SAND2013-7440C

Irregular Large-Scale Computed Tomography on Multiple Graphics Processors Improves Energy-Efficiency Metrics for Industrial Applications

Edward S. Jimenez, Eric L. Goodman, Laurel J. Orr, and Kyle R. Thompson

Sandia National Laboratories
PO Box 5800
Albuquerque,NM 87185
{*esjimen,elgoodm,ljorr,krthomp*}*@sandia.gov*

***Abstract*—This paper will investigate energy-efficiency for various real-world industrial computed-tomography reconstruction algorithms, both CPU- and GPU-based implementations. We show that the energy required for a given reconstruction is based on performance and problem size. There are many ways to describe performance and energy efficiency, thus we will investigate multiple metrics including performance-per-watt, energy-delay product, and energy consumption. We found that irregular GPU-based approaches [1] realized tremendous savings in energy consumption when compared to CPU implementations while also significantly improving the performance-per-watt and energy-delay product metrics. Additional energy savings and other metric improvement was realized on the GPU-based reconstructions by improving storage I/O by implementing a parallel MIMD-like modularization of the compute and I/O tasks.**

I. Introduction

Industrial Computed Tomography (CT) is an indirect imaging Technique similar to medical applications except frequently on a larger scale with respect to dose, energy, and resolution [2], [3]. The system consists of an x-ray source and detector with a rotary stage located somewhere in between the two (dependent on the desired magnification and system limitations) along with a post-processing machine, usually a high-end laptop, workstation, or supercomputer. The x-ray source and detector are typically stationary while the measurement object is rotated on the rotary stage. Multiple X-ray images (projections) are acquired about the axis of rotation, usually measured with just one revolution [4]. For this paper, large-scale CT can mean that numerous projections are acquires (greater than 1000), the projections have a large pixel count (greater than 10 megapixels), or any combination thereof.

CT reconstructions are very computationally and bandwidth intensive and thus requires significant computing resources such as a high-end workstation or cluster. For many industrial CT applications, the reconstruction algorithm used has complexity of $O(n^4)$ [5], with some special CPU-based algorithms that reduce the complexity to $O(n^3 \log(n))$ [6], [7]

Due to the enormous input datasets (usually between 1 gigabyte and 20 gigabytes) and the computational complexity, reconstruction of industrial datasets can easily consume a significant amount of energy. Therefore, consideration should be taken in choosing an appropriate computing platform.

This work will evaluate multiple implementations of the standard Feldkamp-David-Kress (FDK) reconstruction algorithm [8] that is routinely used for non-destructive testing and industrial scale applications. The evaluation will include multiple energy-efficiency metrics, including energy consumption, performance-per-watt, and the energy-delay product to obtain a broad understanding of the energy-efficiency characteristics of this real-world application.

II. Energy Efficiency

A very simple way of looking at energy efficiency is to measure the total energy used by each system. The less energy used by one system in comparison to another, the better. However, this misses the very important dimension of performance. Two systems may expend the same amount of energy for a given computation, but the total time required may be very different. Thus, other metrics are needed in order to include the notions of power and performance.

One such common measure that incorporates the speed of computation is performance per watt, often MIPS/W or million instructions per second per watt. This incorporates the power of the system into the energy efficiency metric. However, this measure is more appropriate for

laptops or mobile devices, where reducing energy consumption is vital to save the battery. For many integrated circuit designs, power consumption is proportional to the square of the voltage, and reducing voltage by half reduces the frequency of the circuit by much less because of this quadratic term [9]. Thus one could easily sacrifice speed to get a better value for performance per watt. When speed of computation is a more vital factor, a different metric is warranted.

Gonzalez and Horowitz [10] propose a metric with a greater emphasis on performance, namely *energy* $\times$ *delay*, in other words, the total energy used times the amount of time for the computation, with the lower the number the better. Brooks et al. [11] propose even squaring or cubing the delay, to give an even greater emphasis on performance.

We will use all three of these metrics to compare energy efficiency of the various implementations of the FDK reconstruction algorithm we examine in this paper.

