Bloch Simulations on a Desktop Computer Exploiting a Parallel Framework

[1]P. Latta, [2]M. L.H. Gruwel, [2]J. Hovdebo, [2]B. Tomanek

[1]CEITEC – Central European Institute of Technology, Masaryk University, Brno, Czech Republic
[2] NRC-CNRC, 435 Ellice Avenue, Winnipeg, Manitoba, Canada R3B 1Y6
Email: lattape@gmail.com

Abstract. *A common approach in magnetic resonance imaging is to use Bloch simulations to explore different aspects of the spin behaviour. Typical for these calculations is the high computational demand, which put restrictions on the number of spins involved in such simulations. In this paper we present a performance comparison for two parallelization techniques; multi-core versus graphics processing unit. Both techniques result in an increase in computational performance. The results from 2D and 3D experiments are presented with comparison of the computational accelerations. The performance tests indicate that the graphics processing unit introduces an effective and affordable increase in computational efficiency.*

Keywords: *MRI Simulation, Bloch Equation, Graphics Processing Unit* (*GPU*)

1. Introduction

The easy and convenient way to investigate various aspects of spin behavior in magnetic resonance imaging (MRI) is to use Bloch simulations. A common problem for such simulations is the finite, however large volumes of isochromats. This makes simulation very demanding on the computational power. In many occasions this is solved by the usage of computer clusters [1]. Despite of its computational effectiveness, such clusters might be expensive, maintenance demanding and are not always accessible. However, recent progress and development in multi-core (MC) CPU and a graphics processing unit *(*GPU) use provides an opportunity to use massive parallelism on desktop or even on laptop computers [2]. Here we present and compare the effectiveness of two different parallelization approaches.

2. Subject and Methods

The home made Bloch simulator [3] has been extended with two parallel mechanisms:

1. Intel® Threading Building Blocks – allows to take advantage of multi-core computation.
2. CUDA – allows harnessing the computational power of the graphics processing unit GPU.

In order to evaluate the MC and GPU computational performance, three different simulations tasks have been performed:

a) reconstruction from projections using the two dimensional (128x128 voxels) Shepp-Logan phantom shown in Fig 1(A -B),
b) inhomogeneity simulation with a cylindrical phantom (128x128 voxels) containing an axially placed air-filled tube using 3 isochromat per voxel, (see [3]) shown in Fig 1(C-D),
c) gradient echo sequence with slice selection using 3D (128x128x64) phantom shown in Fig 1(E-F).

All simulations were performed on a DELL Precision T7500 desktop computer , with 2 CPUs Intel Xeon® X5570, 2.93 GHz, 4 cores each, and capable to run 16 hyper threads in total, with 6GB memory. The Dell T7500 was equipped with a GeForce GTX285 (240 Processor Cores) graphics card which was used for GPU computing. All simulations were running on the Linux/Kubuntu 11.10 (Oneiric Ocelot) operating system and the sequence preparation and data reconstruction were performed using Octave mathematical package.

Table 1 Comparison of effectiveness of MC and GPU performance against the single core simulation experiment.

		Total simulation time / simulation speedup relative to single core	
Phantom size	single core	multi core with 16 hyper threads	GPU
128x128 reconstruction from projections (1 isochromat per voxel)	23.1 sec	4.6 sec / 5.0x	2.63 sec / 8.7x
128x128 inhomogeneity simulation (3 isochromats per voxel)	68 sec	9.5 sec / 7.1x	5.8 sec / 11.7x
128x128x64 grad. echo with slice selection (1 isochromat per voxel)	5937 sec	807 sec / 7.3x	293 sec / 20.2x

3. Results

The simulation experiments were tested and the net execution time recorded i.e. time for data reconstruction was excluded from time measurement. The results are summarized in Table 1. The parallelization of simulation brought a significant increase of performance for all types of scenarios and even for relatively small 2D input data matrices. For 2D simulations the relative acceleration was in the range of 5 – 11.7 times, however, there was no significant difference between MC and GPU parallelism. This changed dramatically when huge volumes of data were involved as in the case of 3D simulations, where a 20.2 times GPU acceleration was measured against a single core computing and a 2.77 acceleration of GPU versus MC. This can be explained by fact that for such computational and time demanding simulations the data transfer from CPU to GPU memory does not represent substantial time consumption and the full power of many GPU processor cores brings a full advantage [2].

