition [1] of \( \Phi(r_{1b}) + 2\kappa \hat{n} \cdot \nabla \Phi(r_{1b}) = 0 \) (where \( r_{1b} \) corresponds to the point on the boundary, \( \hat{n} \) is a unit vector pointing outward (from the tissue to probe) and normal to the tissue-probe interface, and \( A \) is the boundary mismatch factor determined by the relative refractive indices of the tissue domain and the probe (air) domain). These equations formulate into the FEM framework \( [K(\kappa) + C(\mu_a) + B/(2A)] \Phi = Q_0 \) where the \( K \), \( C \) and \( Q \) are volume integrals of each element with regard to \( \kappa, \mu_a \) and \( q \), and \( B \) is the surface integral of the boundary element. The FEM forward solver results in a set of linear equations containing sparse matrices. The inverse problem performs a non-linear optimization of the objective function of \( b = \| \Phi_{\text{measured}} - \Phi_{\text{estimated}} \| \) by updating the pixel or voxel-wise values of \( \kappa \) and \( \mu_a \). The inverse solver requires finding \( \partial b/\partial \kappa \) and \( \partial b/\partial \mu_a \), which are integrated into the forward computation process by using the adjoint method of deriving the Green’s function associated with an impulse source at the detector position, as shown in Fig. 1(a) (b). Therefore, the number of sources, \( s \), and the number of detectors, \( s \) generate \( 2s \) sets of linear equations for solving by the CG method.

Our open-code FEM-solver is developed in MATLAB (Mathworks, Inc. Natick, MA) platform. A comparison of our solver with the NIFAST package [10] is given in Fig. 1(c), where the target has an absorption coefficient of 0.02 mm\(^{-1}\) and a reduced scattering coefficient of 1.2 mm\(^{-1}\), in a background of 0.002 mm\(^{-1}\) absorption and 0.8 mm\(^{-1}\) reduced scattering. The performance of our solver is comparable to that of NIFAST, at the same 1% noise-level.

2.2 Implementation of the conjugate gradient solution of the linear equations in FPGA
BSuD18.pdf

The FEM implementation of the linear equations for DOT forward computation is composed of four modules. The calculation module, which includes two floating point adders and two floating point multipliers, is capable of two simultaneous floating point vector operations with an approximately throughput of four floating point operations per clock cycle. The memory modules utilize on-chip block memory. And the sparse FEM matrix is stored by compressed row storage (CRS) [11]. The DOT sources are considered Gaussian and the adjoint sources are impulse, which also lead to sparse structure.

We have temporarily used the RS232 protocol for the data transfer between the FPGA and PC (shown in Fig. 2). We have also temporarily implemented only the forward computation in FPGA, and performed the inverse solver algorithm on a PC. The control module, which is a finite state machine (FSM), controls the data flow in forward computation, as shown in Fig. 3. The data flow starts from loading FEM matrix from PC and then clears all the intermediate memories. The source profile is restored as originally vector sequence according to the address offset and address index. At the same time, the initial calculation of residual norm, or \( \alpha_0 \), which is out of the CG loop, is calculated. Then it enters the CG iteration and three states, updates \( \alpha_0 \), \( \beta_0 \), and \( \Phi \), runs iteratively until \( \alpha_0 \) is below a threshold. Then the FSM jumps out of the CG iteration, sends the computed fluence to PC, clears the state, and loads the next source or adjoint source profiles. The complete fluence data set are transferred to PC for running the inverse solver that leads to a new set of FEM matrices for being transferred to FPGA for the next iteration.

3. RESULTS AND DISCUSSIONS
The FEM solver is implemented on PC only and on FPGA-PC unit as specified above for comparison of the speed. On PC, the FEM-solver is executed on an Intel® quad-core 2.33 GHz processor. The linear equation is solved by using “bicgstab” function in Matlab with the FEM matrix being specified as sparse, which means the nonzeros in the matrix are stored together in main memory and the low spacial-locality caused by the zeros in sparse matrix is solved. A 2-D mesh with 1,705 nodes and 11,593 non-zeros in the corresponding FEM matrix, for an imaging dimension of 54mm×30 mm with 6 sources and 6 detectors on the boundary, is generated for DOT image reconstruction. The background is set at absorption coefficient of 0.01mm\(^{-1}\) and reduced scattering of 1mm\(^{-1}\), with a target of 0.025mm\(^{-1}\) absorption and 1.75mm\(^{-1}\) reduced scattering. A Xilinx VirtexII Pro FPGA (XC2VP30 Package f896 Speed Grade -7) is used, which contains 30,816 logic cells, 136 18×18 multipliers and 2,448 Kbits of Block RAM [12] with maximum clock frequency of 150MHz (the actual clock frequency being used is 100MHz). The execution speeds of PC-only and FPGA-PC unit, both in IEEE 754R double precision, are given in Fig. 4 (red and blue bars). The bars #1 and #2 correspond to CG algorithms with 50 and 100 iterations, respectively, for a 1705×1705 matrix. The bars #3 correspond to solving one above-mentioned FEM-associated linear equation. The bars #4 correspond to one complete forward computation of solving 12 linear equations. For each group of the bars, there are two FPGA-runtime settings. The “one-instance” corresponds to using 32 out of 136 on-board DSP modules that is necessary to computing the linear equation one by one, and the “two-instances” corresponds to using 64 out
of 136 on-board DSP modules to compute two linear equations simultaneously. Compared with PC running time, there is a 1.8 folds and 3.6 folds of speed improvements with the “1-instance” and “2-instances”, respectively. The 136 DSP modules ideally allow simultaneous computation of 4 linear equations that could lead to 7.2 folds of speed improvement at the given clock frequency, but the insufficient on-chip memory has limited implementing more than 64 DSP modules for this study. The images reconstructed by PC only and by FPGA-PC unit are compared in Fig. 5.

It is noted that the FPGA used in this study is a low-end sample unit with limited on-board resources. Using high-end FPGAs with more on-board resources could further speed up the above computations. Higher-end FPGAs such as Virtex 5 and Virtex 6 families has more DSP resources which accommodate 25×18 instead of 18×18 multipliers, thereby could further improve the speed. Table 1 lists the performance improvement that could be expected, with the present study listed as the first one, by using the existing higher-end FPGAs.

The use of RS232 protocol in this initial study has resulted in overall slower DOT iteration due to the initial and final data transmissions between the FPGA and PC. There are a number of approaches to improve the overall performance, including implementing a real-time data transmission protocol such as USB or Ethernet, developing a stand-alone on-board operating system, and performing both the forward and inverse solver algorithms on the FPGA.

4. CONCLUSION AND FUTURE DIRECTION

In summary, FPGA implementation of FEM based forward computation for steady-state DOT is demonstrated. For a system employing 6 sources and 6 detectors with a mesh having 1,705 nodes, forward computations involving 12 linear equations solved by CG method are performed by FPGA. The preliminary results, even though hindered by the slow RS232 data transfer protocol and limited resources on the FPGA, encourage implementing complete forward and inverse iteration on FPGA for efficient reconstruction.

Acknowledgement: This work has been supported by Oklahoma Center for the Advancement of Science and Technology (OCAST) (HR06-171), and DOD Prostate Cancer Research Program (#W81XWH-07-1-0247).

REFERENCES