III. Irregular Computed Tomography

CT reconstruction algorithms frequently consist of some variation of a back projection. The back projection operation in a CPU-based environment is typically coalesced with respect to the x-ray image or sinogram data. Neighboring voxels in the reconstruction volume will usually access image data from a small neighborhood of pixels of a given x-ray image. In a multi-threaded CPU-based implementation where the number of threads is on the order of 16 and each thread is updating its individually assigned voxel simultaneously, the memory access pattern can still be made relatively small (i.e. small enough so that CPU L1 and L2 cache hit-rates are still acceptable) by assigning neighboring voxels to CPU threads on the same multiprocessor. Regardless of the position of this neighborhood in the volume, the magnified projected neighborhood on the x-ray image is still reasonably small for almost any volume of interest and thus performance is not significantly degraded due to the memory access pattern.

In a GPU-based environment, hundreds to thousands of threads could potentially update hundreds to thousands of voxels simultaneously. The magnified projection data footprint could be potentially enormous (at least larger than the on-chip cache structure can handle). Additionally, these thousands of threads do not run in lock-step; the consequence is neighboring threads potentially accessing entirely different x-ray images, or worse, updating entirely different image planes from a disjoint subset of images. In the CUDA environment, lock-step execution only occurs at the warp level (32 threads), with no control on warp execution ordering [12]. Therefore, the memory access pattern could easily become irregular over the large projection data neighborhood and throughout the entire reconstruction, thus severely hindering performance. This problem is exacerbated by the introduction of large-scale data; this include more projections, large magnifications, large images, large voxel counts, or any combination thereof. Although GPUs have worked very well on reconstruction algorithms on smaller scale datasets (medical datasets for example); for large-scale data, the algorithm would still require hours to days to reconstruct a 64 billion voxel volume.

IV. Implementation

The implementation of this work revolves around the kernel design presented in the work by Jimenez et. al. [1] in order to achieve a majority of the energy metric improvement. The kernel design focuses on the general architecture of a graphics processor and exploits various features of the hardware such as:

1) *Fast Device Memory:* Data access can be faster and in parallel fetches on Graphics processors, this allows for faster evaluation of the bilinear interpolation step required of most back projection algorithms.
2) *Massive Multi-threading:* This is the most well known feature and appeal of GPGPU applications. For reconstruction, each thread is assigned a set of voxels to update, arranged such that no more than one voxel per image plane is assigned to a given thread.
3) *Texture Memory:* Fast read-only memory that has its own dedicated on-chip cache. Reading x-ray subimages through texture memory frees up L1-cache for voxel information and thus increasing computational (and ultimately energy) efficiency.
4) *Hardware Interpolation:* An additional benefit of utilizing texture memory is the exploitation of hardware-based interpolation. Although GPU-based interpolation is usually done in a lower precision, it has been shown to not affect numerical stability noticeably [13].
5) *Constant Memory:* To further reduce L1-cache pressure, all geometry parameters defining the arrangement of the CT system are stored in this user customizable on-chip cache.

The work by Jimenez et. al. also showed that in order to maximize voxel processing throughput by a GPU, one must mostly dedicate the device memory to voxel storage and upload small sets of x-ray projection data.

Implementing the irregular kernel is only half the the task. Energy efficient optimizations made on the kernel will be moot if the host cannot provide the device with data fast enough in order to minimize device idling. To address the host side implementation, the method developed by Orr and Jimenez [14]. This approach attempts

to minimize the GPU downtime by implementing an MIMD-like environment on the host.

Many traditional reconstruction algorithms, both CPU and GPU-based, execute tasks serially in a SIMD approach where a single thread on a CPU is tasked with storage I/O and compute/GPU kernel launch tasks. The consequence is that when a CPU thread is performing I/O tasks, the GPU is idling and not contributing towards the completion of any tasks while still consuming energy (albeit in a potentially lower power state).

