GPU-Enabled Individualized Acceleration Apportionment for SENSE and CAIPIRINHA

Eric A. Borisch[1], Paul T. Weavers[1], and Stephen J. Riederer[1]
[1]*Mayo Clinic, Rochester, MN, United States*

PURPOSE

2D SENSE [1] and CAIPIRINHA [2] are techniques that can each be used to provide acceleration in 3DFT acquisitions. For a given overall acceleration R, it is generally not clear how to optimally choose the individual acceleration factors R_Y and R_Z for SENSE or the specific R_Y, R_Z, Δ sampling kernel for CAIPIRINHA. Recently methods have been described for improving performance through individualized selection of acceleration parameters for each patient/coil combination [3]. Implementing such a technique generally requires searching through a parameter space of potential acceleration combinations or kernels, and calculating the expected exam "quality" for each. A likely input to this "quality" is a metric derived from the g-factor penalty values calculated over the entire 3D volume. In order for this acceleration optimization to be practical, it is desirable that the optimum configuration be determined within seconds after acquisition of the coil sensitivity data and prior to the accelerated scan, akin to the time required in prescanning for selection of transmitter and receiver gains. To test the dozens (CAIPIRINHA) or hundreds (2D SENSE) of potential acceleration configurations within a short, 10 sec evaluation phase, it is desirable that the evaluation time for each be under about 50 msec. However, calculation of these g-factor penalties for SENSE or CAIPIRINHA 3D exams is a computationally intensive task. The purpose of this work is to describe the issues involved in such a computation and how we have attained the 50 msec target by migration of the computation from a cluster of multi-processor computers to a Graphics Processing Unit (GPU).

METHODS

Computational Requirements: The g-factor penalty in SENSE or CAIPIRINHA, describing the noise penalty within a voxel due to aliasing, is defined [1] as:

$$g_p = \sqrt{[(S^H \Psi^{-1} S)^{-1}]_{p,p} (S^H \Psi^{-1} S)_{p,p}} \quad (1)$$

where: n_p is the number of aliased voxels; p is an index over n_p; S is the complex-valued coil sensitivity matrix with a size of $n_{coils} \times n_p$; and Ψ is the complex-valued coil-to-coil noise correlation matrix with a size of $n_{coils} \times n_{coils}$. To calculate the g-factor values for any (R_Y, R_z) acceleration to be considered, this equation must be evaluated for each set of n_p aliasing voxels, leading to multiple thousands of floating point operations per set. Scaled by the acquired (aliased) resolution, this can lead to multiple billions of floating point operations that must be performed per each considered R-pair (R_Y, R_z) acceleration for SENSE, or each (R_Y, R_z, Δ) kernel for CAIPIRINHA.

Computing Techniques: G-factors for each set of aliasing voxels can be calculated as a "by-product" during the creation of the SENSE unfolding matrix, defined by $U = (S^H \Psi^{-1} S)^{-1} S^H \Psi^{-1}$. The result from the inversion of the parenthesized product is also the key matrix of interest for calculating g-factors. This implementation was used on an eight-node cluster of dual-processor (16 CPUs total) computers to initially test searches over the R-pair space. The computation, however, was much too slow for our desired sub-10s search time. Two traits of the g-factor calculation align well with current GPU: (i) each set of aliasing voxels can be treated independently, and is therefore directly parallelizable; and (ii) the coil sensitivity volume (source of the S matrices) is static and can be transferred to the GPU once and then used as input (with less than 64 bytes/R-pair of additional data transfer) for all R-pairs considered.

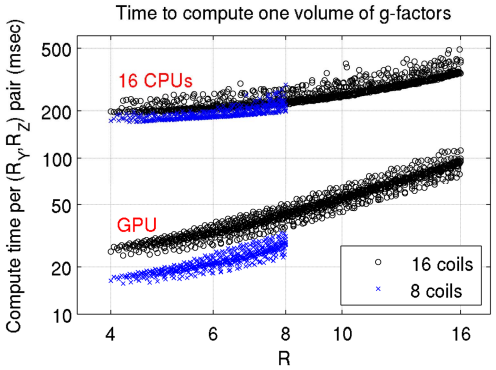

Figure 1: 16-CPU cluster (upper two sets) and GPU compute timings for $420 \times 336 \times 132$ 16-coil and 8-coil datasets over a range of acceleration factors. Compute times are sub-50ms for all $R \leq 8$ cases. For a given R, multiple R_Y, R_Z combinations can be used, with subtle time differences for each.

GPU Implementation: Linear algebra applications utilizing GPUs have, in general [4], focused on working with large (dimensions well above 100 per side) matrices, while this work uses millions of small (sides ≤ 32) matrices. Each work-item within the GPU operates on a set of aliasing voxels, with memory arranged to optimize performance for this approach. OpenCL [5] is used to implement the calculation, with all intermediate matrices stored in private memory. As the low-level matrix calculation and inversion code is being implemented directly, the calculation is "tuned" to incorporate a priori knowledge, eliminating operations whose results are known by construction, and omitting calculations whose results do not impact the final output. For example, in this case only the diagonal elements of the key matrix inversion of Eq. 1 are of interest.

Reference Application: The methods were evaluated using an application demanding good computational performance: large-FOV, high spatial resolution, high coil count, high acceleration imaging of the calves as used for contrast-enhanced MRA [6]. The matrix size was $420 \times 336 \times 132$. Computation times were noted for accelerations factors over the range R=4 to 16 for multiple (R_Y, R_z) combinations using coil sensitivity data from both 8- and 16- element receiver arrays.

RESULTS

Computing one set of g-factors on the reference 16-element calf dataset at $(R_Y, R_z) = (6, 2)$ acceleration (acceleration levels achieved previously with good results [7]) required >1 sec on the previously described 16-CPU cluster. The initial GPU (nVidia C2050) implementation required more than 300 msec for the same task. After tuning the calculation, as described above, this was reduced to 65 msec. Migrating these optimizations back to the 16-CPU cluster yielded a 290 msec computation. Computational times for are shown in Figure 1 for the original 16-CPU cluster and GPU over a range of accelerations with 8- and 16-coil configurations. The rapid pace of complexity growth with increasing acceleration and coil counts is apparent.

CONCLUSIONS

The ability to calculate hundreds of g-factor maps in less than ten seconds, a performance level that will enable integrating individualized acceleration prescription intra-exam, has been successfully demonstrated. With appropriate tuning and parsing of the computation, a GPU can provide anywhere from a 4.5-20x speed improvement depending on the selected comparison – indeed comparison against fewer processors than used here would result in an even higher factor. More importantly, when compared against contemporary hardware, a ~15x improvement in hardware cost/performance is observed. A prime limitation for this work is memory capacity on the GPU. In the future, scaling using multiple GPUs is expected to enable further reduction in the computation time, as this is a separable calculation as described, to consistently reach a sub-50ms goal with larger coil counts and accelerations.

REFERENCES

[1] Pruessmann, MRM, 42:952 (1999) [2] Breuer, MRM, 55:549 (2006) [3] Weavers, ISMRM 2012, #2224 [4] Lahabar, IPDPS 2009
[5] http://www.khronos.org/opencl/ [6] Haider, Radiology, 253:831 (2009) [7] Weavers, ISMRM 2012, #2648