GPU Accelerated CSD-based Probabilistic Tractography

Parnesh Raniga[1], Kai Wong[1], Donald Tournier[2], Alan Connelly[2,3], and Olivier Salvado[1]

[1]CSIRO Preventative Health National Research Flagship ICTC, The Australian e-Health Research Centre, Brisbane, Queensland, Australia, [2]Brain Research Institute, Florey Neuroscience Institutes (Austin), Melbourne, Victoria, Australia, [3]Department of Medicine, University of Melbourne, Melbourne, Victoria, Australia

Introduction: Tractography methods are increasingly used to detect white matter alteration and study brain connectivity. High angular resolution diffusion imaging (HARDI) acquisition along with modelling techniques such as constrained spherical deconvolution (CSD) [1] can overcome the issue of crossing fibres [2] and generate tractography information that can be analysed at very high resolutions, using track density imaging for example [3]. However this requires generating vast amounts of tracks or streamlines using probabilistic [4] and/or bootstrap tractography [5] approaches to estimate a solution within certain confidence bounds, resulting in long processing times. Processing times can be reduced by conducting tractography in parallel on multicore systems as each computed track or streamline is assumed to be independent of every other. Another avenue for reducing processing times is to use graphical processing units (GPUs). In recent years, the processing power of GPUs has greatly surpassed central processing units (CPUs) and GPUs can greatly speedup algorithms that are parallel in nature and map easily to their architectural paradigm. In this work, we present a GPU implementation of an established CSD-based multi-fibre tractography algorithm [4] and compare it to the CPU-based implementation which is available as part of the MRtrix software package (http://www.brain.org.au/software/mrtrix/).

Methods: The tracking algorithm proceeds by initiating tracks at given seed points. This is done by randomly sampling the amplitude of the fibre orientation distribution functions (FOD) at the seed point in multiple random directions. If any of sampled amplitudes exceeds a given threshold, a track is initiated. Tracking then continues via rejection sampling of the FODs along the direction of the track [4]. Points along the track are stored as the tracking continues. A track ends once the rejection sampling of the FOD fails to exceed a given threshold.

Current methods using multicore CPUs compute each track in its entirety for each computation thread, which will result in execution dead time on GPUs as the execution is limited by the processing time taken by the longest track. Instead we used a novel implementation that allocates each GPU thread a given number of tracks until the total number of points across all the tracks in each GPU thread of computation reaches a given limit. Another novelty of our implementation is the use of texture units on the GPU for interpolating the spherical harmonics coefficients of the FOD. The GPU algorithm was implemented in C++ in the CUDA framework [6].

The GPU and CPU versions of the algorithms were compared by performing whole-brain probabilistic tractography using HARDI data (60 directions and 8 b=0 images, b-value = 3000 s/mm^2) from 10 elderly subjects. For each subject, 10 tractography runs were computed with whole brain seeding with one seed per mm^3. This is different from the official MRtrix distribution which uses random sampling to pick seed points. The running time for each of the ten executions was compared. The parameters for both CPU and GPU version were identical and were the defaults used in the MRtrix package except for minimum track length, which was set to 0. A single thread/core was used for the execution of the CPU version of the algorithm. Track density imaging (TDI) maps were generated for each tractography run. Average TDI images were computed for 2 groups of 5 out of the 10 TDI images. The mean squared error (MSE) was computed between average TDI maps, comparing the CPU algorithm to itself and to the GPU algorithm (CPUvCPU and CPUvsGPU). Processing and analysis were conducted on machine with 8 CPU cores (2 × E5540 Intel Xeon CPUs @ 2.53 GHz) with 12GB of RAM and an Nvidia GTX460 GPU (2Gb of ram, 332 GPU cores).

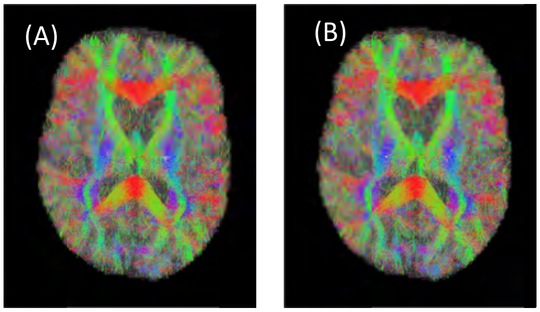

Figure 1. Tractography maps from the same subject using the CPU version (A) and GPU version (B) of the algorithm.

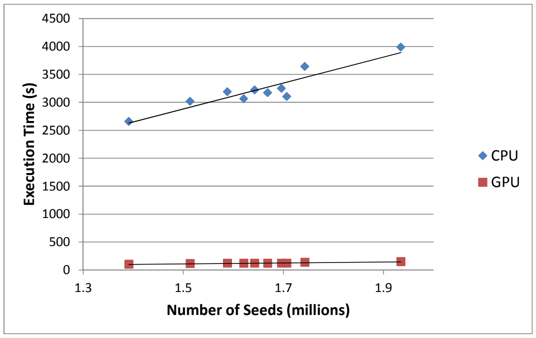

Figure 2. Graph of the average running times of the CPU and GPU version of the algorithm for each subject with seeding of 1 seed/mm^3. Variation in the number of seeds is due to differences in head size of subjects.

Results: Examples of tractography maps of the CPU and GPU versions are shown in Figure 1. No qualitative differences in the tractography maps between the GPU and CPU versions were observed. A graph of the running times of the GPU and CPU versions is presented in Figure 2, showing that the GPU version was 25 times faster compared to the CPU version. That allows us to compute whole brain tractography in 2 mins instead of 50 mins on a single core/thread. There were no significant differences (p-value > 0.05, Wilcoxon signed-rank test) in the MSE of average TDI maps between the GPU and CPU (CPUvGPU) (3.45 ± 0.43) versions of the algorithm when compared to the MSE computed between different executions of the CPU algorithm (CPUvsCPU) (3.43 ± 0.44).

Discussion and Conclusion: Advanced probabilistic algorithms can provide a wealth of important information over the whole brain, but at the cost of increased processing times. We have implemented a GPU tractography algorithm that makes use of textures to interpolate FOD and split the tracks spatially to be processed by independent GPU threads. Our GPU implementation of a probabilistic tractography algorithm was able to achieve speed-up factors of over 25 times on a standard GPU board (Nvidia GTX 460) while producing the same results as the CPU version. Newer generation of GPU boards should achieve even greater speedups as more computation power becomes available.

References:

[1] J.-D. Tournier et al., *NeuroImage*, vol. 35, no. 4, pp. 1459-1472, May 2007. [2] T. E. J. Behrens et al., *NeuroImage*, vol. 34, no. 1, pp. 144-155, Jan. 2007. [3] F. Calamante et al., *NeuroImage*, vol. 53, no. 4, pp. 1233-1243, Dec. 2010. [4] J.-D. Tournier et al., *ISMRM*, no. 13, p. 01343, 2005. [5] B. Jeurissen et al., *Human Brain Mapping*, vol. 32, no. 3, pp. 461-479, Mar. 2011. [6] J. Nickolls et al., "Scalable Parallel Programming with CUDA," *Queue*, vol. 6, no. 2, pp. 40–53, Mar. 2008.