The MIMD-like approach modularizes the read, launch and write tasks. Prior to reconstruction, threads are assigned a duty; either read/launch or read/write. Once assignments are made, the algorithm dynamically determines the amount of system memory available and calculates the number of image planes in the volume to reconstruct simultaneously; allowing for volatile memory storage of the relevant x-ray subimages necessary for the given subvolume and the subvolume itself. Next, all CPU threads perform the reading and pre-processing of the relevant x-ray data. Once reading has completed, a subset of threads (equal to the number of GPUs on the host) each determine its proportion of image planes to process at once, dependent on the GPU's hardware specifications and launches the required kernels while feeding necessary input data to the GPU. Once the reconstruction of the set of image planes is completed, it downloads the image planes to host memory and fetches the next set of image planes to reconstruct and repeats the process until all image planes of the subvolume are processed. The complement of host threads each iteratively check if any image planes are ready to be written. When the thread encounters an image plane that is ready to be written, it fetches the image plane, performs any required post-processing and writes the image plane to storage media. This process is repeated until all image planes of the subvolume are written to storage media. The entire process is repeated until the entire volume is reconstructed. The benefit of the approach is that the storage bottleneck is ameliorated by allowing writing tasks to occur during kernel computation while simultaneously reducing GPU downtime. The drawbacks to this approach is the additional load required on the CPUs while the write threads are iteratively checking for image planes to write and the additional memory required to temporarily store the image planes.

This work will present two implementations of irregular computed tomography; The first is a serialized approach based on the work of Jimenez et. al. and Jimenez and Orr [1], [15], the second is the modularized approach of Orr and Jimenez [14] which expands on the serialized approach by allowing overlapping compute and write tasks. Both implementations were written in C++ and Nvidia's CUDA programming environment (Ver. 5.0). For multiple GPU control, as well as the CPU thread assignment in the modularized approach, OpenMP 2.0 was utilized to implement CPU parallel tasking.

V. Evaluation

Three metrics will be measured:

1) *Energy Consumption:* This will be measured in kilowatt-hours (kWh).
2) *Performance-per-Watt:* Presented as average voxels reconstructed (and stored) per second per watt. Other more well-known performance-per-watt metrics, such as MFLOPs per watt, were not used as reconstruction is limited by the computation, the irregular memory access pattern, and the storage media I/O.
3) *Energy-Delay Product:* This metric is measured to ensure that the algorithm is not trading off energy savings for a slower reconstruction. As a delay in reconstruction is more detrimental and efficiency, we use a square weighting of delay as suggested by Laros III et. al. [16].

Metrics will be measured for four implementations of CT reconstruction. The first is a CPU-based multithreaded approach that uses both MPI and OpenMP to implement parallel processing. The CPU-based implementation is currently used in industrial radiography applications and will be used for comparison. Additionally, to serve as a GPU-based baseline, a naïve GPU-based approach is developed which consists of a brute force GPU-porting of CPU code which reconstructs one image plane per GPU iteratively and does not exploit the irregular nature of GPU-based reconstruction [1], [15]. The other two implementations are the serialized and modular approach described in the previous section.

The experiments were performed on a high-end Supermicro workstation that consists of dual octo-core Intel Xeon E-2687W processors clocked at 3.1 GHz with hyperthreading, 512 GB of system memory, 8 Nvidia Tesla M2090 GPUs ("Fermi"-class) in two Next I/O S2090 units connected via 4 PCI-E 2.0 x16 host interface cards, the storage media made up of 8 x 3GB SATA 6 Gb/s drives in a RAID 0 array controlled by an Intel Controller with 1 GB of DDR3 cache.

Energy metrics were obtained using several P3 International P4460 Kill-A-Watt EZ Electricity Usage Monitors directly connected to the workstation and each S2090 device. For the CPU-based implementation measurements, all GPUs were disconnected from the system before measurements are started to ensure GPU energy consumption was eliminated from the measurements.