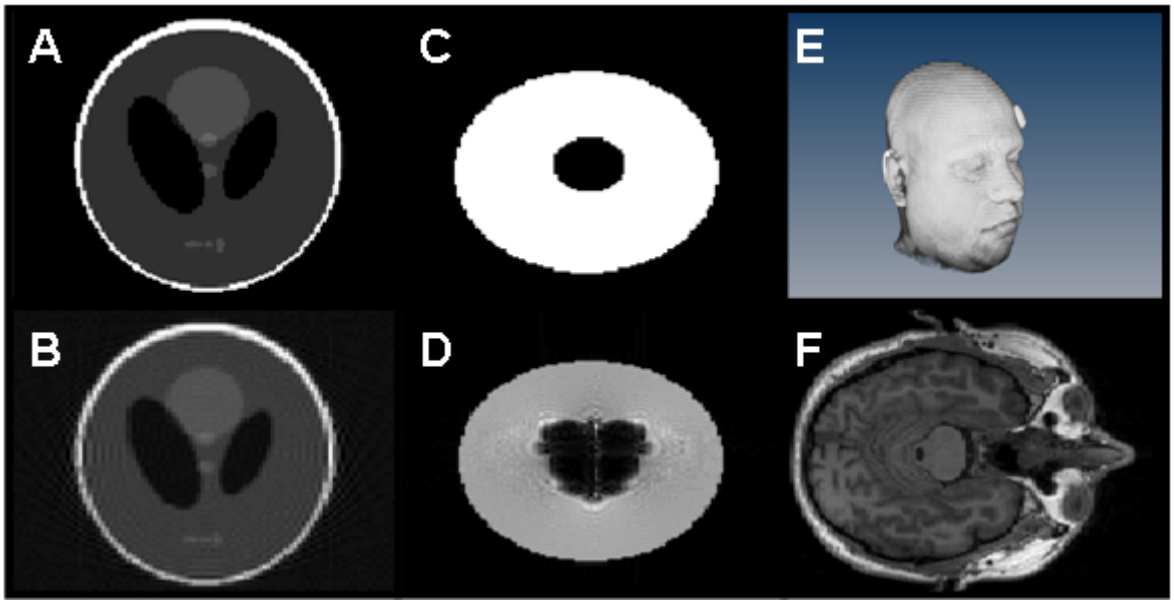

Fig. 1. Input phantoms and output images from GPU simulation experiment: A–B reconstruction from projections, C-D inhomogeneity simulation, E-F gradient echo with slice selection.

5. Conclusions

Using a MC or GPU implementation of the Bloch simulation could significantly speed up the simulation process compared to single core calculation. This is even more obvious for simulations involving 3D data, where the GPU simulation version exhibits an acceleration of more than 20 times. In all simulation experiments the GPU over-performed the MC, most significantly in 3D simulations. A significant advantage of the GPU computational platform is its scalability, which allows the construction of a powerful multiple board configuration and which may help further improvement of the speed performance in relatively reasonable price.

Acknowledgment: We would like to thank Dr. V. Jellúš for his helpful suggestions and help with preparation of this manuscript.

References

[1] Stöcker T, Vahedipour K, Pflugfelder D, Shah NJ. High-performance computing MRI simulations. *Magnetic Resonance in Medicine,* (64): 186–193, 2010.

[2] Lechner SM, Butnaru D, Bungartz HJ, Chen D, Vogel MW. Bloch solver simulation realization on a graphics-processing unit (GPU), p. 2695, *ISMRM 17th Scientific Meeting & Exhibition*, Honolulu, 2009.

[3] Latta P, Gruwel MLH, Vladimír Jellús V,Tomanek B. Bloch simulations with intra-voxel spin dephasing. *Journal of Magnetic Resonance,* (203): 44–51, 2010.