Accelerating Diffusion Tensor Estimation Using General-Purpose Graphics Processing Unit

L-C. Chang[1], and M. A. Gorbachev[1]
[1]Department of Electrical Engineering and Computer Science, The Catholic University of America, Washington, DC, United States

Introduction: A general-purpose graphics processing unit (GPU) is a dedicated graphics rendering device with the capability to perform trillion of instructions per second. It offers a powerful processing platform for both graphics and non-graphics applications. Many computer vision and medical imaging algorithms such as advanced MRI reconstruction [1] and Diffusion Tensor Imaging (DTI) connectivity mapping [2] have been paralleled using the GPU architecture. The introduction of the CUDA (Compute Unified Device Architecture) and Tesla technologies [3] from NVIDIA provides an easy way to take advantage of the high performance GPUs for parallel computing on a personal computer or a workstation. DTI is a non-invasive magnetic resonance technique that produces in vivo images of biological tissues with local microstructural characteristics such as water diffusion [4, 5]. Diffusion tensor maps are computed by fitting the signal intensities of diffusion weighted images (DWIs) as a function of their corresponding b-matrices through a multivariate least-squares regression model [4]. This diffusion tensor computation is typically performed on a voxel-by-voxel basis through the entire 3D volume which makes it an ideal application for GPU parallelization.

The purpose of this work is to apply GPU hardware in the diffusion tensor map estimation by accelerating the weighted multivariate linear least-squares regression. Unlike solving large matrix problems in linear regression by GPU [6], our aim is to perform thousands of independent multivariate linear regressions in parallel on the GPU. We propose a hybrid approach to accelerate the diffusion tensor estimation: compute the weighted multivariate linear regression on the GPU, and perform the logarithm transform and other tensor derived quantities on the CPU. This hybrid approach takes performance advantage of the GPU to speedup vector and matrix operations in a bulk computation.

Methods: In a diffusion tensor model, a signal intensity vector $\mathbf{y} = \{\ln(S_1), \ldots , \ln(S_N)\}^T$, where S_i represents the i^{th} DWI magnitude signal intensity in a DTI acquisition. There are several parameters in a DTI model: $\boldsymbol{\alpha} = \{D_{xx}, D_{yy}, D_{zz}, D_{xy}, D_{xz}, D_{yz}, \ln(A_0)\}^T$, where D_{ij} are elements of the diffusion tensor, and A_0 is the echo intensity with no applied gradients [4]. A log linear model can be written as a first order equation $\boldsymbol{y} = \boldsymbol{B\alpha} + \boldsymbol{e}$, where the j^{th} row of $\boldsymbol{B}$ contains b-matrix entries of the j^{th} DWI acquisition $-\{b_{xxj}, b_{yyj}, b_{zzj}, 2b_{xyj}, 2b_{xzj}, 2b_{yzj}, -1\}$, and $\boldsymbol{e}$ is the error vector. The weighted least squares solution using the linear model is given by $\boldsymbol{\alpha} = \left(\boldsymbol{B}^T\tilde{\Sigma}_{\mathbf{e}}^{-1}\boldsymbol{B}\right)^{-1}\left(\boldsymbol{B}^T\tilde{\Sigma}_{\mathbf{e}}^{-1}\right)\mathbf{y}$, where $\tilde{\Sigma}_{\mathbf{e}}$ is a diagonal matrix that can be obtained from the measured signal intensities for a given noise variance, and matrix $\boldsymbol{B}$ can be obtained by a experimental design [7]. In our hybrid approach for tensor computation, the logarithmic transform of the signal and other tensor derived quantities were computed on the CPU, and the solution to find the tensor $\boldsymbol{\alpha}$ is computed on the GPU. To test the performance of this implementation, simulated 3D brain data is created by using a 30 gradient-direction scheme with b=1000 s/mm^2 plus two non-DW images [8]. The computation time for diffusion tensor estimation on this dataset is compared between a dedicated GPU (Tesla C1060) and dual-CPU (Intel quad core 3.40 GHz Xeon processor) on a computer workstation. The computer programming was implemented using Interactive Data Language (IDL) on the CPU and C language calling CUDA library on the GPU.

Results and Discussion: Figure 1 shows a comparison of GPU vs. CPU computation time for performing weighted multivariate linear least- squares regression on the simulated 3D brain dataset. All computation was performed on a slice-by-slice basis. Each slice has different number of voxels that need to be computed which range from 0 to 5000. Both GPU and CPU obtained the same tensor results. However, GPU has a computation time that is 3 to 6 folds faster than the CPU. This variation is due to the number of voxels within the slice is different.

While the current GPU computation is performed on a slice-by-slice basis which is convenient and effective, it may not be optimal to take the full advantage of our GPU hardware. The optimal choice depends on the maximum number of processing blocks multiple by the maximum number of threads per block in the GPU. For example, the maximum number of voxels that can be transferred to the Tesla C1060 is 32767×32 per function call. Further speedup can be expected by transferring more voxels to the GPU and by better data allocation on the GPU memory.

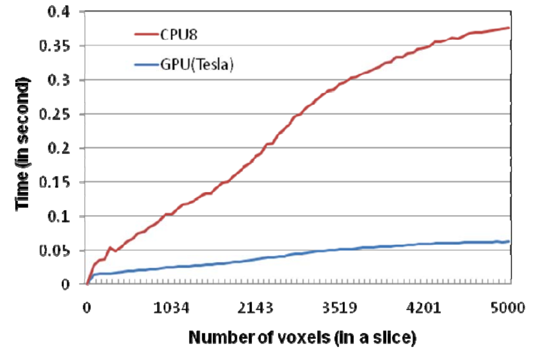

Figure 1. Run time comparison for performing weighted multivariate least-squares regression on a simulated brain DTI data set.

Conclusions: The performance gain of the GPU-based multivariate linear regression is demonstrated in Diffusion Tensor MRI application. Our results show that the Tesla GPU outperformed the dual quad core Xeon processors by a factor of 3 to 6 for diffusion tensor estimation. Further speedup maybe expected by optimizing data transfer to the GPU and by better GPU memory allocation. The proposed GPU framework can significantly accelerate the DTI simulation and can be readily applied to quantitative assessment of the DTI using bootstrap analysis.

References: [1] Stone S, et al. *J. Parallel Distrib. Comput.* 68:1307–1318, 2008. [2] McGraw T, et al. *IEEE TVCG.* 13:1504-12, 2007 [3] Halfhill T, *Microproc. Report* 22(1):1-8. 2008. [4] Basser P, et al. *J Magn Reson B* 103(3): 247-54, 1994. [5] Basser P, et al. *J Magn Reson B* 110: 209-219, 1996. [6] CULATool. http://www.culatools.com, 2009. [7] Mattiello J, et al. *Magn Reson Med* 37(2): 292-300, 1997. [8] Chang L, et al., *Magn Reson Med* 57:141–149, 2007.