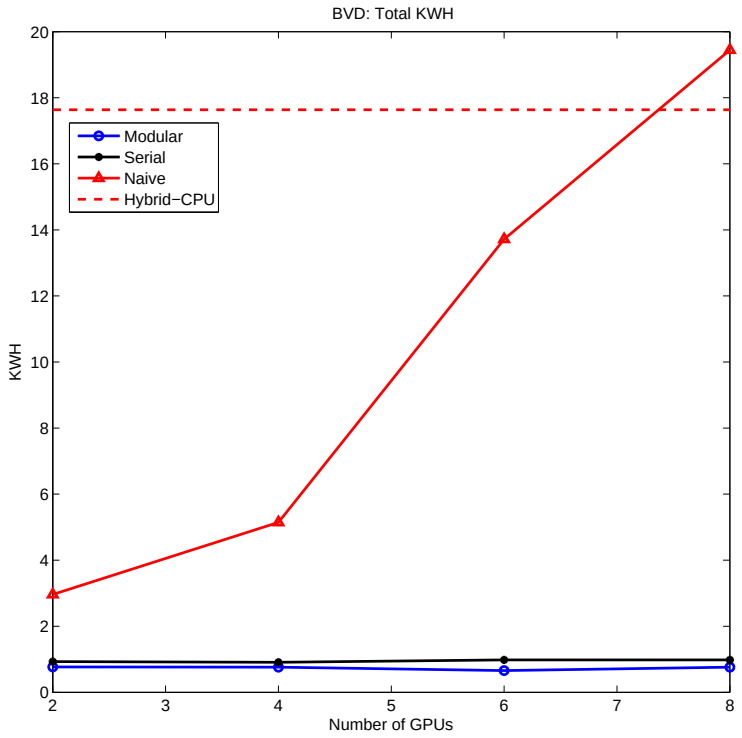

Fig. 1. Energy Consumption in kWh for the 64 gigavoxel reconstruction.

VI. Results

A. Energy Consumption

Figure 1 shows energy consumption for all implementations when reconstructing the 64 gigavoxel dataset with respect to the number of GPUs. It is observed that the GPU-port actually consumes more energy than the CPU-based method when executed using 8 GPUs while both irregular approaches require approximately 1 kWh. For the GPU-port method, the GPUs are never fully utilized and as a result, a majority of the time is spent idling. Figure 2 shows energy consumption for the teravoxel dataset. For both irregular approaches, energy consumption improves with respect to GPUs as the reconstruction is completed in less time; it is noted that the improvement seems to level off towards 8 GPUs and is most likely due to PCI-E bus bandwidth limitations. Figures 1 and 1 show that overlapping compute and storage tasks does indeed improve the energy consumption.

B. Performance Per Watt

Figures 3 and 4 show performance per watt with respect to the number of GPUs on the 64 gigavoxel and teravoxel datasets respectively. It is observed that while both irregular approaches significantly outperform the naïve and CPU-based approaches, the metrics do not improve with respect to GPUs, and in the serialized approach performance actually degrades. The leveled/degraded performance may be due to the fact that while 64 gigavoxels is indeed considered a large-scale dataset, the dataset may not properly push the limitations of the GPUs as is observed in the measurable improvement with respect to GPUs in the teravoxel dataset.

C. Energy-Delay Product

Figures 5 and 6 show the energy-delay product with respect to GPUs on the 64 gigavoxel and teravoxel dataset respectively. These figures validate that the irregular approaches do not trade off efficiency for delay. In fact, for the 64 gigavoxel dataset, the energy-delay product increases by over an order of magnitude for the non-irregular GPU-based approach. Figure 5 shows that while the energy-delay product for the modular approach slightly increases for 8 GPUs, it is still an improvement over all other approaches.

For the teravoxel dataset, the reconstruction pushes all approaches to their limits and thus we see a continued improvement in the irregular approaches with respect to GPU count. In both methods, we see that both irregular approaches achieve an improved energy-delay product

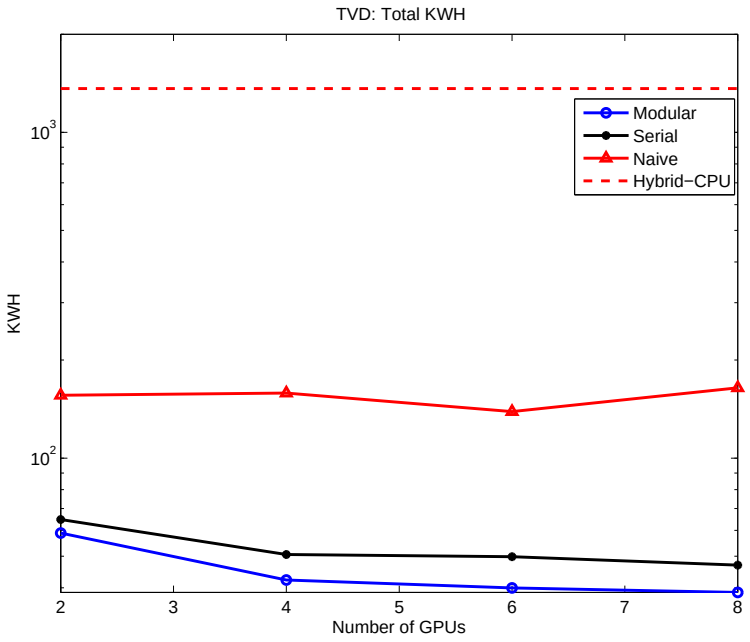

Fig. 2. Energy Consumption in kWh for the teravoxel reconstruction. Note: vertical axis is presented in a log-scale

by 3 orders of magnitude compared to the CPU-based approach.

VII. Conclusion

It has been well documented that computed tomography reconstruction algorithms can greatly benefit from the utilization of GPGPU technology. This has been observed by the remarkable increase in computational performance and reduction in time required to execute the reconstruction task. We have shown that GPU-based approaches also benefit from energy efficiency performance; not just in overall energy consumption, but other energy efficiency metrics as well. As the computing community approaches energy limitations, intelligent algorithm design will be crucial for the exploitation of optimal performance from the hardware. The community needs to explore other approaches outside of reconstruction and investigate whether some algorithms can be further improved by implementing energy efficient methods.

VIII. Acknowledgements

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

References

[1] E. S. Jimenez, L. J. Orr, and K. R. Thompson, "An Irregular Approach to Large-Scale Computed Tomography on Multiple Graphics Processors Improves Voxel Processing Throughput," in *Workshop on Irregular Applications: Architectures and Algorithms*, ser. The International Conference for High Performance Computing, Networking, Storage and Analysis, Nov. 2012.

[2] S. Izumi, S. Kamata, K. Satoh, and H. Miyai, "High energy x-ray computed tomography for industrial applications," *Nuclear Science, IEEE Transactions on*, vol. 40, no. 2, pp. 158 –161, apr 1993.

[3] H. H. Barrett and K. J. Myers, *Foundations of Image Science*. Wiley-Interscience, 2004.

[4] "Enhancement and proof of accuracy of industrial computed tomography (ct) measurements," *CIRP Annals - Manufacturing Technology*, vol. 56, no. 1, pp. 495 – 498, 2007.

[5] F. Xu and K. Mueller, "Ultra-fast 3d filtered backprojection on commodity graphics hardware," in *Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on*, april 2004, pp. 571 – 574 Vol. 1.

[6] S. Xiao, Y. Bresler, and J. Munson, D.C., "Fast feldkamp algorithm for cone-beam computer tomography," in *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, vol. 2, sept. 2003, pp. II – 819–22 vol.3.

[7] C. Axelsson and P. Danielsson, "Three-dimensional reconstruction from cone-beam data in o(n 3 logn) time," *Physics in Medicine and Biology*, vol. 39, no. 3, p. 477, 1994. [Online]. Available: http://stacks.iop.org/0031-9155/39/i=3/a=013

[8] L. Feldkamp, L. Davis, and J. Kress, "Practical cone-beam algorithm," *Journal of the Optical Society of America A*, vol. 1, no. 6, pp. 612–619, 1984.

[9] T. Mudge, "Power: a first-class architectural design constraint," *Computer*, vol. 34, no. 4, pp. 52–58, 2001.

[10] R. Gonzalez and M. Horowitz, "Energy dissipation in general purpose microprocessors," *Solid-State Circuits, IEEE Journal of*, vol. 31, no. 9, pp. 1277–1284, 1996.

[11] D. Brooks, P. Bose, S. Schuster, H. Jacobson, P. Kudva, A. Buyuktosunoglu, J.-D. Wellman, V. Zyuban, M. Gupta, and

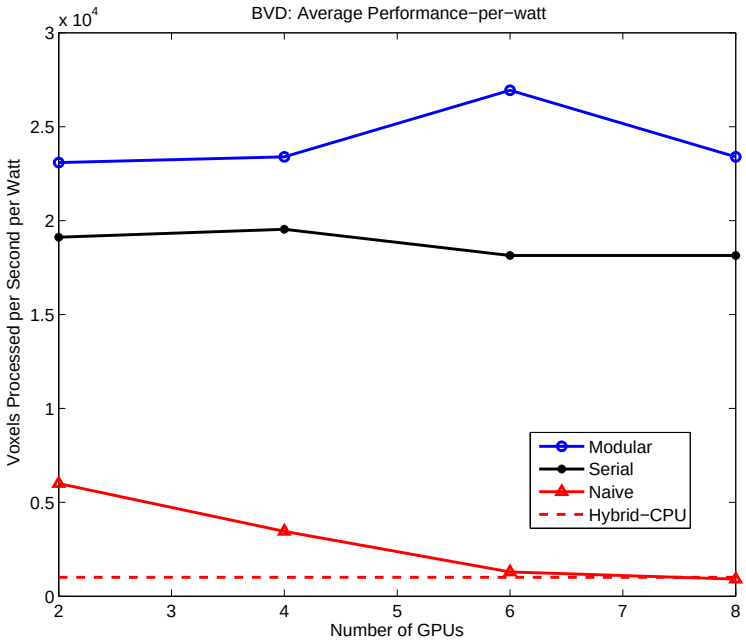

Fig. 3. Performance per watt for the 64 gigavoxel reconstruction

P. Cook, "Power-aware microarchitecture: design and modeling challenges for next-generation microprocessors," *Micro, IEEE*, vol. 20, no. 6, pp. 26–44, 2000.

[12] N. Corporation, *Nvidia CUDA C Best Practices Guide v5.0*. http://www.nvidia.com, 2012.

[13] W. mei W. Hwu, Ed., *GPU Computing Gems - Emerald Edition*. Morgan Kaufmann, 2011.

[14] L. J. Orr and E. S. Jimenez, "Preparing for the 100-megapixel Detector: Reconstructing a Multi-Terabyte Computed-Tomography Dataset," in *Workshop on Penetrating Radiation Systems and Applications XIV*, ser. SPIE Optical Engineering + Applications, Aug. 2013.

[15] E. S. Jimenez and L. J. Orr, "Rethinking the Union of Computed Tomography Reconstruction and GPGPU Computing," in *Workshop on Penetrating Radiation Systems and Applications XIV*, ser. SPIE Optical Engineering + Applications, Aug. 2013.

[16] J. H. Laros III, K. Pedretti, S. Kelly, W. Shu, K. Ferreira, J. Vandyke, and C. Vaughan, in *Energy-Efficient High Performance Computing*, ser. SpringerBriefs in Computer Science, 2013.

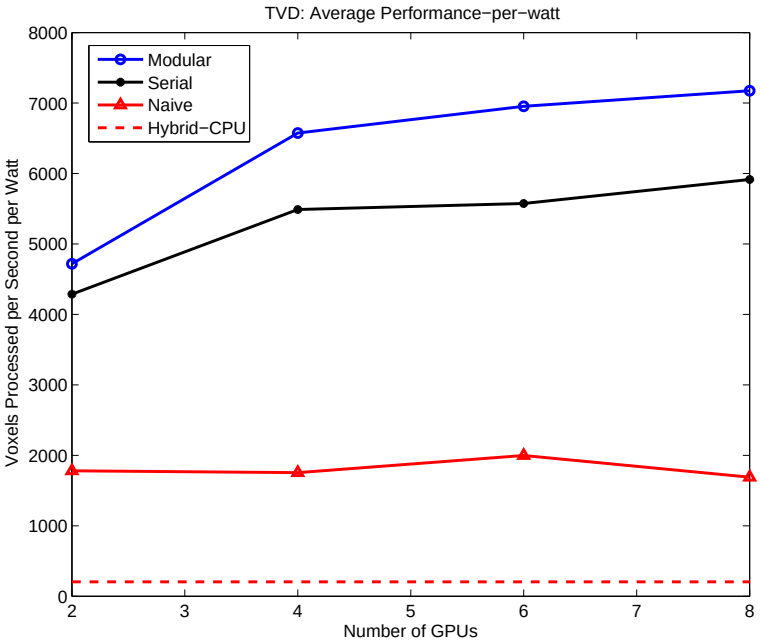

Fig. 4. Performance per watt for the teravoxel reconstruction

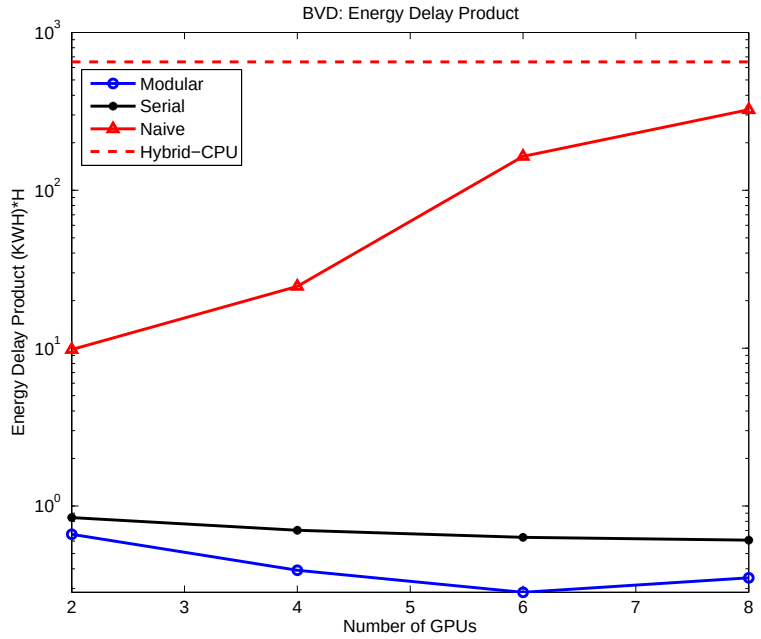

Fig. 5. Energy-Delay Product for the 64 gigavoxel reconstruction

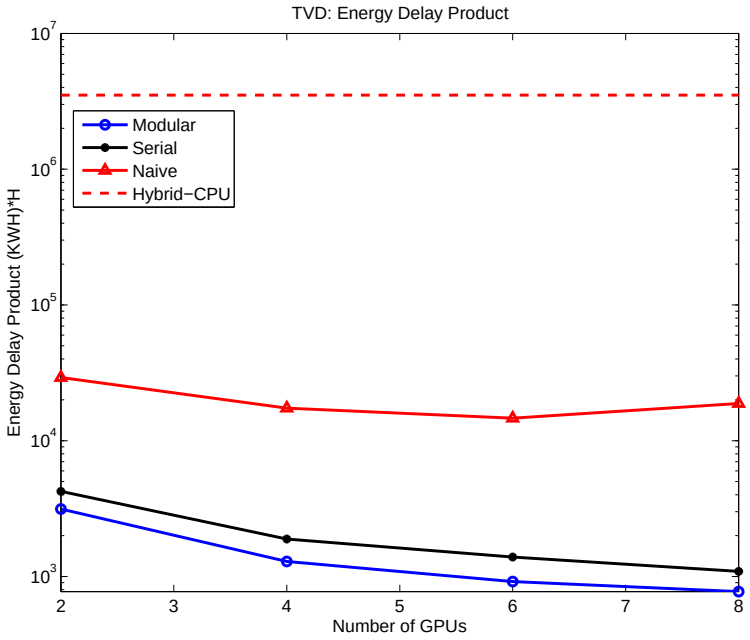

Fig. 6. Energy-Delay Product for the teravoxel reconstruction